



**HAL**  
open science

# Apprentissage profond pour l'analyse de la qualité des pommes de terre

Sofia Marino

► **To cite this version:**

Sofia Marino. Apprentissage profond pour l'analyse de la qualité des pommes de terre. Intelligence artificielle [cs.AI]. Université de Technologie de Troyes, 2020. Français. NNT : 2020TROY0003 . tel-04213487

**HAL Id: tel-04213487**

**<https://theses.hal.science/tel-04213487>**

Submitted on 21 Sep 2023

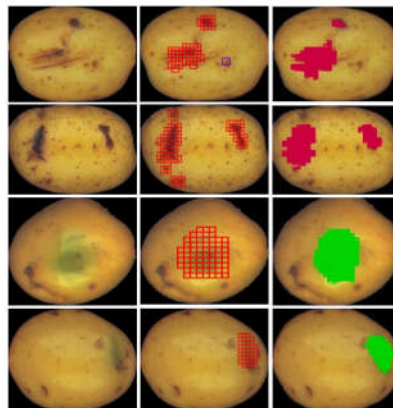
**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse  
de doctorat  
de l'UTT

**Sofia MARINO**

# Apprentissage profond pour l'analyse de la qualité des pommes de terre



**Champ disciplinaire :**  
Sciences pour l'Ingénieur

2020TROY0003

Année 2020



---

---

# THESE

*pour l'obtention du grade de*

## DOCTEUR

de l'UNIVERSITE DE TECHNOLOGIE DE TROYES

EN SCIENCES POUR L'INGENIEUR

**Spécialité : OPTIMISATION ET SURETE DES SYSTEMES**

*présentée et soutenue par*

**Sofia MARINO**

*le 15 mai 2020*

---

---

**Apprentissage profond pour l'analyse  
de la qualité des pommes de terre**

---

---

## JURY

M. Frédéric MORAIN-NICOLIER	PROFESSEUR DES UNIVERSITES	Président
M. Christian GERMAIN	PROFESSEUR DES UNIVERSITES	Rapporteur
M. François ROUSSEAU	PROFESSEUR DES UNIVERSITES	Rapporteur
M. Luc DEROUERS	INGENIEUR	Examineur
Mme Latifa OUKHELLOU	DIRECTRICE DE RECHERCHE IFSTTAR	Examinatrice
M. Pierre BEAUSEROY	PROFESSEUR DES UNIVERSITES	Directeur de thèse
M. André SMOLARZ	MAITRE DE CONFERENCES - HDR	Directeur de thèse

## Remerciements

Je tiens à remercier les nombreuses personnes qui m'ont aidée, accompagnée et encouragée tout au long de mon doctorat. Sans eux, ce travail de thèse n'aurait pas été possible.

Je tiens à remercier M. Christian GERMAIN et M. François ROUSSEAU d'avoir pris le temps de lire et évaluer mon travail. Je voudrais remercier également M. Luc DEROU-  
LERS, M. Frédéric MORAIN-NICOLIER et Mme. Latifa OUKHELLOU d'avoir accepté d'être membres du jury, ainsi que d'examiner le travail accompli.

Je remercie profondément mes deux directeurs de thèse, Pierre BEAUSEROY et André SMOLARZ, de m'avoir donné l'opportunité de faire mon doctorat à l'Université Techno-  
logique de Troyes (UTT). Vos précieux conseils et votre accompagnement constant pendant ces trois ans m'ont permis de mener à bien cette thèse.

Je salue et remercie toute l'équipe d'Eurocelp : Luc DEROU-  
LERS, Christopher FOUQUE, Yvan MARRE, Dominique PETIT et Aymeric ROUSSEAU. La confiance, la patience et le soutien que vous m'avez accordés depuis le début sont inestimables. L'excellente am-  
biance de travail a rendu le chemin parcouru pendant ces trois années nettement plus aisé.

Je remercie sincèrement tous mes collègues et amis qui m'ont accompagnée, aidée et conseillée. Merci « Team colombia », Martin, Nicolas, Ramiro, Nacef, Zied et les docto-  
rants de l'UTT. Je tiens également à m'adresser à tous mes amis qui m'ont accompagnée depuis l'Argentine, merci infiniment pour votre soutien inconditionnel.

Ma gratitude s'étend à toutes les personnes de l'UTT, notamment Véronique, Berna-  
dette, Pascale, Isabelle et Thérèse pour leur compréhension, leur disponibilité et leur aide pendant ces trois années.

Je veux remercier spécialement Samir. Je te l'ai dit peut-être plus d'un million de fois, mais sans ton aide et ton soutien inconditionnel, cela n'aurait pas été possible.

Enfin, je remercie énormément toute ma famille, en particulier mes parents, Claudia et Daniel, et mon frère, Nicolas. Je vous remercie de m'avoir toujours soutenue et encouragée à poursuivre ce projet. C'est grâce à vous que j'ai pu arriver là où je suis aujourd'hui.



## Résumé

La pomme de terre est l'un des produits agricoles les plus consommés dans le monde. L'aspect visuel de ce tubercule est d'une importance fondamentale pour la plupart des consommateurs, qui sont de plus en plus exigeants. Une analyse rigoureuse de la qualité du produit est donc essentielle afin de négocier son prix et de définir le marché auquel il sera destiné. En effet, la qualité des tubercules peut être couramment affectée par divers défauts qui altèrent leur peau. Depuis plusieurs années, des méthodes manuelles ont été appliquées afin de détecter et de classer ces défauts. Cette tâche manuelle engendre quelques inconvénients tels que : un coût élevé, un temps de traitement important et des résultats subjectifs. Par conséquent, le développement de méthodes qui automatisent le contrôle qualité est d'une importance capitale afin d'augmenter l'efficacité, réduire les coûts et obtenir des résultats objectifs qui renforcent la confiance des clients. L'objectif ultime de cette thèse est donc de développer un système de vision artificielle qui soit capable de fournir des informations sur la qualité des échantillons de pommes de terre de manière automatique. Tout d'abord, nous utilisons un système de prise d'images afin de constituer une base de données large et variée. Ensuite, nous proposons et évaluons trois méthodes de classification et de localisation de divers défauts et maladies. Nous avons combiné des techniques traditionnelles d'apprentissage automatique ainsi que des techniques plus récentes reposant sur l'apprentissage profond afin de maximiser les performances de chacune des méthodes proposées. Enfin, afin de veiller au bon fonctionnement du système face à d'éventuels changements de la distribution des données traitées, nous avons proposé une méthode d'adaptation de domaine non supervisée fondée sur l'apprentissage antagoniste. Les résultats expérimentaux obtenus démontrent la pertinence de chacune des méthodes présentées, ainsi que leur applicabilité dans un environnement industriel.

**Mots-clés : Apprentissage profond, Vision par ordinateur, Classification automatique, Qualité des produits, Pommes de terre – Maladies.**

## Abstract

Potato is one of the most widely consumed agricultural produce in the world. The visual appearance of this tuber is of crucial importance to most consumers who are increasingly demanding. In fact, the quality of tubers can be commonly affected by defects or diseases that alter their skin. For several years now, manual methods have been applied to detect and classify these defects. Nevertheless, this manual task is laborious, subjective and time-consuming. Therefore, the development of methods that automate quality control is of paramount importance in order to increase efficiency, reduce costs, and obtain objective results that enhance customer confidence. The main objective of this thesis is to develop a computer vision system that is able to provide information on the quality of potato samples in an automatic way. First of all, an imaging system is used in order

to build up a large and varied image database. Secondly, three methods for classifying and localizing several defects and diseases are proposed and evaluated. Traditional machine learning techniques as well as more recent techniques based on deep learning are combined to maximize the performance of each new proposed approach. Finally, to ensure that the system continues to perform well even if changes occur in the training data distribution, an unsupervised domain adaptation method based on adversarial learning is proposed. The experimental results obtained demonstrate the relevance of each of the proposed methods as well as their applicability in an industrial environment.

**Keywords : Deep learning, Computer vision, Automatic classification, Quality of product, Potatoes–Diseases and pests.**

# Sommaire

<b>Table des figures</b>	<b>ix</b>
<b>Liste des tableaux</b>	<b>xiii</b>
<b>Abréviations</b>	<b>xvii</b>

<b>Chapitre 1</b>	
<b>Introduction Générale</b>	<b>1</b>
1.1 Contexte industriel de la thèse . . . . .	6
1.2 Contributions . . . . .	8
1.3 Organisation de la thèse . . . . .	9
1.4 Publications . . . . .	11
1.4.1 Revue . . . . .	12
1.4.2 Conférence . . . . .	12
1.5 Logiciels . . . . .	12

<b>Chapitre 2</b>	
<b>État de l'art</b>	
2.1 Introduction . . . . .	13
2.2 Acquisition et prétraitement des images . . . . .	16
2.2.1 Acquisition des images . . . . .	16
2.2.2 Prétraitement . . . . .	22
2.3 Modèle de reconnaissance de forme adopté pour la classification des images	26
2.3.1 Méthodes reposant sur une extraction ciblée de caractéristiques . .	26
2.3.2 Méthodes reposant sur l'apprentissage de caractéristiques . . . . .	35
2.4 Conclusion . . . . .	46

**Chapitre 3**

**Apprentissage supervisé pour la classification et la localisation de défauts externes**

3.1	Introduction . . . . .	49
3.2	Description théorique . . . . .	50
3.2.1	Réseaux de neurones convolutifs . . . . .	51
3.2.2	Algorithmes d'optimisation . . . . .	52
3.2.3	Auto-encodeur . . . . .	59
3.2.4	Machine à vecteurs de support . . . . .	62
3.3	Création de la base de données . . . . .	67
3.4	Système proposé reposant sur l'apprentissage supervisé . . . . .	69
3.4.1	Classification des défauts par face de pomme de terre . . . . .	71
3.4.2	Localisation de défauts . . . . .	75
3.4.3	Classification de défauts par gravité . . . . .	75
3.5	Résultats expérimentaux . . . . .	76
3.5.1	Critères d'évaluation . . . . .	77
3.5.2	Résultats de la classification . . . . .	78
3.5.3	Résultats de la localisation de défauts : endommagés et verts . . . . .	82
3.5.4	Résultats de la classification de défauts par gravité . . . . .	85
3.5.5	Résultats de la classification multi-classes multi-étiquettes . . . . .	88
3.6	Conclusion . . . . .	90

**Chapitre 4**

**Apprentissage faiblement supervisé pour la classification et la localisation**

4.1	Introduction . . . . .	94
4.2	Système proposé reposant sur l'apprentissage faiblement supervisé . . . . .	95
4.2.1	Classification des images par défauts . . . . .	97
4.2.2	Localisation des défauts . . . . .	98
4.2.3	Segmentation fine des défauts . . . . .	100
4.2.4	Classification des défauts de pommes de terre par gravité . . . . .	101
4.3	Résultats expérimentaux . . . . .	102
4.3.1	Résultats de l'étape de classification . . . . .	102
4.3.2	Résultats de localisation des défauts . . . . .	103
4.3.3	Résultats de segmentation des défauts . . . . .	104

4.3.4	Résultats de la classification des défauts par gravité . . . . .	105
4.3.5	Évaluation globale du tubercule . . . . .	108
4.4	Réseau entièrement convolutif pour la segmentation sémantique d'images .	111
4.5	Système proposé pour la segmentation sémantique d'images . . . . .	115
4.5.1	Étiquetage automatique au niveau du pixel d'une base de données .	115
4.5.2	Segmentation sémantique d'images de pommes de terre . . . . .	116
4.5.3	Classification de défauts par gravité . . . . .	118
4.6	Résultats expérimentaux . . . . .	119
4.7	Conclusion . . . . .	122

## **Chapitre 5**

### **Adaptation de domaine non supervisé par réseaux de neurones antagonistes**

5.1	Introduction . . . . .	125
5.2	Problème d'adaptation de domaine . . . . .	126
5.3	Adaptation de domaine profonde non supervisée : état de l'art . . . . .	128
5.3.1	Les approches fondées sur la divergence . . . . .	129
5.3.2	Les approches fondées sur les réseaux antagonistes . . . . .	135
5.3.3	Les approches fondées sur la reconstruction . . . . .	140
5.4	Système proposé fondé sur l'apprentissage antagoniste . . . . .	141
5.4.1	Apprentissage du réseau source . . . . .	143
5.4.2	Apprentissage antagoniste pour l'adaptation de domaine . . . . .	143
5.5	Résultats expérimentaux . . . . .	145
5.5.1	Bases de données utilisées . . . . .	145
5.5.2	Méthodes comparatives . . . . .	146
5.5.3	Détails des implémentations . . . . .	148
5.5.4	Résultats sur la base de données avec un changement de luminosité	149
5.5.5	Résultats sur différentes variétés de couleur de pommes de terre . .	152
5.6	Conclusion . . . . .	155

## **Chapitre 6**

### **Conclusions et perspectives**

6.1	Rappel de la problématique traitée . . . . .	157
6.2	Réalisations au cours de la thèse . . . . .	158
6.3	Perspectives et approfondissements . . . . .	160





# Table des figures

1.1	Analyse de répétabilité de résultats sur deux échantillons de pommes de terre. . . . .	4
1.2	Analyse de variabilité et de subjectivité de résultats sur deux échantillons de pommes de terre. . . . .	5
1.3	Machine de contrôle développée par Eurocelp. . . . .	6
1.4	Exemple des images de pommes de terre utilisées dans cette thèse. . . . .	8
2.1	Vue d'ensemble d'un système de vision traditionnel (en haut) et d'un système de vision reposant sur l'apprentissage de caractéristiques (en bas). . .	16
2.2	Différentes techniques d'éclairage. . . . .	21
2.3	Visualisation des poids dans plusieurs couches d'un réseau de neurones convolutifs [1]. . . . .	35
2.4	Résultats de classification du concours ILSVRC depuis 2010 jusqu'à 2017. .	37
2.5	Exemple d'un réseau de neurones convolutif. . . . .	37
2.6	Visualisation des cartes de caractéristiques, des champs réceptifs et des filtres dans une couche de convolution. . . . .	38
2.7	Unité de rectification linéaire ReLU. . . . .	38
2.8	Exemples d'opérations de pooling : max pooling et average pooling avec un filtre de taille $2 \times 2$ et un pas de 2. . . . .	39
2.9	Exemple de l'auto-encodeur entièrement connecté avec une seule couche cachée de $k$ neurones. . . . .	44
3.1	Réseau de neurones convolutifs «LeNet-5» entraîné pour classer des images en 10 catégories [2]. . . . .	52
3.2	Perceptron multicouche avec une seule couche cachée. . . . .	54
3.3	Exemple de l'auto-encodeur avec une seule couche cachée. . . . .	60
3.4	Fonction sigmoïde. . . . .	61
3.5	Fonction tangente hyperbolique. . . . .	61
3.6	Une illustration de l'hyperplan séparant les observations appartenant à deux classes dans un espace bidimensionnel (cas de classes séparables). . .	63
3.7	Une illustration de l'hyperplan séparant les observations appartenant à deux classes dans un espace bidimensionnel (cas de classes non séparables). .	64
3.8	Une illustration de l'hyperplan séparant les observations de l'origine dans un espace bidimensionnel. . . . .	65
3.9	Exemple d'image fournie par le système d'acquisition d'images. . . . .	67

3.10	Image divisée par patches afin d'aider à l'annotation manuelle. . . . .	70
3.11	Schéma de la méthode proposée basée sur l'apprentissage supervisé. . . . .	70
3.12	Architecture du réseau AlexNet [3] (image extraite de [4]). . . . .	71
3.13	Architecture du réseau VGG-16. . . . .	72
3.14	Architecture du réseau GoogLeNet [5]. . . . .	72
3.15	Module d'inception introduit dans GoogLeNet [5]. . . . .	73
3.16	Pomme de terre endommagée où le même défaut est visible sur deux faces. . . . .	76
3.17	Architecture de l'auto-encodeur proposé. . . . .	82
3.18	Comparaison des patches de l'ensemble de test reconstruits par l'auto-encodeur. . . . .	84
3.19	Résultats obtenus lors de la localisation au moyen de l'AE+2C-SVM <sub>DétE</sub> pour les pommes de terre endommagées. . . . .	85
3.20	Résultats obtenus lors de la localisation au moyen de l'AE+2C-SVM <sub>DétV</sub> pour les pommes de terre vertes. . . . .	86
3.21	Image d'une pomme de terre endommagée grave détectée comme saine par la méthode GoogLeNet+AE+2C-SVM <sub>DétE</sub> +2C-SVM <sub>GravE</sub> . . . . .	89
3.22	Exemple d'un pomme de terre avec deux faces classe verte et deux faces classe dartrose. . . . .	89
4.1	Illustration des étapes d'apprentissage et de prédiction dans le contexte d'un apprentissage faiblement supervisé. . . . .	95
4.2	Schéma de la méthode proposée fondée sur l'apprentissage faiblement supervisé. . . . .	96
4.3	Schéma de la génération de la Defect Activation Map dans le but de localiser le défaut classé par le CNN. . . . .	99
4.4	Carte de caractéristiques de sortie de la couche $l$ . . . . .	100
4.5	Exemple des DAMs générées en utilisant le réseau GoogLeNet et GNet-Modif. . . . .	105
4.6	Exemple des DAMs générées avec des valeurs de seuils $h_{DAM}$ différentes. . . . .	106
4.7	Exemple des sorties intermédiaires de la méthode proposée jusqu'au résultat final. . . . .	107
4.8	Résultats de localisation et de segmentation des échantillons appartenant à l'ensemble de test. . . . .	107
4.9	Exemple d'une image représentant une face de pomme de terre contenant deux défauts (endommagé et vert) segmentée par la méthode GNet-Modif+OC-SVM <sub>Seg</sub> +2C-SVM <sub>Grav</sub> . . . . .	109
4.10	Remplacement des couches entièrement connectées par des couches de convolution ce qui permet notamment d'introduire dans le réseau des images d'entrée de taille différentes [6]. . . . .	113
4.11	Opération de convolution et de convolution transposée. Image extraite de l'article de Noh et al. [7]. . . . .	114
4.12	Architecture du réseau entièrement convolutif proposé par Long et al. [6]. . . . .	115
4.13	Méthode proposée pour la segmentation sémantique d'images. . . . .	116
4.14	Exemple des images étiquetées par pixels appartenant à la base de données d'entraînement. . . . .	117
4.15	Modification de l'architecture du GoogLeNet pour obtenir FCN-GoogLeNet. . . . .	118
4.16	Règle de décision pour la classification d'image. . . . .	119

---

4.17	Résultats de la localisation des défauts sur 4 images de test : 2 endommagements et 2 verts. . . . .	122
5.1	Architecture des réseaux proposées par Tzeng et al. [8] afin de résoudre le problème d'adaptation de domaine. Les deux réseaux sont entraînés avec les poids partagés. . . . .	132
5.2	Architecture de DAN proposées par Long et al. [9] afin d'adapter les représentations générées par plusieurs couches. Les réseaux sont entraînés avec partage de poids. . . . .	133
5.3	Architecture de JAN proposée par Long et al. [10]. Le CNN source et le CNN cible partagent leurs poids. . . . .	133
5.4	Architecture de CORAL proposées par Sun et al. [11]. . . . .	135
5.5	Illustration du GAN. . . . .	136
5.6	Architecture CoGAN pour l'adaptation de domaine non supervisée [12]. . .	137
5.7	Architecture du PixelDA proposée par [13]. D=discriminateur, G=generateur et T=classifieur. On cherche à ce que $\mathbf{x}^s$ cible et $\mathbf{x}^f$ généré aient la même distribution. . . . .	138
5.8	Illustration de la méthode DANN proposée par Ganin et al. [14]. . . . .	139
5.9	Diagramme de la méthode DSN proposée par Bousmalis et al. [15]. Les lignes en tirets impliquent le partage de poids. . . . .	140
5.10	Diagramme de la méthode DRCN proposée par Ghifary et al. [15]. . . . .	141
5.11	Schéma de la méthode proposée pour résoudre le problème d'adaptation de domaine non supervisée. . . . .	142
5.12	Images des différentes classes appartenant au domaine source et cible. . . .	146
5.13	Images des différentes classes appartenant au domaine source en haut (variété jaune) et cible en bas (variété rouge). . . . .	147
5.14	Matrices de confusion obtenues à partir des méthodes (a) seulement source, (b) deep CORAL, (c) JAN, (d) JAN, (e) notre méthode, (f) seulement cible.	151
5.15	Visualisation des caractéristiques extraites des images provenant de domaine cible (variété jaune avec un changement de luminosité) en utilisant la méthode t-SNE à partir des différents méthodes (a) Seulement source, (b) Deep CORAL, (c) JAN, (d) ADDA and (e) Notre méthode. . . . .	152
5.16	Matrices de confusion obtenues à partir des méthodes (a) seulement source, (b) deep CORAL, (c) JAN, (d) JAN, (e) notre méthode, (f) seulement cible.	154
5.17	Visualisation des caractéristiques extraites des images provenant de domaine cible (variété rouge) en utilisant la méthode t-SNE à partir des différents méthodes (a) Seulement source, (b) Deep CORAL, (c) JAN, (d) ADDA and (e) Notre méthode. . . . .	155



# Liste des tableaux

1.1	Défauts et maladies à classer avec une description des symptômes visuels. . .	2
1.1	Défaut et maladies à classer avec une description des symptômes visuels (cont.). . . . .	3
2.1	Revue (par ordre chronologique) des caméras utilisées dans les systèmes de vision par ordinateur dans le domaine de l'agriculture. . . . .	18
2.2	Revue (par ordre chronologique) des techniques d'éclairage utilisées dans les systèmes de vision par ordinateur dans le domaine de l'agriculture. . . .	22
2.3	Revue des méthodes reposant sur l'extraction ciblée de caractéristiques les plus significatives dans le domaine agroalimentaire. . . . .	34
2.4	Revue sur les méthodes reposant sur l'apprentissage de caractéristiques les plus pertinents. . . . .	47
3.1	Les fonctions à noyaux les plus couramment utilisées. . . . .	65
3.2	Base de données des images de pommes de terre comprenant les quatre faces du même tubercule en considération (voir figure 3.9). . . . .	68
3.3	Base de données constituée d'images de faces de pommes de terre (chaque face est annotée séparément). . . . .	68
3.4	Caractéristiques de couleur et de texture extraites à partir des images RVB, TSV et CIELAB. . . . .	74
3.5	Moyenne et écart-type de la précision, le rappel et la $F_1$ -mesure obtenues par classe à partir de la validation croisée sur les trois réseaux. . . . .	79
3.6	Matrices de confusion obtenues après l'entraînement d'AlexNet, de VGG-16 et de GoogLeNet. . . . .	80
3.7	Comparaison de la moyenne de la précision, du rappel et de la $F_1$ -mesure obtenues avec des méthodes d'apprentissage de caractéristiques et d'extraction ciblée de caractéristiques. . . . .	81
3.8	Moyenne et écart-type de la précision, le rappel et la $F_1$ -mesure par classe obtenus par les méthodes basées sur l'extraction ciblée de caractéristiques. . . . .	82
3.9	Matrices de confusion obtenues à partir des méthodes basées sur l'extraction ciblée de caractéristiques. . . . .	83
3.10	Résultats de la classification de patchs endommagés versus non-endommagés. . . . .	84
3.11	Résultats de la classification des patchs verts versus non-verts. . . . .	84

3.12	Résultats obtenus lors de la classification par gravité des pommes de terre endommagées en utilisant les deux méthodes : AE+2C-SVM <sub>DétE</sub> +2C-SVM <sub>GravE</sub> et GoogLeNet+AE+2C-SVM <sub>DétE</sub> +2C-SVM <sub>GravE</sub> . . . . .	87
3.13	Résultats obtenus lors de la classification par gravité de pommes de terre vertes en utilisant les deux méthodes : AE+2C-SVM <sub>DétV</sub> +2C-SVM <sub>GravV</sub> et GoogLeNet+AE+2C-SVM <sub>DétV</sub> +2C-SVM <sub>GravV</sub> . . . . .	87
3.14	Matrices de confusion correspondant à la classification des images de pommes de terre en Saine (S), Endommagée Légère (EL) et Endommagée Grave (EG). 88	
3.15	Matrices de confusion correspondant à la classification des images de pommes de terre en Saine (S), Verte Légère (VL) et Verte Grave (VG). . . . .	88
3.16	Résultats obtenus sur la base de données de test multi-classes multi-étiquettes en utilisant la méthode GoogLeNet+AE+2C-SVM <sub>Dét</sub> +2C-SVM <sub>Grav</sub> . . . . .	90
3.17	Résultats obtenus en utilisant le réseau GoogLeNet pour classer les images appartenant à la base de données de test multi-classes multi-étiquettes. . .	90
3.18	Travaux de la littérature proposés pour la classification de pommes de terre catégorisés selon la taille de la base de données utilisée et selon le nombre de catégories à classer. . . . .	91
4.1	Moyenne et écart-type de la précision, le rappel et la F <sub>1</sub> -mesure obtenus en utilisant la validation croisée sur les réseaux GoogLeNet et GNet-Modif. . .	103
4.2	Matrice de confusion obtenue après l'entraînement de GoogLeNet et GNet-Modif. . . . .	104
4.3	Classification des pommes de terre en S=Saines, EL=Endommagée Légère et EG=Endommagée Grave. Matrices de confusion obtenues avec trois configurations différentes : GoogLeNet+DAM+OC-SVM <sub>Seg</sub> +2C-SVM <sub>Grav</sub> , GNet-Modif+DAM+OC-SVM <sub>Seg</sub> +2C-SVM <sub>Grav</sub> et GoogLeNet+DAM+2C-SVM <sub>Grav</sub> sans l'étape de segmentation de grossière à fine. . . . .	108
4.4	Classification des pommes de terre en S=Saines, VL=Verte Légère et VG=Verte Grave. Matrices de confusion obtenues avec trois configurations différentes : GoogLeNet+DAM+OC-SVM <sub>Seg</sub> +2C-SVM <sub>Grav</sub> , GNet-Modif+DAM+OC-SVM <sub>Seg</sub> +2C-SVM <sub>Grav</sub> et GoogLeNet+DAM+2C-SVM <sub>Grav</sub> sans l'étape de segmentation de grossière à fine. . . . .	109
4.5	Précision, rappel et F <sub>1</sub> -mesure des méthodes proposées appliquées sur la base de données de test. . . . .	110
4.6	Précision, rappel et F <sub>1</sub> -mesure des méthodes proposées sur la base de données de test sans la distinction par gravité pour les classes endommagée et verte. . . . .	112
4.7	Précision, rappel et F <sub>1</sub> -mesure obtenus par les méthodes proposées sur la base de données de test. . . . .	120
4.8	Temps de calcul par image pour chacune des méthodes proposées dans le contexte de cette thèse. . . . .	121
5.1	Nombre d'images dans la base de données de pommes de terre jaunes. . . .	146
5.2	Nombre d'images dans la base de données de pommes de terre jaunes (source) et rouges (cible). . . . .	147

- 
- 5.3 Moyenne et écart-type de la précision, le rappel et la F-mesure sur l'ensemble de test des images cibles. Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone. . . . . 150
- 5.4 Moyenne et écart-type de la précision, le rappel et la F-mesure sur l'ensemble de test des images cibles (variété rouge). Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone.153





# Abréviations

2C-SVM	Two-Class Support Vector Machine.
ACP	Analyse en Composantes Principales.
ADDA	Adversarial Discriminative Domain Adaptation.
AE	Auto-encodeur.
ANN	Artificial Neural Network.
BCE	Binary Cross-Entropy.
BDD	Base de données.
CAE	Convolutional AutoEncodeur.
CCD	Charge Coupled Device.
CE	Cross-Entropy.
CIELAB	L'espace chromatique $L^*a^*b^*$ CIE.
CIFRE	Conventions Industrielles de Formation par la Recherche.
CMOS	Complementary Metal-Oxide-Semiconductor.
CNN	Convolutional Neural Networks.
CoGAN	Coupled Generative Adversarial Network.
DA	Discriminant Analysis.
DAM	Defect Activation Map.
DAN	Deep Adaptation Network.
DANN	Domain Adaptation Neural Network.
DL	Deep Learning.
DRCN	Deep Reconstruction-Classification Networks.
DSN	Domain Separation Networks.
EG	Endommagée Grave.
EL	Endommagée Légère.
EM	Ensemble Methods.
FC	Fully-Connected.
FCN	Fully-Convolutional Networks.
FFT	Fast Fourier Transform.

FPA	Focal Plane Array.
GAN	Generative Adversarial Network.
GAP	Global Average-Pooling.
GLCM	Grey Level Co-occurrence Matrix.
GMP	Global Max-Pooling.
GPU	Graphics Processing unit.
ICD	Institut Charles Delaunay.
ILSVRC	ImageNet Large-Scale Visual Recognition Challenge.
IR	InfraRed.
JAN	Joint Adaptation Network.
JMMD	Joint Maximum Mean Discrepancy.
k-NN	K-Nearest Neighbors.
LBP	Local Binary Pattern.
LDA	Linear Discriminant Analysis.
LED	Light Emitting Diode.
M2S	Modélisation et Sûreté des Systèmes.
MCG	Multi-scale Combinatorial Grouping.
MK-MMD	Multiple Kernel Maximum Mean Discrepancy.
MLP	Multilayer Perceptron.
MMD	Maximum Mean Discrepancy.
MSE	Mean Squared Error.
NIR	Near-InfraRed.
NUV	Near Ultra-Violet.
OC-SVM	One-Class Support Vector Machine.
QDA	Quadratic Discriminant Analysis.
ReLU	Rectified Linear Unit.
RF	Random Forest.
RKHS	Reproducing Kernel Hilbert Space.
ROI	Région d'intérêt ou <i>Region Of Interest</i> .
RTN	Residual Transfer Network.
RVB	Rouge, Vert, Bleu.
SAE	Stacked Autoencodeur.
SGD	Stochastic Gradient Descent.
SVM	Support Vector Machine.

TFA	Taux de Fausse Alarme.
TND	Taux de Non Détection.
TPE	Très Petite Entreprise.
TSV	Teinte Saturation Valeur.
UDA	Unsupervised Domain Adaptation.
UDDA	Unsupervised Deep Domain Adaptation.
UTT	Université de Technologie de Troyes.
UV	Ultraviolet.
VG	Verte Grave.
VL	Verte Légère.



# Chapitre 1

## Introduction Générale

### Sommaire

---

<b>1.1</b>	<b>Contexte industriel de la thèse</b>	<b>6</b>
<b>1.2</b>	<b>Contributions</b>	<b>8</b>
<b>1.3</b>	<b>Organisation de la thèse</b>	<b>9</b>
<b>1.4</b>	<b>Publications</b>	<b>11</b>
1.4.1	Revue	12
1.4.2	Conférence	12
<b>1.5</b>	<b>Logiciels</b>	<b>12</b>

---


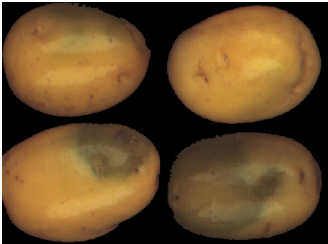
Avec une production mondiale supérieure à 388 millions de tonnes par an [16], la pomme de terre figure parmi les produits agricoles les plus consommés au monde. L'aspect physique de ce tubercule comestible est d'une grande importance afin de déterminer son prix dans les différentes étapes du processus de commercialisation. En effet, les tubercules sont souvent affectés par différents maladies et/ou défauts qui dégradent leur aspect physique. Afin d'appréhender le niveau de cette dégradation et par conséquent, définir un prix de vente correct ainsi que le marché le plus adapté pour les commercialiser, un contrôle qualité minutieux est indispensable. Dans la majorité des cas, le contrôle qualité est effectué manuellement ; un ou plusieurs opérateurs expérimentés font un examen visuel de chaque tubercule afin de lui attribuer des étiquettes reflétant sa qualité.

Le contrôle manuel offre certains avantages, notamment la flexibilité en ce qui concerne les différentes variétés du produit à analyser. Cependant, il existe également des inconvénients majeurs qui viennent altérer le processus de contrôle par les opérateurs humains. Premièrement, la stratégie décisionnelle d'un opérateur peut évoluer au fil du temps en raison de la fatigue et de la monotonie de la tâche, ce qui entraîne un manque de répétabilité. Deuxièmement, les résultats de l'analyse qualité sont généralement subjectifs et dépendent des critères de classification que chaque opérateur adopte pour analyser les produits. Le troisième inconvénient inhérent à la tâche du contrôle qualité manuel est le temps de travail important nécessaire pour effectuer l'analyse de chaque produit. En effet, la précision des résultats est fortement liée au temps consacré à la tâche. Cela signifie que

l'obtention de résultats précis exige un contrôle méticuleux et, par conséquent, lent et coûteux.


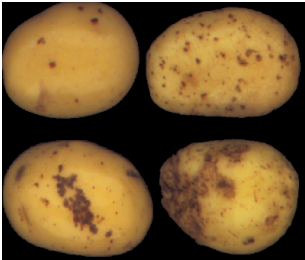

Afin de montrer l'ampleur de cette problématique dans un environnement de travail réel, une expérience dans un laboratoire de contrôle qualité de pommes de terre a été effectuée. Deux opérateurs expérimentés dans le domaine ont été sollicités pour détecter 2 défauts (endommagements et verdissements) et 3 maladies (dartrose, gale commune et rhizoctone) sur deux échantillons de 162 et 120 pommes de terre respectivement. Un exemple de ces défauts et maladies, en détaillant les symptômes visuels de chacun, est présenté dans le tableau 1.1. Chaque opérateur a analysé les deux échantillons et l'opération a été répétée deux fois par chaque opérateur au cours de la même journée afin de mesurer la variabilité des résultats obtenus.

TABLEAU 1.1 – Défauts et maladies à classer avec une description des symptômes visuels.

Classe selon Eurocelp	Exemples	Symptômes
<b>Endommagée</b> (inclut : coupés-blessés, coups d'angle, rongeurs, pourris)		<ul style="list-style-type: none"> <li>• Couleurs variables : chair de la pomme de terre, gris, noir brun, marron, blanc.</li> <li>• Nombreuses formes : rondes, elliptiques, linéaires, courbes.</li> </ul>
<b>Verte</b> (inclut : verdissement dans le champ et verdissement après récolte)		<ul style="list-style-type: none"> <li>• Verdissement dans le champ : tache variable en forme, en taille et en couleur selon le degré d'exposition au soleil.</li> <li>• Verdissement après récolte : voile vert clair homogène et peu dense sur au moins le 1/3 de la face de la pomme de terre.</li> <li>• Les limites de la tache sont difficiles à définir.</li> </ul>

*Suite à la page suivante*

TABLEAU 1.1 – Défaut et maladies à classer avec une description des symptômes visuels (cont.).

Classe selon Eurocelp	Exemples	Symptômes
<b>Dartrose</b> (inclut : dartrose et gale argentée)		<ul style="list-style-type: none"> <li>• Petites taches marron grises ou argentées qui fusionnent souvent faisant des grosses taches assez découpées</li> <li>• A la récolte, certaines taches sont à peine visibles.</li> <li>• Les symptômes évoluent avec le temps.</li> </ul>
<b>Gale commune</b> (inclut : plate et en relief)		<ul style="list-style-type: none"> <li>• Couleurs variables : marron, marron-rougeâtre, foncé ou clair.</li> <li>• Textures variables : plate (ponctuée, liégeuse, rugueuse) ou en relief (pustules creux ou bosse)</li> <li>• Formes variables : points, rondes ou de forme irrégulière.</li> </ul>
<b>Rhizoctone</b> (inclut : plaque type gale commune, crevasses et sclérotés)		<ul style="list-style-type: none"> <li>• Taches très noires.</li> <li>• Les sclérotés (contenant les spores) se collent à la pomme de terre.</li> <li>• Formes variables : points, rondes ou de forme irrégulière.</li> </ul>

Comme on peut l'observer dans la figure 1.1, le manque de répétabilité des résultats est flagrant. À titre d'exemple, un même opérateur est capable de détecter deux fois plus de pommes de terre affectées par la dartrose sur un même échantillon lors de deux analyses effectuées à des moments différents de la journée (figure 1.1c).

Nous avons également analysé à quel point la subjectivité humaine pouvait avoir un



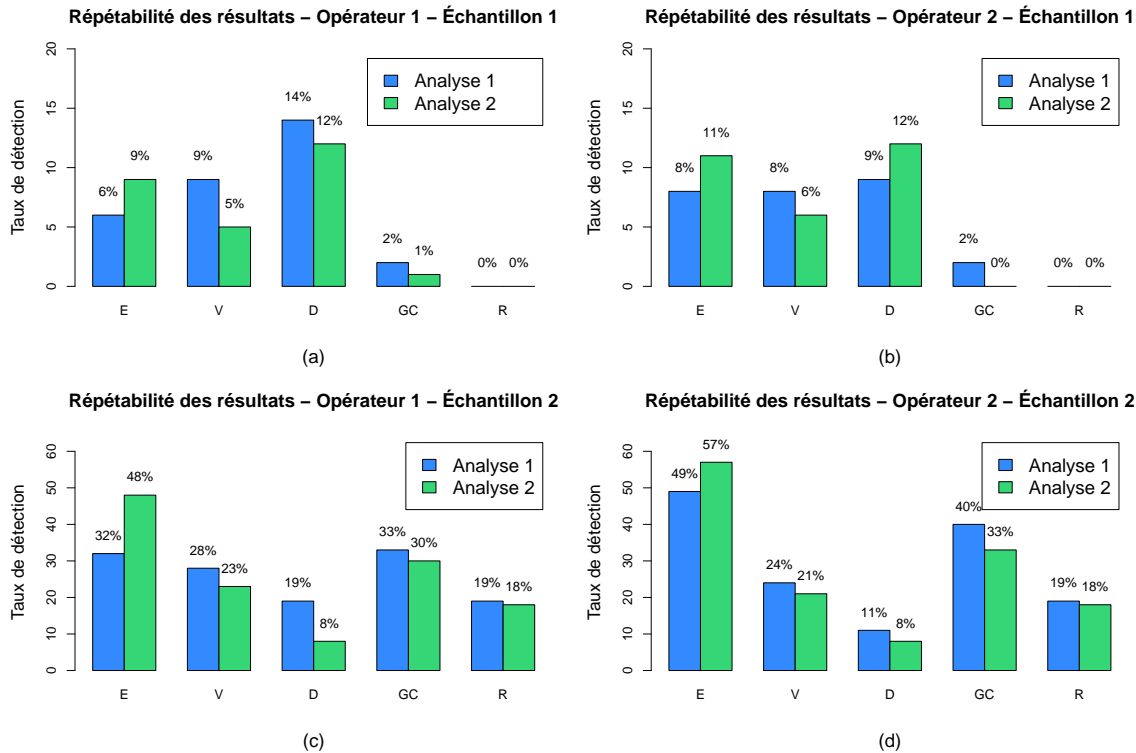


FIGURE 1.1 – Analyse de répétabilité de résultats sur deux échantillons de pommes de terre. Classes : E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone. (a) Résultats des deux analyses sur l'échantillon 1 effectués par l'opérateur 1, (b) Résultats des deux analyses sur l'échantillon 1 effectués par l'opérateur 2, (c) Résultats des deux analyses sur l'échantillon 2 effectués par l'opérateur 1, (d) Résultats des deux analyses sur l'échantillon 2 effectués par l'opérateur 2.

impact négatif sur la fiabilité des résultats. Dans la figure 1.2 nous pouvons remarquer que la quantité de défauts/maladies trouvés par les deux opérateurs dans un même échantillon de pommes de terre varie de façon considérable. À titre d'exemple, l'opérateur 1 a détecté 32% de pommes de terre endommagées sur l'échantillon 2, tandis que le second opérateur en a détecté 49% (figure 1.2b).

Pour faire face à ces inconvénients, un grand nombre de techniques de vision par ordinateur et d'apprentissage automatique ont été proposées dans la littérature afin de classer et de localiser de manière automatique les défauts et les maladies, non seulement dans la pomme de terre, mais dans divers produits agricoles. Les premières méthodes se sont concentrées sur l'extraction ciblée de caractéristiques (*handcrafted features*) combinées avec des classifieurs conventionnels [17, 18, 19, 20, 21, 22, 23]. Le problème majeur de ces systèmes est qu'un extracteur de caractéristiques doit être conçu pour chaque motif d'intérêt afin de transformer l'image brute d'entrée en une représentation exploitable pour réaliser la tâche de classification ou de détection visée. En outre, ces caractéristiques sont adaptées à chaque problème particulier et sont difficilement généralisables.

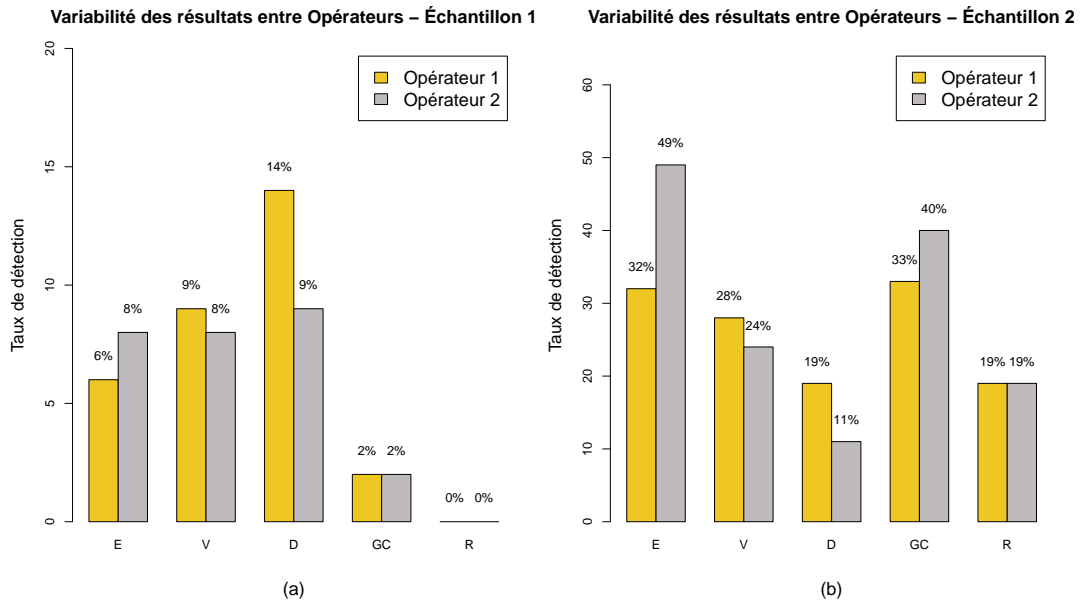


FIGURE 1.2 – Analyse de variabilité et de subjectivité de résultats sur deux échantillons de pommes de terre. Classes : E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone. (a) Résultats obtenus sur l'échantillon 1 par les opérateurs 1 et 2, (b) Résultats obtenus sur l'échantillon 2 par les opérateurs 1 et 2.

Au cours des dernières années, les techniques d'apprentissage profond, ou *Deep Learning* (DL) en anglais, ont commencé à être utilisées dans des systèmes de vision par ordinateur appliqués à l'agriculture [24, 25, 26, 27]. Le DL est un sous-ensemble de l'apprentissage de représentations. Le principal avantage des méthodes d'apprentissage profond est leur capacité à extraire automatiquement des représentations appropriées pour accomplir une tâche ciblée à partir de données brutes. La plupart des méthodes fondées sur le DL, exploitées dans le secteur de l'agriculture, sont des méthodes d'apprentissage supervisé, où la disponibilité d'une large base de données variée et étiquetée est une condition sine qua non. Cependant, il est admis que la conception de telles bases de données peut s'avérer une tâche ardue et particulièrement chronophage.

C'est dans ce cadre complexe que les travaux de cette thèse ont été menés. Nous allons analyser diverses approches pour concevoir des solutions qui permettent d'exploiter le potentiel de l'apprentissage profond dans le domaine de l'agriculture tout en limitant les difficultés inhérentes à la création de bases de données étiquetées importantes. Concrètement, notre objectif principal est le développement de méthodes efficaces reposant sur le DL pour classer et localiser une grande variété de défauts et de maladies qui affectent les pommes de terre. Les contraintes en termes d'accès à des bases de données étiquetées manuellement seront prises en compte dans la proposition des différentes solutions de classification et de localisation. En outre, nous mettrons l'accent sur l'importance d'obtenir des temps de traitement réduits. En effet, les solutions proposées doivent être non seulement plus robustes que l'analyse qualité manuelle, mais également plus efficaces.

## 1.1 Contexte industriel de la thèse

Les travaux de recherche menés dans le contexte de cette thèse ont été réalisés dans le cadre d'une collaboration entre la société Eurocelp et l'équipe Modélisation et Sécurité des Systèmes (M2S) de l'Institut Charles Delaunay (ICD). La collaboration s'est formalisée sous forme d'une Convention Industrielle pour la Formation par la Recherche (CIFRE), subventionnée par l'ANRT.

La société Eurocelp est une entreprise créée en 2012 qui souhaite développer et commercialiser des outils pour l'automatisation du processus de contrôle qualité des pommes de terre. Son ambition est de devenir une référence qualité dans la filière de la pomme de terre, dans un premier temps en Europe et dans un second temps dans le reste du marché mondial. Face à l'absence sur le marché de systèmes capables de fournir des informations objectives et pertinentes sur la qualité des produits agricoles, l'entreprise Eurocelp a initialement mis au point, en collaboration avec l'équipe M2S, une première génération de machines capables de noter la qualité de présentation des pommes de terre de manière automatique (voir figure 1.3). Dans cette machine, les pommes de terre à analyser sont lavées et acheminées à l'aide d'un tapis roulant sous une caméra placée dans un dôme qui prend quatre images pour chaque tubercule. Puis, plusieurs caractéristiques prédéfinies sont extraites des images afin d'obtenir des informations telles que la taille des pommes de terre, le poids et leur qualité de présentation. Cependant, l'usage de cette première



FIGURE 1.3 – Machine de contrôle développée par Eurocelp.

génération de machines est limité à une notation globale de chaque pomme de terre. C'est pourquoi l'entreprise Eurocelp, en proposant cette thèse, s'est fixée comme objectif d'étendre les fonctionnalités de la machine à la classification des défauts et des maladies précédemment détaillés : endommagé, vert, dartrose, gale commune et rhizoctone (voir

tableau 1.1). Pour atteindre cet objectif, il est nécessaire de développer de nouvelles méthodes de classification, de localisation des défauts et d'estimation de leur importance ou gravité.

La création d'un tel dispositif se heurte à de nombreuses difficultés, parmi lesquelles nous pouvons citer les suivantes :

- **Étalonnage des caméras** : dans de nombreux cas, la performance globale du système de vision industrielle dépend fortement de la précision de l'étalonnage de la caméra [28]. En effet, les machines que Eurocelp met sur le marché doivent être en mesure de fournir les mêmes résultats pour un même échantillon. C'est pour cette raison qu'un calibrage précis des caméras est primordial. Même si l'étalonnage ne fait pas partie de ce travail de thèse, il est important de noter que la précision du calibrage a une influence significative sur les résultats de nos travaux.
- **Création de la base de données** : comme nous l'avons vu précédemment, un inconvénient majeur des méthodes à base d'apprentissage profond réside dans la nécessité de disposer d'une large base de données étiquetée. Compte tenu du fait que la création d'une telle base de données est complexe et exige beaucoup de travail, nous nous restreignons à étiqueter un grand nombre d'images de façon globale (un label par image) et un petit ensemble d'images par patches. Ce dernier ensemble est notamment utilisé pour localiser certains défauts.
- **Forte similitude entre certains défauts/maladies** : comme nous pouvons le constater dans la figure 1.4, il existe une grande ressemblance entre certains défauts/maladies, ce qui rend la tâche de classification difficile, même pour un expert. À titre d'exemple, certains endommagements sont très similaires aux maladies de la gale commune ou du rhizoctone, deux maladies qui sont elles-mêmes très semblables. Une confusion est aussi souvent faite entre la dartrose et les pommes de terre saines dont la peau est rugueuse.
- **Variabilité entre un même défaut/maladie** : le fait qu'un même défaut puisse se manifester de nombreuses façons rend sa classification plus complexe. Dans la figure 1.4, nous pouvons observer que les endommagements peuvent se présenter de manières très variées (coupure, blessures, ravageurs, pourriture). La même situation se produit avec la gale commune, qui peut apparaître comme des pustules (d'une couleur foncée ou claire) s'enfonçant en cratères dans les tubercules, ou comme des taches liégeuses superficielles.

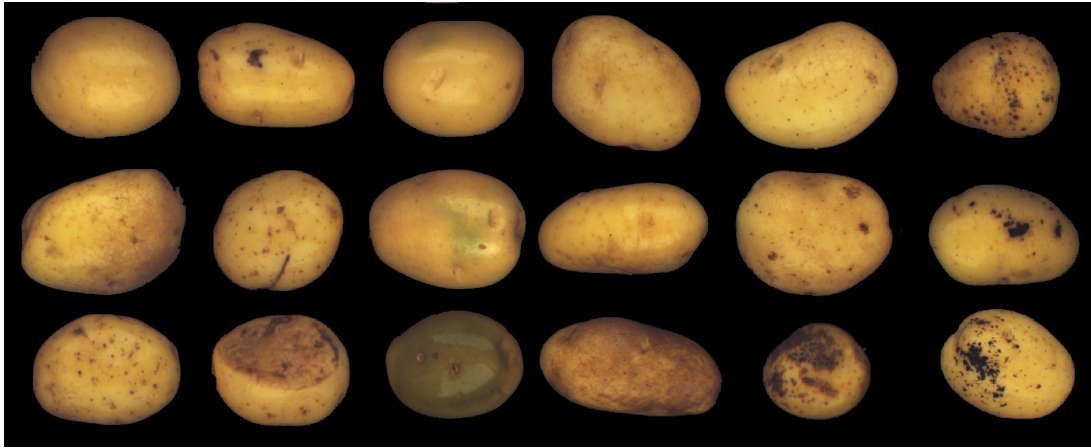


FIGURE 1.4 – Exemple des images de pommes de terre utilisées dans cette thèse. Par colonnes, de gauche à droite les classes : saine (S), endommagée (E), verte (V), dartrose (D), gale commune (GC) et rhizoctone (R).

## 1.2 Contributions

Les principales contributions de cette thèse sont essentiellement liées au développement d'un système de classification de pommes de terre en fonction de certains défauts et maladies à partir de méthodes fondées principalement sur l'apprentissage profond. Certains défauts doivent être également classés selon leur gravité, il est donc essentiel de les localiser ou de les segmenter pour en estimer l'importance. Une approche d'adaptation du domaine non supervisée sera également proposée afin de rendre la méthode de classification plus flexible, c'est-à-dire qu'elle pourra facilement s'adapter aux images provenant d'une nouvelle machine ou aux images représentant une nouvelle variété de pomme de terre différente de celle constituant la base de donnée d'apprentissage du système. Ci-dessous, nous présentons brièvement nos principales contributions, qui sont organisées en 3 chapitres :

1. *Apprentissage supervisé pour la classification et la localisation de défauts externes* : dans de nombreuses tâches de vision par ordinateur, d'excellents résultats ont été obtenus grâce à l'application de méthodes reposant sur l'apprentissage de caractéristiques. En s'appuyant sur ces approches, nous proposons une nouvelle méthode de classification et de localisation de défauts et maladies sur la surface de pommes de terre. La méthode est divisée en trois étapes principales. Premièrement, un réseau de neurones convolutifs (CNN) est entraîné de manière supervisée afin de classer chaque image de pomme de terre en 6 catégories différentes. Ensuite, une combinaison d'un auto-encodeur et d'un classifieur 2C-SVM est utilisée pour classer des patches extraits des images de pommes de terre abîmées, et ainsi localiser les défauts classés. Les informations acquises lors de l'étape de localisation sont ensuite utilisées pour classer les défauts par gravité. Grâce à cette première approche, nous montrons la supériorité des méthodes à base d'apprentissage de représentations, notamment les CNNs, par rapport à celles reposant sur l'extraction ciblée de caractéristiques.

2. *Apprentissage faiblement supervisé pour la classification et la localisation de défauts externes* : bien que les résultats obtenus avec la méthode initialement proposée soient convaincants, la dépendance à une base de données étiquetée par patches pour obtenir une localisation précise de défauts nous a conduit à étudier les méthodes fondées sur un apprentissage faiblement supervisé. Dans ce contexte, l'objectif est de localiser et de segmenter les défauts des pommes de terre en utilisant uniquement une base de données étiquetée au niveau de l'image. Dans cette perspective, nous avons donc proposé une seconde approche à base d'apprentissage faiblement supervisé. Dans cette nouvelle méthode, l'étape de localisation de défauts est réalisée en tirant partie de la capacité du CNN à localiser les motifs des images qui conduisent à la prédiction d'une classe pour obtenir une estimation peu précise de l'emplacement des défauts. Ensuite, une étape de segmentation fine est introduite afin d'améliorer la précision des résultats de localisation précédents.

Dans la deuxième partie du chapitre, nous proposons d'utiliser notre méthode de segmentation faiblement supervisée afin de créer, de manière automatique, une base de données avec des annotations au niveau du pixel. Cet ensemble de données est ensuite utilisé afin d'entraîner de manière supervisée une nouvelle architecture de réseau de neurones convolutifs, capable de classer chaque image comme un tout, ainsi que les pixels qui la composent. Cette approche a deux avantages majeurs : un seul réseau entraînable de bout en bout est utilisé et l'usage de ce réseau réduit considérablement la complexité de la phase d'apprentissage de la méthode. En outre, l'utilisation de ce réseau unique permet aussi une réduction significative du temps de traitement pendant la phase de prédiction, ce qui est essentiel pour notre application.

3. *Adaptation de domaine non supervisé par réseaux de neurones antagonistes* : une fois obtenu un système de classification et de localisation des défauts dont les performances sont conformes aux exigences industrielles, nous poursuivrons nos travaux dans l'hypothèse de futures évolutions du système d'acquisition d'images ou de l'analyse d'une nouvelle variété de pommes de terre. L'occurrence de l'une ou l'autre de ces deux situations conduit à un changement dans la distribution des données. Si rien n'est fait, ce changement de distribution aura pour effet de dégrader les performances du système d'analyse. Dans ce contexte, nous proposons une nouvelle méthode non supervisée d'adaptation du domaine afin d'atténuer la baisse de performance. Cette méthode est fondée sur l'apprentissage antagoniste et a pour objectif principal d'adapter le domaine sans utiliser d'étiquettes pour les images modifiées.

## 1.3 Organisation de la thèse

Ce manuscrit est constitué de quatre chapitres principaux ainsi que d'une introduction et d'une conclusion générale. Nous commençons par donner, au chapitre 2, un aperçu des méthodes de vision par ordinateur et d'apprentissage automatique présentes dans la littérature pour la classification et la localisation des défauts et des maladies de divers produits

agricoles. Le chapitre 3 se focalise sur les approches de classification et de localisation des défauts sur les pommes de terre fondées sur un apprentissage supervisé. Dans le chapitre 4 nous étudions différentes approches permettant notamment d'éviter la construction de bases de données étiquetées pour la localisation et segmentation fine de défauts. Afin de garantir l'adaptabilité et la robustesse des méthodes de classification face à d'éventuelles modifications du système d'acquisition ou des produits à analyser, nous introduisons dans le chapitre 5 les approches d'adaptation de domaine non supervisées. Finalement, dans le chapitre 6, nous concluons les travaux menés en exposant également les perspectives pour les travaux futurs. Une description plus détaillée de chaque chapitre de ce manuscrit est donnée ci-dessous suivie des produits de la recherche.

## **Chapitre 2 : État de l'art**

Dans le deuxième chapitre nous présentons l'état de l'art approfondi de la classification et/ou la localisation des défauts et des maladies pour divers produits agricoles par méthodes fondées sur la vision par ordinateur. Les approches les plus utilisées dans la littérature, y compris les méthodes traditionnelles d'extraction ciblée de caractéristiques et les approches plus récentes fondées sur l'apprentissage de caractéristiques sont introduits. Une attention particulière est portée aux étapes communes aux deux approches, notamment l'acquisition et le prétraitement des images. Ensuite, nous explorons les étapes spécifiques de chaque approche afin d'analyser les avantages et inconvénients de chacune d'entre elles.

## **Chapitre 3 : Apprentissage supervisé pour la classification et la localisation de défauts externes**

Le troisième chapitre est consacré aux approches supervisées de la classification et de la localisation de certains défauts. Le chapitre est divisé en quatre sections principales. Les aspects théoriques nécessaires aux développements de la méthode proposée sont introduits en premier. Nous revisitons les réseaux de neurones convolutifs (CNNs), l'auto-encodeur (AE) et le classifieur par machine à vecteurs supports (SVM). Dans la deuxième section, nous détaillons le processus de création et d'étiquetage de la base de données utilisée pendant nos travaux. Nous exposons ensuite la méthode proposée. Finalement, dans la dernière section, nous analysons de manière approfondie des résultats expérimentaux qui montrent la pertinence de la méthode proposée.

## **Chapitre 4 : Apprentissage faiblement supervisé pour la classification et la localisation de défauts externes**

Le quatrième chapitre traite de deux approches permettant d'aborder la question de la localisation des défauts en relâchant l'exigence d'étiquetage de la base de données d'apprentissage. Dans une première partie nous présentons une méthode reposant sur un apprentissage faiblement supervisée afin de classer, localiser et segmenter des défauts sur

les pommes de terre. Ensuite, nous présentons les réseaux de neurones entièrement convolutifs (FCN) entraînés de manière supervisée afin de réaliser la segmentation sémantique d'images. Nous proposons alors une nouvelle architecture du réseau fondée sur les FCNs afin de classer et de segmenter les images des pommes de terre. La particularité de cette approche est qu'elle utilise dans la phase d'apprentissage une base de données contenant des annotations par pixel générées de manière automatique à l'aide de la méthode faiblement supervisée précédente. Nous présentons également, à la fin de chaque partie, les résultats obtenus en mettant l'accent sur une analyse comparative de toutes les méthodes proposées dans cette thèse.

## **Chapitre 5 : Adaptation de domaine non supervisé par réseaux de neurones antagonistes**

Dans le cinquième chapitre, nous introduisons les méthodes d'adaptation de domaine et nous exposons notre approche utilisant les réseaux de neurones antagonistes. Le début du chapitre donne un état de l'art des méthodes d'adaptation de domaine existant dans la littérature, en se concentrant sur les méthodes non supervisées. Une catégorisation de ces méthodes selon le type d'approche utilisée pour atteindre l'adaptation du domaine est proposée (*discrepancy-based*, *adversarial-based* et *reconstruction-based*). Nous présentons ensuite la méthode que nous proposons en montrant son utilité et son efficacité dans deux situations différentes : dans le premier cas, en changeant la luminosité des images dans le but de simuler une modification de l'éclairage du système d'acquisition, dans un second cas, nous utilisons des pommes de terre jaunes pour entraîner le modèle et nous l'adaptions pour qu'il soit capable de bien classer un ensemble de pommes de terre rouges. Finalement, nous comparons les résultats obtenus à l'aide de cette méthode avec ceux obtenus par d'autres méthodes issues de l'état de l'art.

## **Chapitre 6 : Conclusions et perspectives**

Dans le sixième et dernier chapitre nous présentons les conclusions tirées des travaux réalisés et nous proposons des pistes à explorer pour les travaux futurs.

## **1.4 Publications**

Les publications internationales suivantes sont le résultat des travaux de recherche de cette thèse :



### 1.4.1 Revue

- Marino, S., Beuseroy, P. and Smolarz, A., 2019. Weakly-supervised learning approach for potato defects segmentation. Engineering Applications of Artificial Intelligence, Volume 85, pp.337-346, ISSN : 0952-1976.
- Marino, S., Beuseroy, P. and Smolarz, A., 2020. Unsupervised Adversarial Deep Domain Adaptation Method for Potato Defects Classification. (Accepté par la revue « Computers and Electronics in Agriculture »)

### 1.4.2 Conférence

- Marino, S.; Beuseroy, P. and Smolarz, A. 2019. Deep Learning-based Method for Classifying and Localizing Potato Blemishes. In Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods - Volume 1 : IC-PRAM, ISBN 978-989-758-351-3, pages 107-117. DOI : 10.5220/0007350101070117
- Marino, S., Smolarz, A. and Beuseroy, P., 2019, July. Potato defects classification and localization with convolutional neural networks. In Fourteenth International Conference on Quality Control by Artificial Vision (Vol. 11172, p. 111720G). International Society for Optics and Photonics. (Obtention du prix « **Best paper presentation** » dans la conférence internationale).

## 1.5 Logiciels

Dans le cadre de cette thèse, plusieurs logiciels ont été développés et mis en production :

1. Classification\_localisation\_V1 : logiciel correspondant à la méthode fondée sur l'apprentissage supervisée du chapitre 3.
2. Classification\_localisation\_V2 : logiciel correspondant à la méthode fondée sur l'apprentissage faiblement supervisée du chapitre 4.
3. Classification\_localisation\_V3 : logiciel correspondant à la méthode fondée sur l'apprentissage supervisé du chapitre 4. Ce logiciel est actuellement implémenté dans les différentes machines de l'entreprise.

# Chapitre 2

## État de l’art

### Sommaire

---

<b>2.1</b>	<b>Introduction</b>	<b>13</b>
<b>2.2</b>	<b>Acquisition et prétraitement des images</b>	<b>16</b>
2.2.1	Acquisition des images	16
2.2.2	Prétraitement	22
<b>2.3</b>	<b>Modèle de reconnaissance de forme adopté pour la classification des images</b>	<b>26</b>
2.3.1	Méthodes reposant sur une extraction ciblée de caractéristiques	26
2.3.2	Méthodes reposant sur l’apprentissage de caractéristiques	35
<b>2.4</b>	<b>Conclusion</b>	<b>46</b>

---

## 2.1 Introduction

Durant ces dernières années, l’utilisation de la vision par ordinateur s’est considérablement développée. De nombreuses applications dans des domaines variés ont été proposées et adoptées pour des exploitations à grandes échelles, parmi lesquelles on peut citer : la sécurité [29, 30], la santé [31, 32, 33], le traitement automatique du langage naturel [34, 35, 36], l’assistance à la conduite [37, 38], la gestion du trafic automobile [39, 40], l’agriculture de précision [41, 42, 43], le contrôle et l’évaluation de la qualité de divers produits [44, 45, 46, 22].

La prolifération d’outils de vision par ordinateur sur un aussi large éventail d’applications a principalement été rendu possible grâce aux évolutions récentes de l’apprentissage automatique. En effet, des méthodes qui tentent de reproduire le fonctionnement du système visuel humain ont permis le développement de systèmes de vision par ordinateur, capables d’assimiler le contenu des images numériques et d’en extraire des informations pertinentes. Leur proximité avec la perception humaine permet à ces systèmes d’accomplir de manière autonome une large gamme de tâches telles que la classification [3, 47, 5, 48], la détection [49, 50, 51], la segmentation [52, 53, 54] et même la génération de données [55, 56, 57].

Le contrôle qualité dans le secteur agroalimentaire ne fait pas figure d'exception pour ce qui est de l'intégration d'outils avancés de vision par ordinateur. En effet, les consommateurs sont de plus en plus exigeants quant aux produits à acheter. Ils sont non seulement attentifs à la qualité primaire du produit (goût, provenance, condition de production, etc.), mais également à son aspect visuel. Compte tenu de l'évolution des marchés actuels dans un contexte de mondialisation accrue (diversité des produits et des points d'achats, augmentation de la concurrence, méconnaissance du processus de production des produits agricole, etc.) l'aspect visuel du produit (calibrage, couleur, forme, texture) est devenu un critère fondamental. Bien que cet aspect n'affecte pas toujours le goût ou la physiologie du produit, il peut influencer fortement le choix du consommateur et, par conséquent, le prix de vente.

Malgré l'intégration de la vision par ordinateur dans de nombreuses applications du secteur agroalimentaire, les systèmes existants atteignent leurs limites pour certaines tâches complexes. Compte tenu de la grande variabilité qui caractérise un grand nombre de produits agricoles, leur contrôle qualité est encore majoritairement effectué manuellement : des opérateurs humains qualifiés observent scrupuleusement chaque produit pour lui attribuer une étiquette d'appartenance à une classe selon certains critères prédéfinis. Cependant, de nombreux problèmes remettent en cause la pertinence de ce procédé. D'une part, scruter des produits agricoles à la recherche d'indices visuels pouvant renseigner sur leur qualité est répétitif et lassant pour un opérateur. D'autre part, le classement effectué est subjectif puisqu'il se retrouve extrêmement influencé par de nombreux critères variables d'un opérateur à un autre (les facultés visuelles, le niveau de fatigue, le caractère et l'état d'esprit de la personne, etc.) [58]. Afin de répondre à cette problématique, de nombreux travaux de recherche ont eu pour objectif de proposer des méthodes afin d'automatiser le processus de contrôle qualité des produits agricoles de manière la plus efficace et rentable possible.

Les solutions de vision par ordinateur proposées pour les produits agricoles peuvent être analysées selon deux critères : la nature des résultats recherchés et le modèle de reconnaissance de formes adopté. Selon le premier critère, trois cas peuvent être distingués. Le premier consiste à classer chaque image de produit en fonction de classes prédéfinies [59]. Par exemple, dans le cas particulier de la pomme de terre, réaliser une classification en fonction de la présence ou absence de certains défauts. Le deuxième cas vise non seulement à classer chaque image, mais aussi à fournir des informations sur la localisation du défaut ou de la maladie [60]. Enfin, le troisième et dernier cas concerne la segmentation précise de la forme détectée. Cette approche se démarque de la précédente par une plus grande précision dans la localisation, de telle manière que le résultat final soit une classification de chaque pixel de l'image [21]. Dans nos travaux, nous avons étudié chacune de ces 3 approches de manière objective tout en mettant en évidence leurs avantages et inconvénients.

Pour ce qui est du modèle de reconnaissance de formes, deux approches principales peuvent être distinguées. La première consiste à exploiter un modèle de reconnaissance de formes dit traditionnel. Comme on peut le voir sur la figure 2.1 (en haut), ce modèle est constitué de plusieurs étapes :

- La première étape concerne l'acquisition des images. Le bon déroulement de cette première étape dépend de plusieurs facteurs tels que le type de caméra, l'éclairage et le spectre du rayonnement à capter.
- La seconde étape est relative au prétraitement. Cette étape regroupe tous les traitements secondaires appliqués aux données d'entrée pour améliorer ou faciliter les traitements principaux qu'on désire appliquer aux images. Ces traitements sont souvent utilisés pour diminuer le bruit des images ou améliorer leurs contrastes. De plus, dans cette étape, l'espace couleur à utiliser est défini (par exemple RVB, CIELAB, TSV, etc.), ainsi que la segmentation des régions d'intérêt sur lesquelles les traitements principaux doivent être effectués (ROI).
- L'étape suivante est l'étape d'extraction ciblée de caractéristiques. Une fois le prétraitement effectué, des caractéristiques adaptées à la tâche visée sont sélectionnées et extraites. L'extraction de caractéristiques est utilisée pour obtenir des informations essentielles (couleur, texture, forme) pour accomplir la tâche de reconnaissance de formes qui suit.
- Une fois les descriptifs des images d'entrée obtenus, leur classification est accomplie par l'intermédiaire d'un classifieur entraînable tel que les machines à vecteurs supports (SVM) ou la méthode des k plus proches voisins (k-NN). L'étape de classification est souvent la dernière étape du modèle traditionnel de reconnaissance de formes.

Au cours des dernières années, un deuxième modèle, popularisé par l'apprentissage profond s'est imposé (figure 2.1 en bas). Ce modèle se démarque du modèle traditionnel par la substitution de l'étape d'extraction ciblée de caractéristiques par une étape d'apprentissage automatique de représentations descriptives des données d'entrée [61]. Cette évolution permet, notamment, de se défaire de l'étape de sélection de caractéristiques qui nécessite bien souvent une expertise importante et des connaissances préalables. Ce modèle permet d'obtenir directement, à partir des données brutes, des représentations robustes, adaptées et personnalisées pour la tâche de reconnaissance ciblée.

Dans la suite de ce chapitre, nous présenterons les travaux les plus pertinents associés à la classification et/ou à la localisation des défauts et des maladies dans divers produits agricoles. Les étapes communes entre les deux modèles de reconnaissance de formes (acquisition et prétraitement d'images) seront d'abord introduites. Ensuite, les deux modèles de reconnaissance de formes seront expliqués plus en détail. Cela nous aidera à avoir une vue globale de chacune des deux approches et à analyser les atouts et les limites de leur utilisation dans le contexte de notre application.

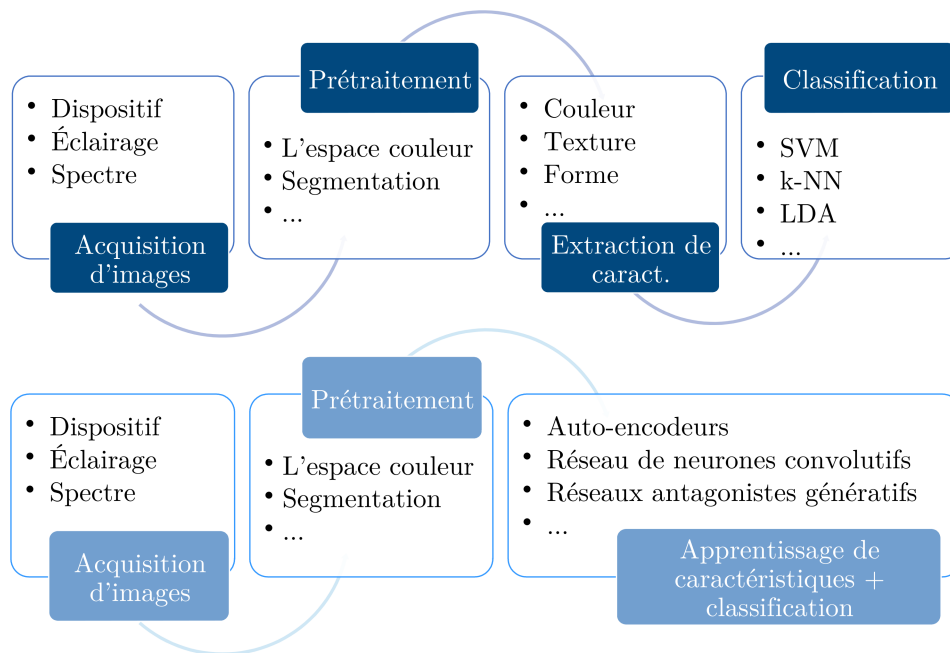


FIGURE 2.1 – Vue d’ensemble d’un système de vision traditionnel (en haut) et d’un système de vision reposant sur l’apprentissage de caractéristiques (en bas).

## 2.2 Acquisition et prétraitement des images

Dans la majorité des systèmes de vision par ordinateur, deux étapes sont indispensables : l’acquisition des images et leur prétraitement. En appliquant successivement ces deux étapes, nous cherchons à obtenir une image de grande qualité pour faciliter son analyse ultérieure.

### 2.2.1 Acquisition des images

La première étape fondamentale d’un système de vision par ordinateur est l’acquisition d’images. Pour accomplir cette étape, nous pouvons souligner deux éléments principaux à définir : la caméra et l’éclairage.

#### Sélection de la caméra

La caméra est un élément important à définir pour tout système de vision par ordinateur. Son objectif est de créer une image en transformant la lumière captée de la scène en signaux électroniques. Il existe plusieurs types de caméras avec des spectres d’observation différents. Le choix de la caméra est généralement fait en fonction de l’application souhaitée.

L’une des caméras les plus utilisées dans les systèmes de vision par ordinateur destinés au secteur agricole est la caméra couleur sensible uniquement à la lumière visible. Celle-ci cherche à imiter la vision de l’œil humain en utilisant trois filtres centrés sur trois longueurs d’onde différentes : le rouge (longueur d’onde 700 nm), le vert (longueur d’onde 546,1 nm) et le bleu (longueur d’onde 425,8 nm) [62, 22, 63, 64, 65]. Une autre catégorie de caméras

fréquemment utilisée dans les systèmes de vision pour le contrôle de la qualité des produits agricoles regroupe les caméras multispectrales et hyperspectrales [66, 67, 68]. En utilisant ces dernières, la réflectance spectrale de chaque pixel est acquise non pas pour une seule valeur mais pour une gamme de longueurs d'onde dans les spectres électromagnétiques. Les longueurs d'onde peuvent inclure les ultraviolets (10 - 400 nanomètres), les régions visibles (400 - 700 nanomètres) et les infrarouges (700 - 1000 nanomètres).

La principale différence entre ces deux types de caméras réside dans le nombre de bandes capturées et leur étroitesse, allant de centaines à des milliers dans le cas de l'hyperspectrale, et de 3 à 10 dans le multispectral. Le fait de pouvoir capturer, à l'aide de ces caméras, de nombreuses images monochromatiques dans le domaine spectral fait qu'elles soient employées dans les cas où les défauts à détecter sont difficiles à observer à partir des images traditionnelles. Cependant, ces caméras ont un coût plus élevé que les caméras traditionnelles, ce qui limite leur usage.

Un autre critère important à considérer pour choisir une caméra est la technique de balayage à adopter : matriciel [21, 69, 70] ou linéaire [71, 72, 20]. Dans le premier cas, une image est capturée en une seule exposition grâce à l'utilisation d'un capteur formé par une matrice de pixels, d'où le nom de caméra matricielle. Celles-ci sont utilisées dans une grande partie des systèmes de vision numérique, car elles sont d'usage simple. Le balayage matriciel est habituellement le plus approprié pour inspecter des objets individuels qui sont caractérisés par des formes ou des bords bien définis, par exemple des fruits ou des légumes. La zone d'intérêt doit pouvoir être contenue dans une seule image. Ces caméras sont moins appropriées quand la zone d'intérêt est ininterrompue, par exemple lorsqu'il s'agit d'un matériau continu tel que du tissu ou des métaux. Avec une caméra matricielle, la zone d'intérêt n'est obtenue qu'en assemblant des images traitées au préalable par un logiciel afin d'éliminer les distorsions possibles. Dans ce dernier cas, le balayage linéaire est plus adapté car l'acquisition de l'image est faite ligne par ligne lorsque l'objet se déplace devant la caméra, sans traitement préalable. Cette technique est généralement utilisée lorsque la zone d'intérêt ne peut pas être capturée grâce à une seule image et/ou lorsqu'il existe des contraintes extrêmes en termes de vitesse et de volume des données.

Dans le choix de la caméra, le type de capteur est également un critère important qui rentre en considération. Les technologies les plus couramment utilisées sont le dispositif à transfert de charge (CCD pour *Charge Coupled Device*) [62, 22, 68] et les semi-conducteurs à oxydes métalliques complémentaires (CMOS pour *Complementary Metal Oxide Semiconductor*) [73, 74]. La finalité des capteurs est de transformer la lumière (photons) en courant électrique (électrons) [75]. Dans les dispositifs CCD, les charges de pixels sensibles à la lumière sont converties en tensions pour ensuite délivrer un signal analogique qui sera numérisé par la caméra. Ces capteurs sont assez simples, mais ils ont l'inconvénient de nécessiter une puce électronique supplémentaire pour effectuer la numérisation des signaux fournis par le capteur, ce qui se traduit par une augmentation du coût ainsi qu'une charge matérielle supplémentaire. En revanche, les capteurs CMOS effectuent la numérisation des pixels en interne à l'aide de transistors que chaque pixel intègre. La puce électronique est donc superflue, ce qui entraîne une réduction des coûts et des équipements généralement plus compacts.

TABLEAU 2.1 – Revue (par ordre chronologique) des caméras utilisées dans les systèmes de vision par ordinateur dans le domaine de l'agriculture.

Publications (par ordre chronologique)	Spectre			Balayage		Capteur	
	Monochro- matique	Couleur Visible	Multi/ Hyperspectrales	Matricielle	Linéaire	CCD	CMOS
[72]	x				x		
[71]		x			x	x	
[62]		x				x	
[76]			x			x	
[20]		x	x		x		
[21]		x		x			
[77]			x				
[66]	x		x		x	x	
[67]			x				
[22]		x		x		x	
[23]		x		x		x	
[69]		x		x		x	
[68]			x			x	
[73]		x		x			x
[74]							x
[63]		x		x			
[78]		x		x			x
[79]		x		x		x	
[70]		x		x			x
[80]		x					x
Remarques	Plusieurs travaux utilisent plus d'une caméra pour prendre des images, et même combinent des caméras RVB et des caméras multispectrales. Quant aux capteurs, les premiers travaux ont utilisé des capteurs CCD et dans les dernières années l'utilisation de capteurs CMOS a augmenté.						

Dans le tableau 2.1, nous listons par ordre chronologique les travaux les plus pertinents réalisés dans le domaine de l'agriculture. Pour chaque méthode, les caractéristiques des caméras utilisées (le spectre, le balayage et le type de capteur) sont renseignées. Nous pouvons observer que l'utilisation des caméras matricielles sensibles à la lumière visible est très répandue, ce qui peut se justifier par son faible coût et sa facilité d'emploi. Pour ce qui est des capteurs, il y a une tendance à remplacer le capteur CCD par le CMOS, de telle sorte qu'on constate que les travaux les plus récents ont pour la plupart recours à ce dernier.

Dans le contexte de cette thèse, le système d'acquisition était déjà défini et installé dans les machines de l'entreprise. Pour des raisons financières, les caméras hyperspectrales

ou multispectrales ont été exclues du système. Une caméra couleur sensible uniquement à la lumière visible a donc été adoptée. En ce qui concerne la technique de balayage, le choix de l'utilisation d'une caméra matricielle a été fait en raison du type d'inspection souhaité. En effet, les pommes de terre ont une forme définie et une seule image est suffisante pour capturer la zone d'intérêt dans son ensemble. Enfin, le capteur CMOS est utilisé en raison de son faible coût et de sa vitesse de lecture. De plus, selon les tendances du marché, on peut prédire un remplacement progressif des capteurs CCD par des capteurs CMOS dans les applications à haute performance. Il est donc judicieux d'intégrer un capteur CMOS pour assurer la durabilité du système et des solutions proposées.

### Sélection de l'éclairage

L'éclairage est un élément clé de tout système de vision par ordinateur. Il doit être soigneusement défini dans le but de mettre en évidence les motifs à inspecter tout en rendant l'environnement aussi homogène que possible pour faciliter l'extraction de descripteurs robustes et représentatifs des zones d'intérêt contenues dans les images. La précision des images obtenues par le système est fortement corrélée avec le niveau et la qualité de l'éclairage [81]. Le choix de la source lumineuse peut être influencé par différents facteurs parmi lesquels on peut citer la géométrie du produit à contrôler, sa taille, ainsi que certaines caractéristiques spécifiques que nous cherchons à visualiser. Les sources d'éclairage le plus utilisées dans les systèmes de vision industrielle sont la lampe incandescente [76, 20], l'halogène à quartz [66, 68], la lampe fluorescente [62, 22, 82] et la LED (*Light Emitting Diode*) [74, 63, 79].

La lumière produite par une lampe à incandescence est due au réchauffement d'un filament de tungstène par le passage du courant électrique. Ce filament est enfermé dans une ampoule de verre pour l'empêcher de se volatiliser en raison des températures élevées. Au fil du temps, le tungstène qui s'évapore se condense progressivement sur la surface intérieure de l'ampoule vitrée, ce qui réduit la transmission lumineuse. De plus, ces lampes ont une consommation d'énergie élevée et leur utilisation est actuellement en baisse. L'une des améliorations apportées aux lampes à incandescence fut l'isolation du filament de tungstène dans une petite enveloppe de quartz remplie d'un gaz halogène (iode ou brome). Cette amélioration se traduit par la célèbre lampe halogène à quartz. Lorsque le tungstène s'évapore du filament, il se combine au gaz halogène pour former un halogénure de tungstène qui ne réagit pas avec l'enveloppe de quartz. Néanmoins, le principal inconvénient de ces lampes est la durée de vie utile qui est d'environ 2000h, nettement inférieure à celle des lampes fluorescentes et LEDs [83].

Les lampes fluorescentes, quant à elles, sont constituées de tubes en verre avec deux électrodes à leurs extrémités. À l'intérieur de ces tubes se trouvent de petites quantités d'un gaz inerte (argon, néon, krypton, xénon ou un mélange de ceux-ci) et du mercure gazeux. Lorsque le courant traverse le tube, le gaz situé entre les électrodes est ionisé et émet un rayonnement ultraviolet (UV). Les rayons UV sont ensuite transformés en lumière visible grâce aux substances fluorescentes présentes sur la surface interne du tube. Bien que les tubes fluorescents coûtent plus cher que les lampes à incandescence, leur faible consommation d'énergie entraîne un gain à long terme. Cependant, pour ce qui est de la durée de vie, elle est plus courte que celle des lampes LED.



Les lampes LEDs sont une des sources lumineuses les plus utilisées pour les systèmes de vision artificielle. Cette source lumineuse peut se présenter sous une large gamme de couleurs et peut être exploitée pour travailler dans les bandes de fréquences proche de l'ultraviolet (NUV) et de l'infrarouge (NIR). Son fonctionnement est principalement fondé sur une diode qui permet à l'énergie de circuler dans une seule direction, en devenant lumière. Cette source lumineuse est très efficace et sûre, car elle ne nécessite que de faibles tensions de fonctionnement. En outre, sa durée de vie peut atteindre 100000h, surpassant la durée de vie des lampes vues précédemment. En plus de leur durabilité, les LEDs fournissent une lumière uniforme de haute intensité avec un rendement sans scintillement et sans émission thermique, produisant un éclairage très brillant adapté à la plupart des systèmes de vision par ordinateur.

En plus de la source lumineuse, il est également nécessaire de déterminer la technique d'éclairage à exploiter. En utilisant une technique adaptée, il sera possible d'obtenir une lumière uniforme sur l'objet à inspecter, ainsi qu'un contraste correct qui rehaussera les caractéristiques visuelle de l'objet à inspecter. Une technique d'éclairage adaptée peut aussi permettre d'estomper les zones de l'image n'appartenant pas à l'objet ciblé. Un éclairage adapté et exploité de manière optimale peut permettre d'éviter l'utilisation d'algorithmes de traitement d'image après l'acquisition. Les techniques les plus couramment utilisées sont les suivantes [84] :

- Éclairage frontal (*Front Lighting*) [70, 79] : la caméra et la lumière, qui se présente généralement sous forme d'un anneaux lumineux, sont positionnées de sorte à faire face à l'objet à inspecter. Cette disposition produit une réduction des ombres et minimise les imperfections propres à l'objet. Ce type d'éclairage a tendance à aplatir l'objet analysé, car il ne projette pratiquement pas d'ombres visibles (figure 2.2a).
- Rétro-éclairage (*Back Lighting*) [85] : la lumière est située à l'arrière de l'objet inspecté ainsi l'objet se retrouve entre la source lumineuse et la caméra. Il est important que toute la surface de l'objet reçoive une lumière uniforme. Cette technique d'éclairage est généralement utilisée pour effectuer des mesures précises à partir de la silhouette de l'objet, sans donner d'information sur ses caractéristiques visuelles de surface (figure 2.2b).
- Éclairage directionnel (*Bright Field Lighting*) [68, 74] : la caméra regarde l'objet inspecté tandis que la source lumineuse est positionnée à un angle de plus de 45 degrés par rapport à l'axe horizontal. Le degré d'inclinaison de la source lumineuse est déterminé en fonction du relief recherché. Ce type d'éclairage convient pour mettre en évidence les bords, les fissures et les rayures dans une direction donnée. Toutefois, elle est limitée à l'inspection de pièces plates (figure 2.2c).
- Éclairage en champ sombre (*Dark Field Lighting*) [73, 80] : la source lumineuse est émise latéralement sous un angle très faible par rapport à l'axe horizontal. La lumière rebondit sur les défauts de l'objet inspecté et tombe directement sur la ca-

méra. Ce type d'éclairage met en évidence les bords de l'objet inspecté, tandis que les surfaces plates restent sombres. Cette technique d'éclairage n'est généralement pas utilisée pour des objets présentant de nombreuses irrégularités dans des directions différentes. (figure 2.2d).

- Éclairage diffus (*Diffuse Lighting*) [62, 63] : cet éclairage est couramment utilisé parce qu'il fournit un éclairage uniforme et une reproduction réaliste de l'objet à inspecter. Pour obtenir ce type d'éclairage, différentes techniques peuvent être combinées, par exemple l'éclairage directionnel et l'éclairage en champ sombre, dans le but d'obtenir une lumière uniforme sur la surface à inspecter (figure 2.2e).

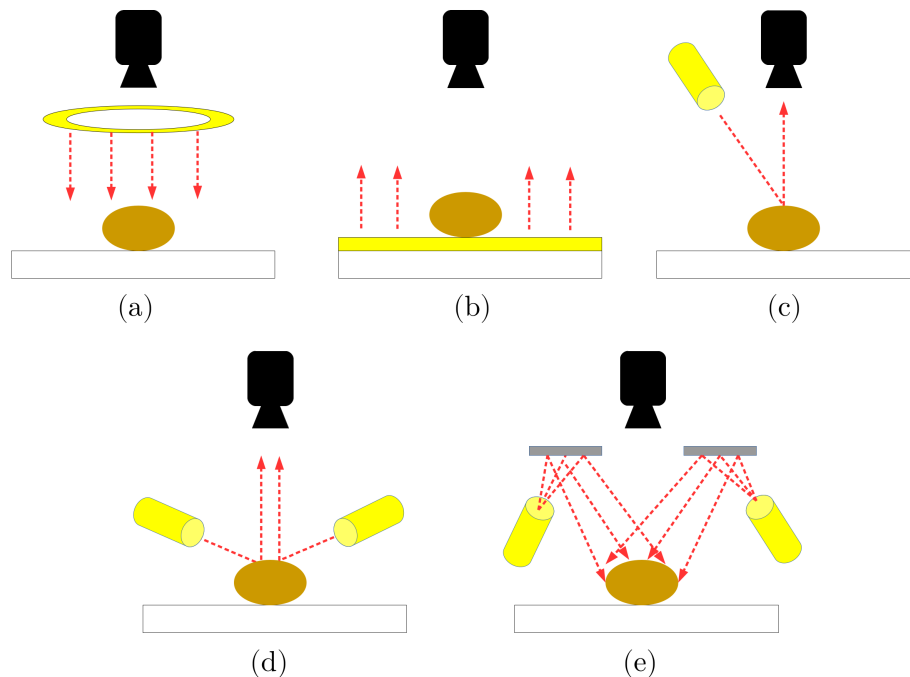


FIGURE 2.2 – Différentes techniques d'éclairage. (a) Éclairage frontal, (b) Rétro-éclairage, (c) Éclairage directionnel, (d) Éclairage en champ sombre, (e) Éclairage diffus.

Les techniques d'éclairage listées ci-dessus sont fréquemment utilisées pour les systèmes fonctionnant dans des conditions d'éclairage contrôlées. Cependant, il existe d'autres systèmes conçus pour être utilisés dans des conditions d'éclairage non contrôlées, l'agriculture de précision est un secteur où ces systèmes sont répandus. Ces systèmes doivent utiliser des algorithmes de traitement supplémentaires qui leur permettent de réduire leur sensibilité aux scénarios d'éclairage [86, 24, 64, 65].

Comme le montre le tableau 2.2, les systèmes de vision développés pour le contrôle de qualité des produits agricoles (systèmes à éclairage contrôlé) utilisent généralement la technique de l'éclairage diffus. De plus, les lampes fluorescentes et les LEDs sont les plus adoptées pour la conception d'un système d'éclairage. Dans notre système, la technique

TABLEAU 2.2 – Revue (par ordre chronologique) des techniques d'éclairage utilisées dans les systèmes de vision par ordinateur dans le domaine de l'agriculture. Inc. = Incandescence, Hal. = Halogène, Flu. = Fluorescente, Fro. = Frontal, Rét. = Rétro-éclairage, Dir. = Directionnel, CS = Champ sombre, Dif. = Diffus

Publications (par ordre chronologique)	Source d'éclairage				Technique				
	Inc.	Hal.	Flu.	LED	Fro.	Rét.	Dir.	CS	Dif.
[71]			x						x
[62]			x						x
[76]	x		x						x
[20]	x								
[85]			x			x			
[66]		x							x
[67]									x
[23]			x						x
[22]			x		x				
[68]		x					x		
[73]								x	
[74]				x			x		
[63]				x					x
[79]				x	x				
[70]				x	x				
[80]								x	
Remarques	Dans la plupart des travaux, un dôme dans lequel se trouve la source d'éclairage est utilisé. De cette façon, le système n'est pas perturbé par des facteurs externes (fluctuations de la lumière naturelle.)								

d'éclairage diffus avec deux panneaux LEDs carrés est utilisée. Cette technique d'éclairage convient à la détection de défauts et d'anomalies car elle assure un éclairage uniforme sur toute la surface des pommes de terre.

### 2.2.2 Prétraitement

Dans le prétraitement des images, on peut distinguer 3 étapes différentes : l'amélioration de la qualité de l'image, la définition de l'espace couleur à utiliser et la segmentation.

#### Amélioration de la qualité de l'image

L'objectif principal de la première étape est d'améliorer la qualité de l'image capturée, afin qu'il soit plus facile d'observer certaines caractéristiques utiles pour le problème spécifique visé. Différentes techniques de traitement d'image sont appliquées afin d'améliorer le contraste, corriger les distorsions, redimensionner l'image, éliminer le bruit, etc. Dans les travaux réalisés dans le domaine de l'agriculture, une grande variété de méthodes de prétraitement ont été appliquées. Pour la réduction du bruit, différents types de filtres

ont été explorés, tels que le filtre médian [23, 87, 78], le filtre gaussien [88, 89, 90, 70], ou les opérations morphologiques [91, 66]. Pour améliorer le contraste, l'égalisation d'histogramme [84] a aussi fréquemment été utilisée. De plus, la plupart des travaux reposant sur l'apprentissage profond effectuent un redimensionnement des images à une taille fixe qui dépend de l'architecture du réseau de neurones employé [92, 59, 93].

### Sélection de l'espace couleur

Une autre caractéristique importante pour la qualité du prétraitement est le choix de l'espace couleur. La couleur est une façon d'interpréter les différentes longueurs d'onde du rayonnement électromagnétique. Dans de nombreux cas, l'information couleur est essentielle pour classer les régions d'intérêt. C'est pour cette raison que la détermination de l'espace couleur à utiliser est d'une importance capitale. Les différents modèles de couleurs les plus couramment utilisés dans le contrôle de la qualité sont les suivants :

- RVB [94, 95, 96, 97] : c'est un modèle chromatique reposant sur la synthèse additive, c'est-à-dire qu'au moyen d'une fusion par addition des trois couleurs rouge, vert et bleu, nous pouvons représenter différentes couleurs. Ce modèle est utilisé par la plupart des caméras traditionnelles en raison de sa facilité d'interprétation. Un avantage de ce modèle est le fait qu'aucune transformation supplémentaire n'est nécessaire après l'obtention de l'image par la caméra. Cependant, la tonalité des couleurs est facilement affectée par les changements d'éclairage. Pour compenser efficacement cette lacune, certains travaux utilisent la version normalisée du RVB dont l'élément de luminosité est supprimé [98].
- TSV [99, 78] : ce modèle repose sur trois composantes, la teinte (T), la saturation (S) et la valeur (Va). Bien que ce modèle soit plus conforme à la vision humaine, il est moins intuitif que le modèle RVB. Un grand avantage de ce modèle est son immunité aux changements d'illumination, car cette dernière se trouve dans la composante de la valeur. En revanche, selon les équations utilisées lors de la transformation d'une image RVB vers le modèle TSV [100], la tonalité peut être indéfinie. En effet, selon l'équation 2.3, lorsque les valeurs RVB maximum et minimum sont identiques, la tonalité devient impossible à calculer. Ceci est généralement résolu en remplaçant la valeur indéfinie de la tonalité par zéro (couleur rouge), ce qui provoque des interprétations incorrectes de la couleur. Le même problème peut se produire avec la saturation lorsque la valeur RVB maximale est zéro (image complètement noire), voir les équations 2.2 et 2.1.

$$Va = \max(R', V', B') \quad (2.1)$$

$$S = \begin{cases} 0 & \text{if } Va = 0 \\ \frac{C}{Va} & \text{si sinon} \end{cases} \quad (2.2)$$

$$T = \begin{cases} 60^\circ \times \left(\frac{V'-B'}{C} \bmod 6\right) & \text{si } Va = R' \\ 60^\circ \times \left(\frac{B'-R'}{C} + 2\right) & \text{si } Va = V' \\ 60^\circ \times \left(\frac{R'-V'}{C} + 4\right) & \text{si } Va = B' \end{cases} \quad (2.3)$$

avec  $Va$  la valeur,  $S$  la saturation,  $T$  la teinte,  $C = (Va - m)$ ,  $m = \min(R', V', B')$ ,  $R' = \frac{R}{255}$ ,  $V' = \frac{V}{255}$ ,  $B' = \frac{B}{255}$ .

- CIELAB ou L\*a\*b [88, 101, 102, 103] : ce modèle se compose de trois éléments : la luminosité (L) ou parfois nommée brillance et deux composantes chromatiques, a et b. Il est souvent utilisé pour améliorer les images couleurs, notamment, pour sa capacité à exprimer une plus large gamme de couleurs que le RVB. Par contre, à l'inverse du RVB dont les couleurs obtenues sont naturelles, le CIELAB génère des images couleur difficiles à interpréter pour l'être humain. De plus, la procédure de transformation des images vers le modèle CIELAB est plus complexe que dans le cas du modèle TSV.

De nombreux travaux utilisent une combinaison de plus d'un espace couleur, afin de tirer avantage de chacun d'eux [104, 21, 105, 87, 106]. D'autres travaux comparent les différents modèles pour définir l'espace couleur le plus approprié pour chaque application particulière [107, 108]. La plupart des méthodes reposant sur l'apprentissage de caractéristiques, notamment l'apprentissage profond, utilisent l'espace couleur RVB [65, 109, 70, 27]. Cela peut s'expliquer par le fait que les réseaux de neurones convolutifs les plus populaires (AlexNet [3], VGG-16 [47], GoogLeNet [5], etc.) ont initialement été appris en utilisant des images RVB. C'est également pour cette raison que les différents travaux proposés dans le cadre de cette thèse nous avons opté pour l'utilisation de l'espace couleur RVB. Grâce à ce choix nous évitons également des transformations d'images vers les espaces TSV ou CIELAB, ce qui réduit la complexité des prétraitements.

## Segmentation de régions d'intérêt

Lors des prétraitements, une étape de segmentation est souvent appliquée. La segmentation consiste à catégoriser les différentes régions d'une même image en fonction de certains critères prédéfinis [110]. Ce traitement permet notamment d'isoler des régions d'intérêt. À titre d'exemple, la segmentation peut être exploitée pour dissocier le produit à inspecter de l'arrière-plan ou afin d'extraire directement des régions d'intérêt, telles que des surfaces affectées par des défauts ou des maladies. Les différentes méthodes utilisées pour la segmentation d'images peuvent être regroupées, selon Brosnan et al. [111], en trois catégories : segmentation par régions, segmentation par seuillage et segmentation par frontières.

Pour ce qui est de la segmentation par régions, la division/fusion (*split and merge*) et la croissance des régions (*grow and merge*) sont les deux approches les plus populaires. La première approche est dite descendante parce qu'elle divise de manière successive une image en régions de plus en plus petites jusqu'à satisfaire certains critères. La deuxième correspond à une approche ascendante avec laquelle on obtient des régions de taille croissante en regroupant les pixels selon des critères d'homogénéité pré-définis. Les algorithmes reposant sur la segmentation par régions figurent parmi les plus utilisés. La popularité de ces algorithmes est principalement due au fait qu'ils ne nécessitent pas d'informations préalables sur la forme à segmenter. À titre d'exemple, Lanthier et al. [112] ont proposé

d'utiliser un algorithme reposant sur la croissance des régions pour segmenter différentes cultures dans des images hyperspectrales de terres cultivées. Les auteurs Dorj et al. [78] ont, quant à eux, utilisé une segmentation par ligne de partage des eaux (*watershed* en anglais), pour différencier des pixels correspondant aux oranges de ceux relatifs aux arbres.

Le deuxième groupe réunit des méthodes de segmentation par seuillage qui sont souvent utilisées pour séparer les objets d'intérêt de l'arrière-plan. Dans ce contexte, le contraste de couleur entre l'objet et l'arrière-plan est d'une importance primordiale pour faciliter la tâche de segmentation. Diverses méthodes de segmentation par seuillage ont été exploitées dans les travaux de la littérature. Par exemple, Riquelme et al. [104] ont proposé d'utiliser une méthode de segmentation par seuillage automatique nommée Otsu, proposée initialement par Otsu et Nobuyuki [113], pour séparer des olives de l'arrière-plan. Dans les travaux de ElMasry et al. [23], un seuil global a été défini sur le canal rouge à partir des images RVB pour séparer les pommes de terre du fond. Tous les pixels d'une valeur inférieure à ce seuil ont été considérés comme appartenant à l'arrière-plan. Les auteurs Moallem et al. [106] ont utilisé un fond sombre pour faciliter la segmentation par seuillage dans le but de séparer les pommes de l'arrière-plan. Le seuil a été défini pour chaque image de manière automatique sur la base d'un histogramme. Les méthodes de segmentation par seuillage sont caractérisées par leur rapidité et facilité de calcul. Cependant, dans certains cas, ces techniques ne sont pas suffisantes pour obtenir une segmentation précise, notamment en raison des variations de contraste et de luminosité qui peuvent exister dans certaines images.

Les approches de segmentation par frontières, quant à elles, visent à détecter les transitions qui surviennent entre deux régions adjacentes. Le détecteur de Canny [114] est souvent utilisé dans ces méthodes pour déterminer les pixels appartenant à un contour. Même si ces méthodes sont robustes et invariantes par rapport à différentes transformations telles que les rotations et les translations, leur utilisation dans les travaux relatifs à des produits agricoles est restreinte, notamment à cause de l'irrégularité et de la complexité des contours de certains produits. À titre d'exemple, les auteurs Thendral et al. [115] ont montré que les résultats de la segmentation des oranges obtenus à partir d'un algorithme par seuillage étaient supérieurs que ceux calculés en utilisant le détecteur de Canny.

Dans le contexte de cette thèse, les images des pommes de terre sont déjà séparées du fond composé d'un tapis roulant noir grâce à un système de vision industriel pré-installé. De ce fait, aucune autre technique de segmentation n'a été utilisée pour isoler le produit de l'arrière-plan. Cependant, comme nous allons voir plus en détails dans la section 3.3, chaque image produite par le système d'acquisition contient 4 clichés, un cliché par face de pomme de terre. Pour cette raison, un traitement spécifique est nécessaire pour extraire chaque face de cette image composite afin de procéder à l'analyse distincte de chacune des faces.

## 2.3 Modèle de reconnaissance de forme adopté pour la classification des images

Une fois le système d'acquisition et les techniques de prétraitement sélectionnés, le traitement principal peut être réalisé sur les images obtenues. En vision par ordinateur, le traitement des images est souvent synonyme de classification, détection et/ou segmentation des motifs contenus dans ces images ou parties d'images. Pour réaliser cette tâche, deux approches majeures peuvent être adoptées. Ces deux approches sont fondamentalement différentes et chacune caractérisée par un modèle de reconnaissance de forme. L'approche traditionnelle est orientée vers une l'extraction ciblée de caractéristiques alors que l'approche fondée sur le modèle d'apprentissage profond préconise, quant à elle, un apprentissage des représentations [1]. Cette section propose un état de l'art de ces deux types d'approches.

### 2.3.1 Méthodes reposant sur une extraction ciblée de caractéristiques

Les premiers travaux de vision par ordinateur dans le domaine de l'agriculture se sont focalisés sur une sélection et extraction de caractéristiques bien définies et personnalisées en fonction du cas traité (*hand-crafted features*). Ces caractéristiques sont ensuite utilisées comme données d'entrée pour des classifieurs conventionnels afin de réaliser la tâche de reconnaissance souhaitée [18, 116, 117, 22, 94, 118, 106, 119]. Dans ce qui suit, nous allons détailler les fondements de ces approches tout en mettant l'accent sur les travaux les plus pertinents l'ayant adopté.

#### Extraction de caractéristiques

Pour la plupart des applications de vision par ordinateur, les images d'entrée brutes ne sont pas directement exploitables. Afin d'y remédier, de nombreuses méthodes permettent de traiter les images dans le but de les projeter dans un nouvel espace de variables restreint où le problème de reconnaissance est plus facile à résoudre. Dans le modèle traditionnel de reconnaissance de forme, cette transformation s'effectue par une extraction ciblée de descripteurs, également appelées attributs ou caractéristiques. Ces derniers contiennent généralement des informations sur la couleur, la texture et la forme des motifs présents sur l'image. Ces descripteurs peuvent être calculés soit à partir de l'image complète, et dans ce cas ils seront appelés caractéristiques globales, ou bien à partir de zones d'intérêt de l'image, formées par un nombre limité de pixels. Dans ce dernier cas, ces descripteurs sont nommés caractéristiques locales.

De nombreuses caractéristiques visuelles sont utilisées dans des applications de vision par ordinateur, la couleur figure parmi les plus utilisées [18, 21, 22, 120, 96, 108]. Cela s'explique notamment par sa facilité d'obtention. En effet, la couleur est une information présente naturellement dans les images brutes [121], son extraction ne nécessite

donc pas de traitements particuliers. La caractéristique couleur peut servir à définir des descripteurs plus aboutis tels que les histogrammes de couleur [122]. Par exemple, un histogramme couleur extrait à partir du canal T des images TSV a été utilisé par Tao et al. [18] pour faire la distinction entre les pommes de terre de couleur standard et les vertes, ainsi qu'entre les pommes vertes et jaunes. Les moments statistiques calculés dans un espace couleur peuvent également être utilisés comme caractéristiques de l'image [123]. Ces moments sont : la moyenne (moment du 1<sup>er</sup> ordre), l'écart type (moment du 2<sup>ème</sup> ordre), l'asymétrie (moment du 3<sup>ème</sup> ordre) et le kurtosis, ou coefficient d'acuité, (moment du 4<sup>ème</sup> ordre). Tous les espaces couleur (RVB, TSV, etc.) peuvent être utilisés pour le calcul de ces moments. À titre d'exemple, les auteurs Pereira et al. [108] ont proposée d'utiliser des caractéristiques couleur afin de prédire la maturation de la papaye. Au total, 21 caractéristiques ont été extraits des images dans plusieurs espaces colorimétriques (RVB, RVB normalisé, TSV et CIELAB) pour représenter chaque fruit.

La texture est également un descripteur populaire en vision par ordinateur. Elle permet d'obtenir des informations sur la distribution spatiale des couleurs ou des intensités dans une image ou dans une région spécifique de l'image [124]. Cette caractéristique peut être obtenue au moyen de différentes méthodes telles que les motifs binaires locaux (*Local Binary Pattern* - LBP) [125], la matrice de co-occurrence des niveaux de gris (*Grey Level Co-occurrence Matrix* - GLCM) [126] et les filtres de Gabor [127]. Même si les caractéristiques de texture sont largement utilisées dans le contexte du contrôle qualité des produits agricoles, elles sont généralement combinées avec d'autres caractéristiques afin de mieux caractériser les motifs visés.

Dans certaines applications où la classification repose sur la géométrie du produit à inspecter, l'utilisation de descripteurs de la forme de l'objet est essentielle. Comme nous l'avons vu dans la section 2.2.2, une étape de segmentation est généralement préconisée afin de différencier l'objet d'intérêt du fond. Une fois cette séparation réalisée, différentes caractéristiques descriptives de la forme de l'objet peuvent être extraites, parmi elles, la superficie, la longueur du périmètre, le centre de gravité, l'axe de moindre inertie, l'énergie de flexion moyenne, l'excentricité, la circularité, la rectangularité, etc. [128].

Malgré l'efficacité des caractéristiques citées précédemment, leur utilisation de manière individuelle reste insuffisante dans la plupart des applications liées au contrôle qualité des produits agricoles ou alimentaires. Afin de répondre efficacement à cette problématique, de nombreux travaux proposent de combiner plusieurs caractéristiques. Razmjoo et al. [22] ont présenté une méthode combinant plusieurs caractéristiques pour trier les pommes de terre par taille et détecter leur défauts. Pour la tâche de tri, des attributs de forme tels que le diamètre maximal, le diamètre minimal et le rapport entre les deux, ont été calculés sur des images dont les objets ont préalablement été séparés du fond. Pour détecter les défauts, des caractéristiques couleur ont été extraites à travers les valeurs R, V, B de chaque pixel de l'image. Prabha et al. [94] ont proposé, quant à eux, de se servir des caractéristiques de couleur et de taille pour détecter le stade de maturité des bananes fraîches. La moyenne, l'écart type, l'asymétrie et le kurtosis ont été calculés à partir des images RVB. Les attributs de forme ont été définis par la surface, la longueur du



périmètre, la largeur de l'axe mineur et la longueur de l'axe majeur de chaque fruit. Une méthode pour détecter des tomates endommagées a été proposée par Arakeri et al. [118]. À partir des images RVB, des caractéristiques de couleur et de texture ont été extraites à l'aide des moments statistiques et de la matrice de co-occurrence des niveaux de gris. Moallem et al. [106] ont proposé une méthode de classement des pommes golden. Des caractéristiques de couleur (la moyenne et l'écart-type des canaux R, V, B et T), de texture (avec la matrice de co-occurrence) et de forme (la proportion de défauts, le périmètre des défauts et la longueur des axes médians des défauts) ont été extraites à partir des défauts préalablement segmentés. Une méthode de classification des maladies sur des plants de pomme de terre a été proposée par Islam et al. [119]. Dix attributs ont été extraits pour représenter chaque image. La matrice de co-occurrence des niveaux de gris a été utilisée pour extraire des caractéristiques de texture, telles que le contraste, la corrélation, l'énergie et l'homogénéité. Outre la texture, les auteurs ont proposé également d'exploiter des caractéristiques de couleur sur la base des histogrammes.

D'après les travaux cités précédemment, nous pouvons constater que la diversité des attributs utilisés est immense.

## Classification

Une fois les caractéristiques retenues extraites, une étape de classification est appliquée. Chaque observation est représentée par un vecteur de caractéristiques qui fournit les informations nécessaires à l'algorithme de classification pour accomplir sa tâche. Dans le domaine de l'apprentissage automatique [129], le processus de classification se compose généralement de deux parties. Tout d'abord, l'apprentissage d'une fonction de décision est réalisé à partir d'un ensemble de données d'apprentissage. Puis, cette fonction apprise est exploitée afin de prédire les classes des nouvelles observations. En fonction des informations exploitées ou des problèmes, les approches d'apprentissage automatique sont communément divisées en deux grandes catégories : supervisé et non supervisé.

Dans les approches d'apprentissage supervisé, la classe réelle  $y \in \mathcal{Y}$  de chaque observation d'apprentissage  $x \in \mathcal{X}$  est connue. Cette information est donc utilisée afin d'étudier la relation existant entre chaque observation d'apprentissage et sa classe. Par conséquent, la fonction de décision  $\varphi$  qui donne la meilleure approximation de  $y$  est trouvée en minimisant le risque suivant :

$$\mathcal{R}(\varphi) = \int \mathcal{J}(\varphi(x), y) P(x, y) dx dy \quad (2.4)$$

où  $\mathcal{J}$  est une fonction de coût qui pénalise l'écart entre la réponse de la fonction  $\varphi(x)$  et  $y$ ,  $P(x, y)$  est la loi jointe des paires de données  $(x, y)$ .

Étant donné que  $P(x, y)$  est généralement inconnue, la minimisation du risque  $\mathcal{R}$  est accomplie en estimant le risque empirique à partir des  $n$  observations de l'ensemble d'apprentissage :

$$\mathcal{R}_{emp}(\varphi) = \frac{1}{n} \sum_{i=1}^n \mathcal{J}(\varphi(x_i), y_i) \quad (2.5)$$

Pour de nombreux cas d'application, la création d'une base de données étiquetées assez conséquente pour permettre d'obtenir des résultats pertinents est souvent très difficile, voire impossible. Pour cette raison, de nombreux travaux se sont focalisés sur le développement de méthodes de classification alternatives, reposant sur un apprentissage non supervisé. Dans cette catégorie d'approches, les observations disponibles  $x$  de l'espace d'entrée  $\mathcal{X}$  n'ont pas d'étiquettes assignées. L'algorithme d'apprentissage non supervisé cherche donc à trouver des relations sous-jacentes entre les observations d'entrée non annotées. La minimisation du risque est alors effectuée afin d'estimer la fonction de décision  $\varphi$  :

$$\mathcal{R}^u(\varphi) = \int \mathcal{J}(\varphi(x))P(x) dx \quad (2.6)$$

où  $\mathcal{J}$  est une fonction de coût et  $P(x)$  la loi de probabilité sur  $\mathcal{X}$ .

Étant donné que  $P(x)$  est inconnue, le risque empirique est estimé à partir des  $n$  observations disponibles dans l'ensemble d'apprentissage :

$$\mathcal{R}_{emp}^u(\varphi) = \frac{1}{n} \sum_{i=1}^n \mathcal{J}(\varphi(x_i)) \quad (2.7)$$

Une fois l'apprentissage (supervisé ou non supervisé) accompli, la fonction de décision  $\varphi$  apprise est exploitée afin de prédire les classes de nouvelles observations non contenues dans l'ensemble d'apprentissage.

Dans la littérature, de nombreuses méthodes de classification supervisées et non supervisées ont été proposées afin de réaliser la classification de produits agricoles et alimentaires à partir de caractéristiques ciblées (*hand-crafted features*). Certaines méthodes utilisent des règles de décision simples, reposant souvent sur des seuillages pour classer les produits. Cependant, d'autres systèmes tirent parti de l'apprentissage automatique pour effectuer des classifications plus abouties. Dans notre travail, nous allons nous focaliser sur ces derniers :

- Machine à vecteur de support (*Support Vector Machine* - SVM) : Le SVM est une méthode d'apprentissage supervisé proposée par Cortes et al. [130] pour résoudre des problèmes de classification ou de régression. Dans le contexte d'une classification binaire, une fonction de décision est construite par le classifieur SVM binaire (ou 2C-SVM) afin de classer les nouvelles observations en deux classes distinctes. Lorsque ces classes sont linéairement séparables, le SVM cherche l'hyperplan optimal qui sépare les observations d'apprentissage des deux classes avec une marge maximale [131]. En d'autres termes, l'objectif est de maximiser la distance minimale entre l'hyperplan (ou frontière de séparation) et les données d'apprentissage. Le fait de maximiser la marge est justifié par la théorie de Vapnik-Chervonenkis [132] qui montre que plus la marge est large, plus la capacité de généralisation du classifieur augmente.

Le SVM a été initialement proposé comme un classifieur linéaire. Néanmoins, dans la majorité des problèmes de classification, les données sont non linéairement sépa-

rables. Afin de contourner ce problème, une transformation non linéaire des données est effectuée. De cette manière, l'espace de données d'entrée est plongé dans un espace de dimension supérieure, nommé espace de redescription, dans lequel les données deviennent linéairement séparables. Pour ce faire, l'astuce du noyau (*kernel trick*) [133] est utilisée. Au moyen de cette technique, chaque produit scalaire dans l'espace de redescription impliqué dans le problème d'optimisation traité est obtenu à l'aide d'une fonction noyau qui est une fonction bivariée des observations dans l'espace d'entrée. Cette fonction doit être définie positive pour assurer la correspondance avec un produit scalaire (théorème de Mercer). En procédant ainsi, la transformation non linéaire appliquée aux données d'entrée ne doit pas être explicitement définie et le calcul est effectué dans l'espace d'origine qui a une dimension inférieur à l'espace de redescription.

Grâce à ses performances, le SVM a été largement exploité pour réaliser des tâches de classification impliquant des produits agroalimentaires. À titre d'exemple, Moallem et al. [106] ont proposé une méthode pour classer des pommes de deux manières : premièrement, les pommes sont classées en deux catégories, saines ou abîmées. Ensuite, les pommes endommagées sont classées par gravité, premier grade, deuxième grade et rejetée. Les caractéristiques de couleur, texture et forme ont été extraites des images et utilisées comme données d'entrées de plusieurs classifieurs : le SVM, le perceptron multicouche (*Multilayer Perceptron* - MLP) et le  $k$  plus proches voisins (*K-Nearest Neighbors* - k-NN). Les meilleurs résultats ont été obtenus à l'aide du SVM avec un taux de bonne détection de 92,5% et 89,2% pour la première et deuxième classification respectivement. Ces résultats ont démontré la supériorité du SVM par rapport aux deux autres classifieurs pour ce cas d'usage. Une méthode de classification de 8 fruits (pomme, poire, mangue, concombre, orange, fraise, ananas et grenade) a été proposée par Jana et al. [96]. Un SVM a été entraîné à partir de 28 attributs de couleur et de texture, obtenant une précision globale satisfaisante. Islam et al. [119] ont utilisé un SVM avec un noyau linéaire pour classer des images RVB de plantes de pomme de terre en trois catégories : saine, alternariose (*early blight*) et mildiou (*late blight*). Ils ont montré qu'avec un temps de calcul raisonnable, ils pouvaient atteindre une précision globale de 93,7%.

- K-plus proches voisins (*K-Nearest Neighbors* - k-NN) : est une méthode simple de classification non-paramétrique, au sens où la règle de décision se déduit d'un estimateur empirique du rapport de vraisemblance sans faire d'hypothèses sur la distribution des données [134]. De plus, cette méthode est fondée sur un apprentissage faible étant donné que la plupart des calculs sont effectués au moment de la classification. Lorsqu'une nouvelle observation est introduite pour être classée, une mesure de similarité entre elle et les données d'apprentissage est calculée. Ensuite, l'étiquette qui apparaît le plus fréquemment parmi les  $k$  observations d'apprentissage les plus proches lui est attribuée. La distance Euclidienne ou la distance de Mahalanobis [135] sont communément utilisées pour mesurer la similarité. Bien que cet algorithme soit simple, sa complexité ainsi que l'espace mémoire requis augmente linéairement avec le volume de données d'apprentissage. Face à un problème de clas-

sification assez simple, la méthode k-NN peut être appliquée avec succès. À titre d'exemple, Ariana et al. [77] ont utilisé la méthode k-NN pour classer chaque pixel d'images hyperspectrales de concombres en 2 catégories : endommagé ou sain. Pour chaque observation, une classe est attribuée à chaque pixel en fonction du label de son plus proche voisin au sens de la distance Euclidienne. Afin de classer des fruits en 5 catégories, Ninawe et al. [120] ont proposé d'exploiter la méthode k-NN avec  $k = 1$ . Pour chaque fruit, des attributs de couleur (valeur moyenne de chaque canal) et de forme (la circularité, la surface et le périmètre) ont été extraits à partir des images RVB afin de constituer l'ensemble d'apprentissage. Ensuite, des nouvelles images ont été classées par rapport au plus proche voisin en employant une distance Euclidienne.

- Analyse discriminante (*Discriminant Analysis* - DA) : est une méthode prédictive et explicative qui sert à étudier le lien existant entre une catégorie  $y$  et des attributs  $X$ . C'est une approche supervisée qui peut être utilisée à la fois pour la classification et la réduction de la dimensionnalité. Dans la phase d'apprentissage, le DA utilise un ensemble d'apprentissage  $X$  afin de définir une fonction de décision pour chaque classe  $y$ . Pour ce qui est de la phase de prédiction, chaque nouvelle observation  $x$  est attribuée à la classe dont la fonction de décision relève la valeur maximale.

L'analyse discriminante linéaire (*Linear Discriminant Analysis* - LDA) et l'analyse discriminante quadratique (*Quadratic Discriminant Analysis* - QDA) sont deux méthodes d'analyse discriminante parmi les plus populaires. Pour ce qui est de la LDA, la séparation entre classes est obtenue grâce à une frontière de décision linéaire. Cette méthode est fondée principalement sur deux hypothèses pour obtenir des résultats optimaux, la multinormalité et l'homoscédasticité. En effet, on suppose que les données de chaque classe sont générées selon une loi normale avec matrices de variance-covariance identiques.

Pour ce qui est de l'analyse discriminante quadratique (*Quadratic Discriminant Analysis* - QDA), la séparation des données est réalisée à l'aide d'une fonction de décision quadratique. Contrairement à la LDA, la QDA ne suppose pas l'égalité des matrices de variance-covariance. De ce fait, pour chaque classe de l'ensemble de données d'apprentissage une matrice de variance-covariance est estimée par maximum de vraisemblance. La complexité du modèle se verra donc affectée par la quantité des classes et par la dimension de l'espace de représentation des observations d'entrée. Dans le contexte du contrôle qualité de produits agricoles, quelques travaux fondés sur l'analyse discriminante ont été proposés. Parmi ces travaux on peut citer celui de Noordam et al. [71] dans lequel un système de contrôle qualité de pommes de terre a été développé. Dans leurs développements, une analyse discriminante linéaire (LDA) a été combinée avec une distance de Mahalanobis afin de classer les pixels d'images RVB de pommes de terre en régions de couleur homogènes (défauts sombres et verdissement). De plus, une analyse comparative avec un perceptron multicouche (MLP) a été réalisée. Bien que la performance globale du MLP soit légèrement supérieure à celle de la LDA, ils ont décidé d'utiliser cette dernière comme technique de segmentation en raison de sa faible complexité.

Parmi les travaux exploitant la DA on peut également citer celui de Blasco et al. [85]. Les auteurs ont proposé une méthode permettant de classer des quartiers de mandarines. Pour ce faire, des caractéristiques morphologiques (facteur de forme, compacité, allongement, longueur, surface, symétrie) et les 10 premières harmoniques de la transformée de Fourier rapide (*Fast Fourier Transform* - FFT) ont été extraites des images. Ces caractéristiques ont été utilisées afin de procéder à une analyse discriminante quadratique pour classer les images en 3 catégories : complète, brisée et demi-quartier. Les résultats obtenus ont été satisfaisants, atteignant un taux de bonne détection globale de 90%.

- Perceptron multicouche (Multilayer Perceptron - MLP) : est un réseau de neurones artificiel utilisé pour résoudre des problèmes de classification ou de régression de manière supervisée. Le MLP est formé de 3 types de couches : une couche d'entrée, une ou plusieurs couches cachées, et une couche de sortie [136]. Dans le cas de réseaux entièrement connectés, chaque neurone d'une couche est connecté à tous les neurones de la couche précédente. À l'intérieur de chaque neurone, un produit scalaire est effectué entre les entrées et les paramètres du modèle (nommés couramment poids). Puis, une fonction d'activation non linéaire est appliquée afin d'obtenir une sortie. Dans la phase d'apprentissage, les poids du réseau sont modifiés de manière itérative dans l'objectif de minimiser l'erreur de prédiction, c'est-à-dire l'erreur entre les étiquettes prédites et les vraies étiquettes contenues dans la base de données. Plusieurs applications reposant sur l'utilisation d'un perceptron multicouche ont été proposées dans le domaine de l'agriculture. Pour segmenter des images de pommes en 4 catégories (arrière-plan, peau saine, peau endommagée et tige) les auteurs Unay et al. [116] ont proposé d'utiliser un MLP composé d'une seule couche cachée. Les valeurs RVB, la matrice de co-occurrence (l'énergie, l'entropie, l'inertie et l'homogénéité) et le coefficient d'ondelette de Coiflet de 2<sup>ème</sup> ordre ont été extraits des images et puis utilisés comme données d'entrée du réseau afin de classer chaque pixel. Pour ce qui est de la segmentation des défauts sur les pommes « Jonagold », une étude comparative entre diverses techniques de seuillage et de classification a été réalisée par Unay et al. [117]. À partir des résultats obtenus, les auteurs ont montré que les performances des classifieurs, supervisés et non supervisés, dépassaient celles des méthodes de seuillage. Comme on peut s'y attendre, les classifieurs supervisés étaient plus précis que les non supervisés. Parmi les méthodes supervisées utilisées (MLP, SVM, LDA, QDA, k-NN), le MLP semblait la plus appropriée pour l'inspection à grande vitesse des pommes.
- Les méthodes d'ensemble (*Ensemble methods* - EM) : l'idée clé derrière ces méthodes est l'obtention de meilleurs résultats en combinant plusieurs modèles (souvent appelés apprenants faibles ou *weak learners*) entraînés pour résoudre le même problème [137]. Ces méthodes reposent sur l'hypothèse que si plusieurs modèles faibles sont combinés correctement, des résultats plus précis devraient être obtenus. N'importe quel classifieur peut être utilisé comme apprenant faible. Cependant, dans la plupart des cas, un seul type d'algorithme d'apprentissage entraîné de multiples manières

différentes est utilisé pour obtenir une classification homogène. Les trois principaux méta-algorithmes qui combinent des apprenants faibles sont le *bagging*, le *boosting* et le *stacking*. Le premier considère souvent des apprenants faibles homogènes entraînés en parallèle (indépendamment les uns des autres). Chaque algorithme est entraîné à partir d'un sous-échantillon de la base de données d'apprentissage. Dans l'étape de prédiction, les résultats de classification sont moyennés pour générer le résultat final. Dans le cas du *boosting*, l'apprentissage des classifieurs est fait de manière séquentielle. Lors de la prédiction, chaque classifieur est pondéré de manière à ce que les plus précis aient un poids plus fort que les autres. Pour chaque prédiction, les observations sont également pondérées en fonction de la prédiction précédente de telle sorte que les *weak learners* se focalisent davantage sur les observations qui n'ont pas été correctement classées. Enfin, le *stacking* est fondé souvent sur des apprenants faibles hétérogènes entraînés en parallèle. Une fois les apprenants faibles entraînés, leurs prédictions sont utilisées pour entraîner un modèle final. Le fait d'avoir plusieurs *weak learners* rend ces méthodes coûteuses du point de vue calculatoire. Dans le contexte des applications agricoles, les auteurs Barnes et al. [21] ont adapté un des algorithmes de *boosting* les plus populaires, Adaboost, afin de détecter des imperfections sur la peau des pommes de terre. Une version de cet algorithme a été proposée (Adaboost minimaliste) de manière à sélectionner un sous-ensemble minimal de caractéristiques extraites (de couleur et de texture) qui maximisent les performances de détection tout en minimisant le coût de calcul. Deux bases de données, pommes de terre jaunes et rouges, étiquetées manuellement au niveau du pixel ont été utilisées. Grâce à cette méthode, des résultats satisfaisants ont été atteints sur les deux bases de données. Pereira et al. [108] ont proposé d'utiliser un méta-algorithme de *bagging*, connu sous l'appellation de forêts aléatoires (*Random Forest* - RF), dans le but de prédire la maturation de la papaye (3 classes). Le RF, avec 500 arbres de décision comme *weak learners*, a été entraîné avec 21 caractéristiques de couleur et de forme. Une précision de 94,3% a été atteinte ce qui est considéré comme un résultat satisfaisant.

La table 2.3 synthétise les techniques de classification et de segmentation reposant sur l'extraction ciblée de caractéristiques. Un avantage certain de ces méthodes est leur capacité à obtenir des résultats satisfaisants même avec une quantité de données d'apprentissage limitée. Cet aspect est particulièrement décisif lorsque l'objectif du système est de fournir une classification de l'image au niveau du pixel, ce qui est fréquent lors de la résolution de problèmes de segmentation d'images. Dans ce cas, l'étiquetage au niveau du pixel de grandes bases d'images est particulièrement ardue et chronophage. Par conséquent, les méthodes nécessitant peu de données d'apprentissage sont avantagées. Néanmoins, la pertinence de ces méthodes est limitée par la difficulté de concevoir un extracteur de caractéristiques spécifique pour chaque problème. Cette tâche demande une expertise humaine et des connaissances pré-requises sur les objets traités afin de pouvoir transformer l'image brute d'entrée en une représentation adéquate pour réaliser la tâche de classification souhaitée.

TABLEAU 2.3 – Revue des méthodes reposant sur l'extraction ciblée de caractéristiques les plus significatives dans le domaine agroalimentaire. C = couleur, T = texture et F = forme. Lorsque plus d'un classifieur est évalué dans un travail, le symbole \* indique la méthode qui donne la meilleure performance.

	Aut.	Prod.	Extr. de caractér.			Classifieur				
			C	T	F	SVM	k-NN	EM	DA	MLP
Classification	[85]	Mandarines (3 classes)			x				x	
	[106]	Pommes (3 classes)	x	x	x	x*	x			x
	[96]	Fruits (8 classes)	x	x		x				
	[120]	Fruits (6 classes)	x	x	x		x			
	[108]	Papaye (3 classes)	x		x			x		
	[138]	Patate (2 classes)		x		x*				x
Segmentation	[119]	Feuilles (3 classes)	x	x		x				
	[21]	Patate (2 classes)	x	x	x			x		
	[22]	Patate (2 classes)	x			x*	x			x
	[71]	Patate (6 classes)	x		x				x	x*
	[117]	Pomme (2 classes)	x			x	x		x	x*
	[116]	Pomme (4 classes)	x	x						x

### 2.3.2 Méthodes reposant sur l'apprentissage de caractéristiques

Afin d'éviter l'étape complexe du choix d'un espace de représentation spécifique pour chaque problème traité, des méthodes d'apprentissage de caractéristiques ont été développées. Le but de ces approches est d'apprendre automatiquement les représentations nécessaires pour effectuer la tâche de classification, de détection ou de segmentation. L'apprentissage de représentations est une approche qui n'est pas récente dans le domaine de l'apprentissage automatique. Cependant, elle a été sous exploitée durant des décennies et c'est grâce à l'avènement de l'apprentissage profond (*Deep Learning* - DL) que cette approche s'est retrouvée propulsée sur le devant de la scène.

Le DL est une sous catégorie de l'apprentissage de représentations fondée sur de multiples niveaux d'abstraction. Ces niveaux sont obtenus en empilant plusieurs modules de transformations non linéaires, où la donnée d'entrée est transformée successivement jusqu'à obtenir une représentation adaptée pour accomplir une tâche ciblée de reconnaissance de formes. En d'autres termes, l'apprentissage profond fait référence à l'utilisation de réseaux de neurones avec plusieurs couches empilées de manière hiérarchique. Ces couches transforment l'une après l'autre la donnée d'entrée, jusqu'à obtenir une représentation de haut niveau susceptible de décrire et de représenter la donnée d'entrée dans toute sa complexité. En effet, plus on avance en profondeur dans le réseau, plus les couches sont susceptibles de détecter des motifs complexes. Comme le montre la figure 2.3, les couches les moins profondes servent à décrire des motifs simples tels que des lignes ou des bords. Les couches intermédiaires, quant à elles, sont en mesure de détecter des combinaisons de motifs primaires extraits par les premières couches, par exemple des carrés en associant plusieurs lignes droites. Finalement, les couches les plus profondes, sont en mesure d'appréhender des concepts plus complexes tels que des objets ou des visages.

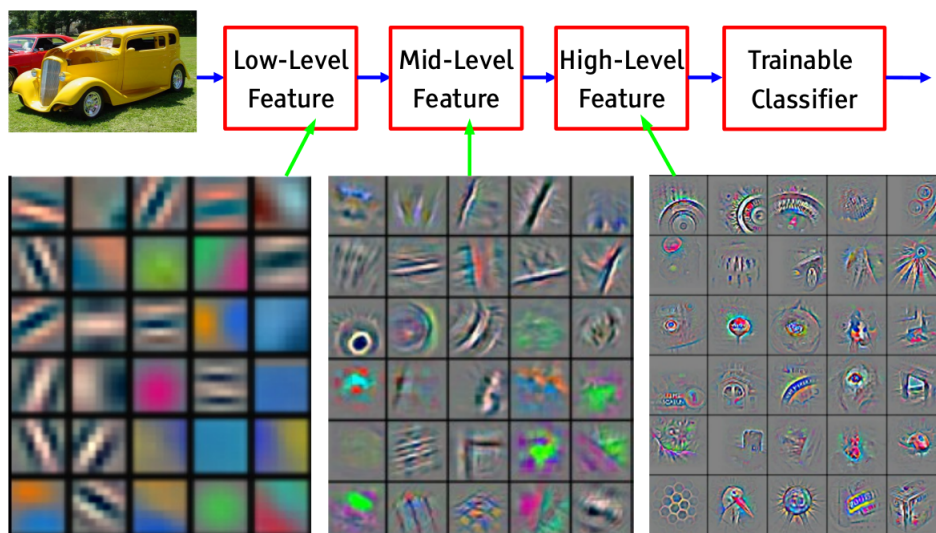


FIGURE 2.3 – Visualisation des poids dans plusieurs couches d'un réseau de neurones convolutifs [1].

Durant ces dernières années, l'apprentissage profond a suscité un intérêt particulier au



sein de la communauté scientifique. Cet intérêt s'est traduit par une augmentation considérable du nombre des travaux impliquant des réseaux de neurones profonds dans divers domaines tels que la classification d'images [3, 47, 5, 48], la détection d'objets [49, 50, 51] et la segmentation d'images [52, 53, 54]. Les méthodes reposant sur le DL ont rapidement été appliquées à l'agriculture, obtenant des résultats prometteurs [24, 25, 26, 27]. Néanmoins, ces méthodes présentent un inconvénient majeur. En effet, elles nécessitent un grand nombre de données étiquetées pour effectuer la tâche de classification ou détection ciblée. D'ailleurs, leur efficacité dans le contexte de la segmentation des défauts et des maladies sur les produits agricoles, repose généralement sur la construction de grandes bases de données étiquetées au niveau du pixel, dont la construction est laborieuse et très complexe.

### Apprentissage supervisé

À l'instar des autres algorithmes d'apprentissage automatique supervisé, les méthodes d'apprentissage profond dites supervisées, nécessitent l'utilisation de bases de données étiquetées pour leur entraînement. Les réseaux de neurones convolutifs (*Convolutional Neural Networks* - CNN) figurent parmi les outils d'apprentissage profond les plus populaires, ils sont largement exploités dans de nombreuses applications de reconnaissance de formes. Le potentiel des CNNs s'est véritablement révélé en 2012, lorsqu'un réseau convolutif profond, connu sous le nom d'AlexNet [3], a remporté la première place du concours international de l'ILSVRC (*ImageNet Large-Scale Visual Recognition Challenge*) [139]. Le fait de gagner la compétition avec une marge de plus de 10,8% par rapport à la méthode ayant obtenu la deuxième place, qui était quant à elle fondée sur le modèle standard de reconnaissance de forme, a permis aux CNNs de démontrer leur efficacité dans les domaines de la vision par ordinateur et de l'apprentissage automatique. Depuis cet événement, tous les concours ILSVRC ont été remportés par des méthodes reposant sur l'apprentissage profond comme nous pouvons le voir dans la figure 2.4.

Le fonctionnement des réseaux de neurones convolutifs a été inspiré par les mécanismes du système visuel des animaux. Comme le montre la figure 2.5, l'architecture d'un CNN est typiquement composée d'une série de couches hiérarchisées. Ces couches sont généralement de trois types différentes : couches de convolution (suivies généralement d'une fonction d'activation), couches de regroupement (ou *pooling*) et couches entièrement connectées.

Les couches de convolution sont l'élément principale de tout réseau de neurones convolutifs, où le plus important travail de calcul est réalisé. Les couches de convolution sont constituées d'un ensemble de paramètres à apprendre, nommés filtres. Chacun de ces filtres est de taille réduite en termes de largeur et de hauteur, mais sa profondeur s'étend sur tout le volume d'entrée. L'objectif de ces couches est d'identifier la présence de différents motifs dans les données d'entrée. Pour ce faire, une convolution est effectuée entre le filtre et une région du volume d'entrée, appelée champ récepteur (*receptive field*). Le filtre est alors translaté sur la totalité de la largeur et la hauteur du volume d'entrée afin d'obtenir comme sortie une carte de caractéristiques (*feature map*). Dans la figure 2.6 nous montrons un exemple de l'opération de convolution dans le contexte des réseaux de neurones. En effet, afin d'obtenir la  $j^{\text{ème}}$  carte de caractéristiques d'une couche de convo-

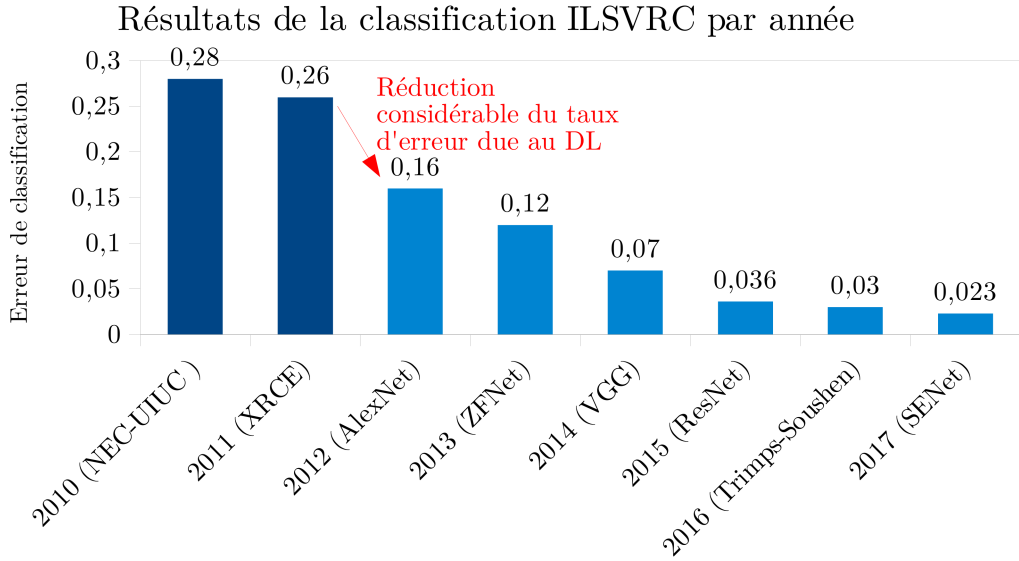


FIGURE 2.4 – Résultats de classification du concours ILSVRC depuis 2010 jusqu'à 2017.

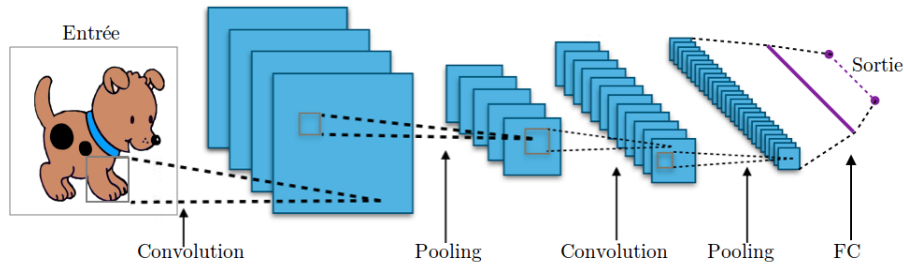


FIGURE 2.5 – Exemple d'un réseau de neurones convolutif.

lution, les  $K$  cartes de caractéristiques constituant le volume d'entrée sont combinées avec  $K$  filtres de convolution bidimensionnels de taille  $S \times S$  et demi taille  $D = \frac{S-1}{2}$ , comme suit :

$$\begin{aligned}
 X_j^{(l+1)}(u, v) &= (X^{(l)} * F^{(l+1)})_j(u, v) \\
 &= \left( \sum_{i=1}^K \sum_{u'=-D}^D \sum_{v'=-D}^D X_i^{(l)}(u - u', v - v') F_{ij}^{(l+1)}(u', v') \right) + b_j \quad (2.8)
 \end{aligned}$$

où  $X_i^{(l)}$  est la  $i^{\text{ème}}$  carte de caractéristiques de la couche  $l$ ,  $F_{ij}^{(l+1)}$  est le  $i^{\text{ème}}$  filtre de la couche de convolution  $l + 1$  utilisé pour obtenir la carte de caractéristiques  $j$  et  $b_j$  le biais associé à la carte de caractéristiques  $j$ . Les cartes de caractéristiques ainsi obtenues sont ensuite empilées afin d'obtenir le volume de sortie.

D'après l'équation 2.8, nous pouvons observer qu'un même filtre de convolution de taille réduite est appliqué sur l'ensemble de la couche précédente, ce qui rend possible le partage de paramètres. Grâce à cette technique, la couche de convolution bénéficie la propriété d'équivariance à la translation. Cela signifie que si l'entrée change, la sortie changera de la même manière.

De manière générale, les couches de convolution sont combinées avec une fonction d'activation non linéaire  $f(\cdot)$  afin d'obtenir l'activation  $Z$  :

$$Z_j^{(l+1)} = f\left(X_j^{(l+1)}\right) \quad (2.9)$$

La fonction d'unité de rectification linéaire (*Rectified Linear Unit* - ReLU) figure parmi les fonctions d'activation les plus populaires dans les architectures de réseaux de neurones modernes (voir figure 2.7).

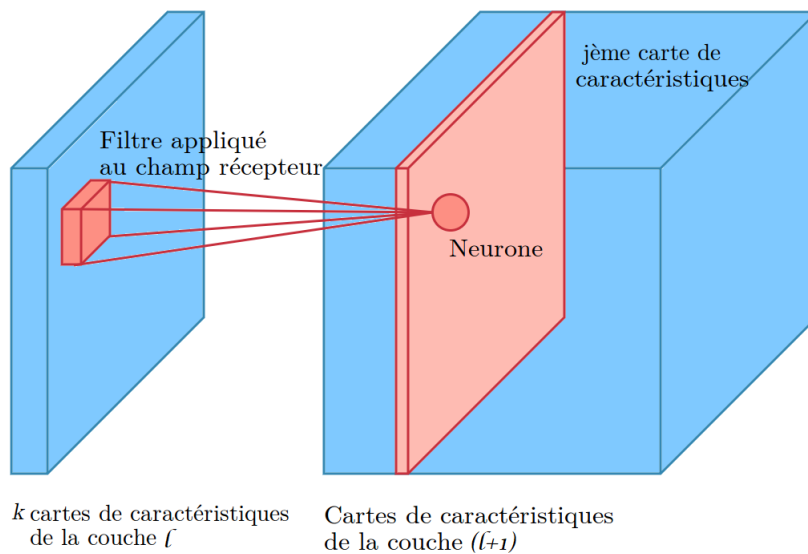


FIGURE 2.6 – Visualisation des cartes de caractéristiques, des champs réceptifs et des filtres dans une couche de convolution.

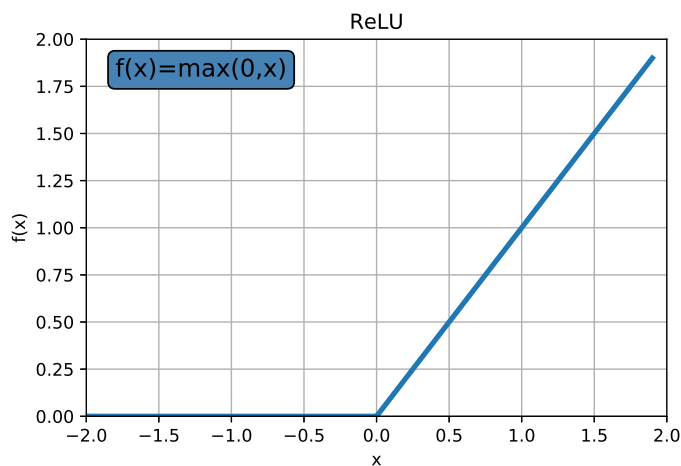


FIGURE 2.7 – Unité de rectification linéaire ReLU.

La couche de regroupement (*pooling*) est également une autre couche fondamentale dans les CNN. Elle est généralement appliquée à la sortie d'un bloc de convolution pour

réduire la dimension des cartes de caractéristiques, ce qui se traduit par une réduction du nombre de paramètres du réseau. De plus, elle est utilisée pour limiter la sensibilité à la translation [140]. La figure 2.8 présente deux exemples parmi les opérations de *pooling* les plus utilisées : le *max-pooling* et l'*average pooling*. Dans la première, un filtre maximal est appliqué aux sous-régions de la carte de sortie de la couche précédente afin de maintenir uniquement les valeurs maximales des sous-régions. La seconde, quant à elle, applique un filtre qui donne la valeur moyenne de chaque sous-région.

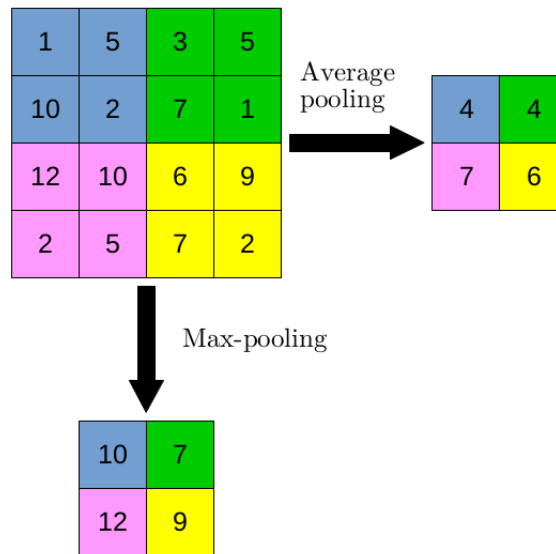


FIGURE 2.8 – Exemples d’opérations de pooling : max pooling et average pooling avec un filtre de taille  $2 \times 2$  et un pas de 2.

Après avoir empilé plusieurs blocs de convolution (couche de convolution suivie d’une activation non linéaire) et de regroupement, les caractéristiques obtenues constituent généralement les entrées d’une ou plusieurs couches entièrement connectées (*Fully-Connected* - FC) afin de partitionner l’espace de représentation appris. Dans une couche entièrement connectée chaque neurone est connecté à la totalité des neurones de la couche précédente. Dans le cas des problèmes de classification, le vecteur de sortie de la dernière couche entièrement connectée passe par une fonction d’activation de type *softmax* et ce dans le but de normaliser la sortie du réseau en une grandeur qui puisse être interprétée comme une distribution de probabilités.

L’apprentissage du CNN est généralement fait avec la méthode d’optimisation du gradient stochastique en utilisant l’algorithme de rétro-propagation : l’erreur entre les sorties et les étiquettes (*ground-truth*) est calculée à partir d’une fonction de perte, telle que l’entropie croisée. Ensuite, le gradient de l’erreur par rapport aux paramètres du réseau est calculé et utilisé pour mettre à jour les paramètres afin de minimiser la fonction de perte.

Les procédures d’apprentissage des CNNs peuvent être divisées en deux groupes : l’apprentissage complet (où *scratch*) et l’apprentissage par adaptation d’un réseau existant

(*finetuning*). Dans le premier cas, les paramètres du réseau sont initialisés de manière aléatoire et l'apprentissage est fait en utilisant une grande quantité de données étiquetées pour éviter le sur-apprentissage (*overfitting*). En raison de la difficulté de disposer d'une base de données suffisamment vaste, le transfert d'apprentissage est souvent utilisé, en particulier le *finetuning*. Ainsi, un réseau de neurones convolutifs est complètement pré-entraîné avec une grande base de données, par exemple ImageNet [141] qui est une base de données publique avec plus de 1,2 million d'images appartenant à 1000 classes différentes. Ensuite, ce réseau pré-entraîné est adapté et utilisé comme point de départ pour parachever l'apprentissage en utilisant la base de données spécifique au problème traité. En procédant ainsi nous évitons l'initialisation aléatoire des paramètres du réseau, ce qui permet de partir d'un modèle plus proche de la solution recherchée et de gagner considérablement de temps de calcul.[142].

De nombreuses méthodes reposant sur des CNNs ont rapidement été proposées et appliquées à l'agriculture. Grâce à ces méthodes, des résultats prometteurs ont été obtenus dans de nombreuses tâches [92, 25, 26, 27]. Mohanty et al. [92] ont utilisé une base de données publique (PlantVillage [143]) constituée de plus de 54000 images dans le but d'entraîner des réseaux de neurones convolutifs à identifier 26 maladies sur 14 espèces de plantes différentes. Deux architectures populaires, toutes deux gagnantes d'une édition du concours ILSVRC, ont été évalués : AlexNet [3] et GoogLeNet [5]. Dans cette évaluation, deux protocoles ont été appliqués aux deux réseaux, le premier consistait à appliquer un apprentissage complet et le deuxième à commencer l'apprentissage à partir des paramètres existant et à les affiner grâce à du *finetuning*. Finalement, les résultats ont démontré, dans ce contexte, la supériorité de GoogLeNet par rapport à AlexNet, ainsi que celle du *finetuning* par rapport à de l'apprentissage complet. Ma et al. [25] ont présenté un travail dans lequel une base de données comprenant 1184 images de plantes de concombres a été créée. Cette base de données a ensuite été utilisée pour l'apprentissage complet d'une nouvelle architecture dérivée du réseau LeNet [2]. Ce CNN permet d'effectuer la classification des images en fonction de 4 maladies pouvant affecter les concombres. En plus de ce réseau, ils ont appliqué du *finetuning* sur le réseau pré-entraîné AlexNet, montrant que ce dernier atteignait une précision supérieure à celle du premier réseau. Une variante de l'architecture ResNet50 [144] a été proposée par Picon et al. [26] pour classer 3 maladies communément présentes dans le blé. Une première étape d'apprentissage complet a été réalisée en utilisant une base de données composée d'ImageNet et d'images de 56 espèces différentes de plantes. Les paramètres de ce réseau ont ensuite été affinés grâce à du *finetuning* en utilisant une seconde base d'images, composée quant à elle de 8178 images de plantes de blé étiquetées par des experts. Grâce à ces travaux les auteurs ont mis en lumière la pertinence des méthodes reposant sur l'apprentissage profond pour la détection et la classification de maladies affectant les plantes. Ils ont également montré l'efficacité de ces méthodes appliquées à des images prises dans des conditions d'éclairage non contrôlées.

D'après les travaux cités ci-dessus, on peut conclure que des résultats de classification plus satisfaisants ont été obtenus en appliquant la technique de *finetuning* aux réseaux de neurones pré-entraînés sur des grands ensembles de données, tel que ImageNet. Comme nous avons vu précédemment, cette technique est notamment utilisée lorsque la base de

données d'apprentissage disponible n'est pas assez vaste pour entraîner le réseau de zéro avec succès. En effet, un réseau pré-entraîné sur un grand ensemble de données a appris à détecter des motifs universels comme des courbes et des bords dans ses premières couches. Puisque ces caractéristiques apprises restent pertinentes et utiles pour la plupart des problèmes de classification d'images, l'application du *finetuning* rend l'optimisation plus rapide et réduit la quantité de données annotées nécessaires afin d'entraîner des nouveaux réseaux.

On peut également noter que les problèmes analysés reposent fondamentalement sur la classification au niveau de l'image. En effet, les différentes méthodes décrites ci-dessus ne sont pas en mesure de fournir des informations concernant la localisation des défauts ou des maladies détectés. La plupart des méthodes reposant sur l'apprentissage profond dans le domaine de l'agriculture n'effectuent la classification qu'au niveau global de l'image. Cela peut notamment s'expliquer par les difficultés qu'engendre la création d'une base de données large et variée avec des annotations au niveau du pixel ou avec des informations sur la localisation des défauts à l'aide de boîtes englobantes (*bounding boxes*). À titre d'exemple, les auteurs de [64] ont dû étiqueter 5000 images avec plus de 43000 boîtes englobantes pour localiser et classer différentes maladies et ravageurs des plants de tomate. Grâce à cette base de données, ils ont pu apprendre et comparer trois réseaux de neurones convolutifs (Faster R-CNN [49], R-FCN [50] et SSD [145]), créés initialement pour la détection d'objets. Comme nous pouvons le concevoir, la création de telles bases de données n'est pas une tâche anodine compte tenu du temps et de l'expertise requises.

#### **Apprentissage faiblement supervisé**

Dans le domaine du contrôle qualité des produits agricoles à l'aide de systèmes de vision par ordinateur, la majorité des travaux proposés œuvrent au niveau de l'image. Cependant de nombreuses tâches nécessitent que la reconnaissance de formes se fasse sur des régions plus restreintes que l'image (patches ou pixels). Le cas de la localisation et de la segmentation des défauts en est le parfait exemple. En effet, localiser de manière exacte l'endommagement ou la maladie qui peut affecter un produit permet d'établir avec précision l'étendue de la dégradation subie. Néanmoins, comme cela a été énoncé précédemment, la création de bases de données annotées au niveau du pixel ou avec des boîtes englobantes est très complexe, voire impossible dans certaines configurations. Pour cette raison, de nombreux efforts ont été faits par la communauté scientifique afin de développer des méthodes de segmentation faiblement supervisées nécessitant uniquement des annotations au niveau de l'image [146, 147, 148, 149, 150]. Ces approches sont souvent axées sur la segmentation de formes aux contours bien définis. L'efficacité de ces méthodes peut donc être considérablement réduite pour des formes aux contours irréguliers ce qui est le cas pour de nombreuses maladies affectant les pommes de terre.

Dans le contexte des réseaux de neurones profonds, il existe différents travaux reposant sur l'apprentissage faiblement supervisé pour la détection ou la segmentation d'objets. Parmi ces travaux, deux catégories principales peuvent être distinguées.

La première catégorie englobe les méthodes qui exploitent l'activation des neurones pour déterminer les motifs responsables de la décision de classification du réseau. Parmi les

travaux les plus significatifs qui ont adopté cette approche, on peut citer celui de Oquab et al. [151] qui ont proposé d'utiliser un réseau de neurones convolutifs dont la dernière couche, avant la couche de classification, est un *Global Max-Pooling* (GMP). À partir de ce réseau, les auteurs ont mis en évidence le potentiel de localisation d'objets d'un réseau entraîné uniquement pour de la classification d'images. Une méthode similaire à celle de Oquab et al. [151] a été introduite par Zhou et al. [149], sauf qu'au lieu d'utiliser une couche de regroupement maximale GMP, les auteurs ont opté pour une couche de *Global Average-Pooling* (GAP). Ils ont mis en avant le fait que la couche de GAP est plus adaptée pour mettre en évidence toute l'étendue de la région qui a contribué à la prise de décision, alors que la couche de GMP se focalise uniquement sur le pixel le plus important (le point maximum). Pinheiro et Collobert [146], quant à eux, ont proposé d'utiliser une couche de regroupement *Log-Sum-Exp* à la sortie de la dernière couche de convolution d'un CNN afin de mettre en évidence les pixels les plus décisifs pour la prise de décision lors de la procédure de classification. La sortie de la dernière couche de convolution du réseau proposé est composée de  $M$  cartes de caractéristiques, où  $M$  est le nombre de classes de la base de données (en incluant l'arrière-plan). Étant donné qu'au moment de l'apprentissage la base de données disponible contiennent uniquement des étiquettes au niveau d'image, la couche de regroupement *Log-Sum-Exp* est appliquée afin d'agréger les cartes de caractéristiques obtenues à la sortie de la dernière couche de convolution en un seul score de classification comme suit :

$$z^m = \frac{1}{r} \log \left[ \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W \exp(r X_{i,j}^m) \right] \quad (2.10)$$

où  $H$  et  $W$  sont respectivement la hauteur et la largeur des cartes de caractéristiques  $X$  obtenues à partir de la dernière couche de convolution,  $z^m$  la sortie obtenue pour la classe  $m$ , avec  $m = 1, \dots, M$ , et  $r$  l'hyper-paramètre à définir pour contrôler l'effet de la couche de regroupement : des valeurs de  $r$  élevées impliquent un effet similaire à une couche de GMP et des valeurs très faibles auront un effet similaire à une couche de GAP. Selon les résultats obtenus, les auteurs ont montré la supériorité de cette nouvelle couche de regroupement par rapport aux couches standard (GMP et GAP). Cependant, le choix de l'hyper-paramètre  $r$  rend la méthode plus complexe. Bien que des résultats intéressants aient été obtenus par ces méthodes, les localisations obtenues ne sont que des estimations assez peu précises des positions des formes ciblées. En effet, afin d'obtenir une segmentation fine des objets, une étape de post-traitement doit être appliquée.

La seconde catégorie de travaux reposant sur l'apprentissage faiblement supervisé concerne les méthodes qui se focalisent sur les données d'entrée du réseau. En effet, le concept clé derrière ces méthodes est fondé sur l'altération de l'image d'entrée, généralement par masquage, et l'observation de l'effet de cette altération sur le résultat de classification du réseau. De cette manière les régions de l'image qui ont le plus d'impact sur l'attribution de la classe sont isolées. Bazzani et al. [152] ont proposé un travail représentatif de cette dernière approche. Les auteurs ont utilisé une technique reposant sur l'évaluation des scores de prédiction à partir de masques générés artificiellement sur différentes parties de l'image. En effet, le masquage d'une région qui inclut les formes

pertinentes pour la classification de l'image produit une baisse significative du score de prédiction. Malgré l'efficacité avérée de cette méthode, elle nécessite plusieurs passes en avant (*forward*) avec différents masques avant de pouvoir définir l'emplacement des motifs recherchés.

Dans le contexte de l'analyse qualité des produits agricoles, peu de travaux exploitant l'apprentissage faiblement supervisé ont été menés. Cela peut s'expliquer notamment par le fait que les défauts à localiser ou à segmenter sont souvent dispersés sur toute la surface du produit. L'absence de contours bien définis des défauts et maladies affectant les produits agricoles peut constituer également une complexité supplémentaire à l'application de ces approches, notamment pour ce qui est de la segmentation. On peut toutefois citer les travaux de Lu et al. [153] qui ont abordé le diagnostic automatique des maladies du blé dans des conditions réelles en se basant sur l'apprentissage profond faiblement supervisé. La méthode proposée permet d'intégrer la classification de maladies du blé et la localisation des zones malades en utilisant une base d'apprentissage contenant des images capturées dans des conditions non contrôlées et des annotations uniquement au niveau de l'image. Pour obtenir la localisation des maladies, les auteurs ont utilisé la sortie de la dernière couche de convolution d'un réseau de neurones profond. Cette sortie est composée de  $M$  cartes de caractéristiques, où  $M$  est le nombre de catégories dans la base de données. Ces cartes passent par une opération de redimensionnement pour obtenir  $M$  cartes thermiques (*heatmaps*) dont chacune possède la même taille que l'image d'entrée. Les zones affectées sont estimées à partir de ces cartes. Plus les valeurs d'une région sur ces cartes sont élevées, plus est le risque qu'une maladie affecte cette dite région. Brahim et al. [24] ont proposé, quant à eux, d'utiliser la méthode d'occlusion proposée par Zeiler et al. [154] pour localiser et analyser les défauts classés par un CNN. Malgré l'obtention de bons résultats de localisation, la méthode proposée est caractérisée par une forte complexité calculatoire compte tenu de la nature itérative de la méthode (plusieurs passes en avant), ce qui limite son intégration à des systèmes industriels quand le temps de traitement est un critère important.

### Apprentissage non supervisé

En raison de la complexité qu'engendre la conception et l'étiquetage de grandes bases de données, différentes approches d'apprentissage non supervisé de représentations ont été explorées. L'avantage de ces méthodes réside dans le fait qu'elles ne nécessitent pas d'étiquettes durant la phase d'apprentissage, mais uniquement des observations.

L'auto-encodeur (AE) est l'une des méthodes d'apprentissage de représentations non supervisé parmi les plus populaires. L'AE est un réseau de neurones multicouches qui a pour objectif principal d'apprendre des représentations descriptives et discriminatives des données d'entrée [155]. Au même titre que l'ACP (*Principal Component Analysis*) [156], l'AE peut également être utilisé pour définir un espace de variables latentes de dimension réduit, quand la représentation produite est de dimension inférieure aux données d'entrée. L'AE est constitué de deux parties : la première, appelée encodeur, permet la transformation de la donnée en une représentation comprimée (le code). La seconde partie, nommée décodeur, rend possible le décodage du code pour reconstruire la donnée d'entrée. Dans la



figure 2.9 nous pouvons voir un exemple d'auto-encodeur entièrement connecté avec une seule couche cachée, où le code est représenté par  $Z$  (la sortie de l'encodeur). Dans le cas d'utilisation d'images comme données d'entrée, chaque neurone est connecté à un pixel de l'image et en sortie on obtient une image de même taille que l'image d'entrée. L'AE est entraîné de manière non supervisée pour reconstruire l'entrée en minimisant l'erreur de reconstruction entre l'entrée et la sortie. Après l'apprentissage, le réseau devient capable de générer une représentation comprimée (appelée «code») pour chaque donnée d'entrée qui peut ensuite être utilisée comme entrée d'un classifieur.

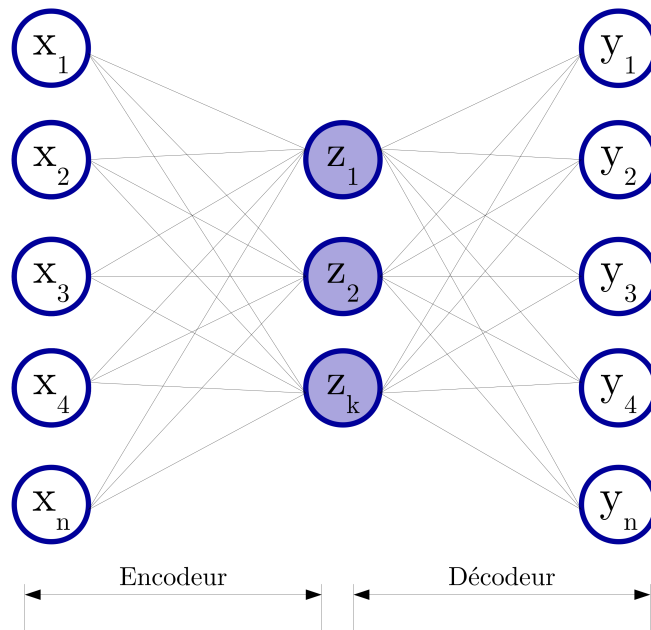


FIGURE 2.9 – Exemple de l'auto-encodeur entièrement connecté avec une seule couche cachée de  $k$  neurones.

Différentes variantes de l'auto-encodeur ont été développées dans le but d'améliorer les représentations extraites et de les adapter à une gamme d'applications de plus en plus large. Une contrainte de parcimonie a été introduite afin de proposer l'AE parcimonieux. Cet auto-encodeur permet d'éviter le sur-apprentissage (*overfitting*) et d'extraire des caractéristiques pertinentes et moins distribuées [157]. Le principe de ces AEs réside dans l'intégration d'une pénalité de parcimonie sur la couche de sortie de l'encodeur afin d'obtenir des unités cachées dont l'activité moyenne est proche de 0. Le bon fonctionnement de ce réseau nécessite généralement un plus grand nombre de neurones cachés que d'entrées. Une autre variante de l'AE est l'AE débruiteur [158]. Durant l'apprentissage de ce réseau, un bruit (bruit gaussien, masquant ou sel-et-poivre) est ajouté à l'image d'entrée et le réseau est entraîné à reconstruire l'image sans bruit. Pour pouvoir générer l'image sans bruit l'AE sera contraint d'apprendre uniquement les caractéristiques pertinentes des images. Cette approche vise à obtenir des représentations plus robustes des images d'entrée. Pour apprendre des motifs complexes, il est possible d'exploiter de manière séquentielle plusieurs auto-encodeurs. En effet, il est possible d'entraîner un premier AE et d'ensuite

utiliser la sortie de l'encodeur de ce réseau pour générer des données d'apprentissage d'un second AE. Ce processus peut être réitéré avec autant d'auto-encodeurs que nécessaire. Une fois tous les AEs entraînés, les encodeurs de ces réseaux sont extraits et empilés pour être utilisés comme un extracteur de caractéristiques de haut niveau. Cette architecture est désignée dans la littérature par *Stacked Autoencodeur* - SAE [159].

Bien que les AEs ont démontré leur efficacité dans de nombreuses applications, l'utilisation de réseaux entièrement connectés n'est pas le plus adéquat pour des données 2D, telles que des images, où les relations spatiales sont d'une importance primordiale. Les auto-encodeurs convolutifs (*Convolutional AutoEncodeur* - CAE) sont plus adaptés à ce type de données. En effet, dans ces réseaux, les couches entièrement connectées sont remplacées par des couches préservant les relations spatiales, telles que les blocs de convolution et les couches de *pooling*.

Malgré la diversité des méthodes d'apprentissage de représentations non supervisées, et l'avantage considérable que représente leur indépendance vis-à-vis des grandes bases de données étiquetées, la grande majorité des méthodes de contrôle qualité des produits agroalimentaires proposées dans la littérature sont supervisées. Cette tendance peut s'expliquer, d'une part, par l'objectif de classification de la majorité des applications et, d'autre part, par les performances des méthodes supervisées dans ce cas. Néanmoins, quelques travaux combinent l'apprentissage de caractéristiques non supervisé et la classification supervisée. Hung et al. [160] ont proposé une méthode reposant sur l'utilisation d'un AE parcimonieux pour la segmentation automatique d'amandes dans une plantation. Des patches d'images RVB et IR (infrarouge) ont été utilisés pour entraîner un AE parcimonieux. Ensuite, la sortie renvoyée par l'encodeur, un vecteur de représentation de plus faible dimension, a été utilisée comme entrée d'un perceptron à couche unique intégrant une fonction de sortie *softmax* pour classer chaque pixel selon 5 catégories : feuilles, terre, amandes, ciel et tronc. L'apprentissage du perceptron a été réalisé avec 80 images étiquetées manuellement au niveau du pixel. Cette quantité d'images aurait été insuffisante pour l'apprentissage d'un réseau de neurones convolutifs profond, mais elle est suffisante pour combiner un AE et un perceptron à couche unique. Grâce à cette combinaison ingénieuse, les auteurs ont pu démontrer la supériorité de leur méthode face aux méthodes conventionnelles d'extraction de caractéristiques. Les auteurs Yang et al. [161] ont proposé d'entraîner un SAE, formé par trois auto-encodeurs entièrement connectés, dans le but d'apprendre des représentations robustes à partir d'images de plantes en niveaux de gris. Un classifieur *softmax* a été ensuite entraîné de manière supervisée en utilisant les représentations fournies par l'encodeur du SAE pour classer des images de plantes en trois catégories. Les auteurs ont démontré qu'en empilant 3 AE les résultats étaient plus précis que dans le cas où un seul AE était utilisé (93,3% et 85,4 % respectivement). L'utilisation d'un auto-encodeur combiné avec un classifieur SVM pour la détection de maladies de plantes a été proposée par Pardede et al. [162]. Contrairement à l'utilisation habituelle de la sortie de l'encodeur pour entraîner un classifieur, les auteurs ont opté pour la sortie du décodeur comme entrée du classifieur. Les auteurs ont comparé les résultats en utilisant un AE entièrement connecté et un auto-encodeur convolutif (CAE), en démontrant que le second AE surpassait le premier. Un SAE parcimonieux combiné avec un classifieur *softmax* a été utilisé par Abbaszadeh et al. [163] pour classer les pistaches en deux classes :

saines et abîmées. Une base de données avec 305 images de pistaches a été créée, contenant 214 images de pistaches endommagées et 91 saines. Le SAE a été entraîné en utilisant deux AE parcimonieux entièrement connectés. La sortie de la dernière couche cachée du deuxième encodeur de dimension 500 (plus de 145 fois plus faible que la dimension des données d'entrée) a été utilisée pour entraîner de manière supervisée un perceptron à une seule couche avec une fonction de sortie *softmax*.

Une synthèse des méthodes reposant sur l'apprentissage de caractéristiques les plus pertinentes pour notre étude est présentée dans le tableau 2.4. Les approches sont divisées par rapport à trois critères : la nature de la tâche souhaitée (classification, localisation ou segmentation), le mode d'apprentissage utilisé (supervisé, faiblement supervisé ou non supervisé) et, finalement, l'initialisation du réseau de neurones (apprentissage complet ou *finetuning*).

## 2.4 Conclusion

Dans ce chapitre nous avons introduit les méthodes de la littérature employés pour la classification, la localisation et/ou la segmentation appliquées aux images de produits agricoles et alimentaires. Tout d'abord, les étapes d'acquisition d'image et de prétraitement ont été discutées, tout en justifiant les décisions prises pour la construction de notre propre système. Ensuite, les deux catégories de méthodes les plus couramment utilisées pour l'analyse d'images ont été présentées. Les méthodes classiques reposant sur une extraction ciblée de caractéristiques ont été introduites en premier. Nous avons également établi une revue détaillée des méthodes de la littérature les plus significatives dans cette première catégorie. Nous nous sommes ensuite intéressés aux méthodes les plus récentes, reposant sur l'apprentissage de représentations. Nous avons évoqué un grand nombre de travaux axés sur cette approche tout en nous focalisant sur les plus prometteurs. Tout au long de ce chapitre nous nous sommes efforcés de mettre en évidence les avantages et les contraintes de chaque approche analysée dans le but de mettre en avant les pistes de recherche les plus susceptibles de permettre une intégration concrète.

Dans le chapitre suivant, nous présentons notre première méthode de classification et localisation de défauts sur les pommes de terre. Compte tenu des contraintes relatives à notre problématique, principalement en ce qui concerne la complexité et la variabilité des motifs à détecter, nous avons opté pour une méthode fondée sur les réseaux de neurones convolutifs profonds, notamment en raison de la supériorité avérée de ce type de réseaux dans de nombreuses tâches de reconnaissance de forme, par exemple dans la classification d'images. Cependant, la nécessité d'utilisation de grandes bases de données étiquetées afin d'entraîner des réseaux profonds nous contraint également à prendre en considération des techniques d'apprentissage de représentations peu profonds, comme l'auto-encodeur, ainsi que des méthodes de classification plus conventionnelles, telles que les machines à vecteur de support. Dans le but de minimiser l'effort d'étiquetage manuel d'images, des procédures d'apprentissage alternatives, notamment l'apprentissage faiblement supervisé, seront également étudiées dans le chapitre 4.

TABLEAU 2.4 – Revue sur les méthodes reposant sur l’apprentissage de caractéristiques («Appren. de caract.») les plus pertinents. S = supervisé, F = faiblement supervisé, NS = non supervisé. Lorsque plus d’une méthode est évaluée dans un travail, le symbole \* indique la méthode qui donne les meilleures performances.

	Aut.	Prod.	Base de données		Appren. de caract.			Réseau de neurones		
			Privé	Publique	S	FS	NS	Scratch	Finetuning	
Classification	[163]	Pistaches (2 classes)	305					x		
	[92]	Feuilles (2 classes)		54306	x			x	x*	
	[161]	Feuilles (3 classes)	630				x	x		
	[162]	Feuilles (7 classes)		6004			x	x		
Classification+Localisation	[26]	Feuilles (4 classes)	8178		x				x	
	[64]	Feuilles (9 classes)	5000		x				x	
	[153]	Blé (7 classes)	9230			x			x	
	[164]	Pommes	1115				x			x
		Olives	1402				x			x
		Amandes (comp- tage)	1156	555			x			x
	[24]	Feuilles (9 classes)		14828		x		x	x*	
[60]	Pommes (1 classe)	700		x			x			
Segmentation	[25]	Feuilles (4 classes)		1184	x			x	x*	
	[160]	Amandes (5 classes)	80	477			x	x		
	[165]	Plantes (9 classes)		161	x			x		



# Chapitre 3

## Apprentissage supervisé pour la classification et la localisation de défauts externes

### Sommaire

---

<b>3.1</b>	<b>Introduction</b>	<b>49</b>
<b>3.2</b>	<b>Description théorique</b>	<b>50</b>
3.2.1	Réseaux de neurones convolutifs	51
3.2.2	Algorithmes d'optimisation	52
3.2.3	Auto-encodeur	59
3.2.4	Machine à vecteurs de support	62
<b>3.3</b>	<b>Création de la base de données</b>	<b>67</b>
<b>3.4</b>	<b>Système proposé reposant sur l'apprentissage supervisé</b>	<b>69</b>
3.4.1	Classification des défauts par face de pomme de terre	71
3.4.2	Localisation de défauts	75
3.4.3	Classification de défauts par gravité	75
<b>3.5</b>	<b>Résultats expérimentaux</b>	<b>76</b>
3.5.1	Critères d'évaluation	77
3.5.2	Résultats de la classification	78
3.5.3	Résultats de la localisation de défauts : endommagés et verts	82
3.5.4	Résultats de la classification de défauts par gravité	85
3.5.5	Résultats de la classification multi-classes multi-étiquettes	88
<b>3.6</b>	<b>Conclusion</b>	<b>90</b>

---

### 3.1 Introduction

Ces dernières années, des méthodes fondées sur l'apprentissage profond, en particulier sur les réseaux de neurones convolutifs, ont été largement utilisées pour la classification d'images de toute nature. Motivés par les excellents résultats obtenus par ces méthodes,

nous proposons d’explorer l’application de l’apprentissage profond pour la classification des pommes de terre en fonction de leur état (endommagement, verdissement et maladies).

Dans ce chapitre, nous présenterons la première méthode proposée afin d’obtenir un système fiable de classification d’images de pommes de terre en 6 catégories : saine, endommagée, verte, dartrose, gale commune et rhizoctone. Les défauts verdissement et endommagement seront également classés selon 2 niveaux de gravité, léger ou grave. Pour atteindre cet objectif, un réseau de neurones convolutifs est entraîné afin de classer chaque image selon l’une des 6 classes. Subséquemment, pour les classes verte et endommagée, un auto-encodeur, en conjonction avec un classifieur SVM, est utilisé pour localiser les défauts. Enfin, l’information de localisation de ces défauts est utilisée comme entrée d’un deuxième classifieur SVM dans le but d’obtenir la classification finale par gravité.

L’organisation du chapitre est la suivante. Tout d’abord, nous présenterons les bases théoriques des techniques utilisées dans notre méthode : le CNN, l’auto-encodeur et les classifieurs SVMs (deux classes et mono-classe). Dans un second temps, nous présenterons la base de données créée et utilisée au cours de cette thèse. Ensuite, nous expliquerons en détail chaque étape de la méthode proposée. Avant de conclure ce chapitre, nous présenterons et commenterons les résultats obtenus.

## 3.2 Description théorique

Dans cette section, nous présenterons brièvement les principaux concepts théoriques qui fondent le développement de notre première méthode de classification et de localisation de défauts et maladies de la pomme de terre.

Comme cela a été mis en avant dans le chapitre 2, les CNNs ont montré leur capacité à traiter de nombreuses tâches de reconnaissance de formes. Fort de ce constat, nous avons opté pour l’utilisation d’un CNN dans notre première méthode. En effet, un CNN est utilisé pour classer les faces de pommes de terre en fonction de différents défauts et maladies, ou de leur absence (pomme de terre saine). Dans la section 3.2.1, nous présenterons les concepts clés à la base des réseaux de neurones convolutifs ainsi que les algorithmes d’optimisation les plus couramment utilisés afin d’entraîner ces réseaux (section 3.2.2).

Après l’étape de classification, une étape de localisation de défauts (endommagements et verdissements) est réalisée. Malgré les capacités de classification des CNNs, leur nature supervisée ainsi que la nécessité d’utilisation de grandes bases de données étiquetées pour leur apprentissage nous ont contraint à explorer des techniques alternatives pour cette étape. Nous proposons donc de réaliser la localisation d’endommagements et verdissements à partir de la classification de patches. Dans le but de rendre cette classification plus robuste, nous projetons d’abord les patches extraits des images classées au préalable par le CNN comme endommagées ou vertes dans un espace de variables latentes de dimension réduites. Cette projection est réalisée à l’aide d’un auto-encodeur, qui sera présenté plus en détail dans la section 3.2.3. Ensuite, les représentations obtenues sont utilisées afin de classer chaque patch à l’aide d’un SVM. Il faut remarquer que la réduction de dimensionnalité obtenue grâce à l’AE permet de rendre la classification des patches plus robuste. Comme il a été démontré par Bellman [166], les observations dans un espace à grande

dimension sont souvent très éloignées les unes des autres, il est donc difficile de constituer des clusters compacts, ce qui rend la classification des données en grandes dimensions plus difficile.

Dans la dernière étape de notre méthode, les informations relatives à la localisation de défauts sont utilisées afin de classer les pommes de terre endommagées et vertes par niveau de gravité, léger ou grave. Un classifieur SVM, présenté en détail dans la section 3.2.4, est utilisé pour accomplir cette classification. Comme nous avons vu dans le chapitre 2, le SVM est un algorithme d'apprentissage dont les capacités en terme de classification ont largement été établies [106, 96, 138, 119, 22]. Ce classifieur est notamment apprécié pour sa capacité d'apprentissage sur des ensembles de taille réduite ainsi que pour son efficacité calculatoire. En effet, seul un sous ensemble restreint de points d'entraînement (vecteurs de support) est nécessaire pour le calcul de la solution ce qui permet une maîtrise de la complexité.

### 3.2.1 Réseaux de neurones convolutifs

Un réseau de neurones convolutifs (CNN) est un réseau neuronal profond doté d'une architecture à plusieurs couches disposées généralement en cascade. Grâce à cette disposition de couches, le CNN est capable d'extraire des représentations en intégrant différents niveaux d'abstraction et en visant à reproduire le modèle de perception humaine [167]. Les quatre aspects clés de ces réseaux sont les connexions locales, le partage de paramètres, le regroupement et l'utilisation de plusieurs couches [61]. Dans la figure 3.1 nous pouvons observer le célèbre réseau «LeNet-5» proposé par Lecun et al. [2]. Ce réseau a eu un grand impact sur le domaine de la vision par ordinateur et l'apprentissage automatique. En effet, son architecture a servi de base à de nombreux autres réseaux qui se sont démarqués dans de multiples tâches de reconnaissance de formes. Il est principalement composé de couches de convolution, de couches de regroupement (*subsampling* ou *pooling*) et de couches entièrement connectées. Les couches de convolution sont utilisées pour extraire des caractéristiques représentatives des données d'entrée en employant des filtres (voir équation 2.8 introduit dans le chapitre 2). Les couches de convolution sont généralement combinées avec des fonctions d'activation telles que la fonction ReLU, afin d'introduire de la non-linéarité dans le réseau. En définitive, une couche de convolution suivie par une fonction d'activation forment un bloc désigné par bloc de convolution. Ce bloc produit un ensemble de représentations, connu sous le nom de cartes de caractéristiques (*feature maps*). Ces cartes sont ensuite introduites dans une couche de *pooling*. L'objectif de cette couche est de réduire la résolution des cartes de caractéristiques d'entrée et elle permet également d'acquérir de l'invariance aux translations et aux modifications minimales dans les motifs d'entrée. En empilant plusieurs blocs de convolution et de *pooling* nous obtenons un bloc d'extraction de caractéristiques capable d'extraire des représentations de haut niveau, robustes et représentatives des données d'entrée. Une fois les représentations obtenues, elles sont passées comme entrée à un second bloc dit de classification. Ce dernier est composé d'une ou plusieurs couches entièrement connectées et d'une fonction de décision finale. Dans une couche entièrement connectée, chaque neurone est relié à tous les



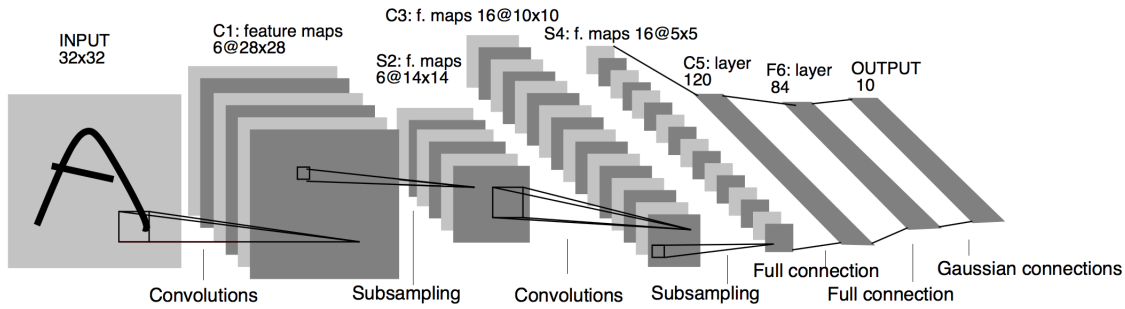


FIGURE 3.1 – Réseau de neurones convolutifs «LeNet-5» entraîné pour classer des images en 10 catégories [2].

neurones de la couche précédente, elle est notamment utilisée afin d’obtenir une caractérisation globale de l’image d’entrée. Dans le contexte des réseaux de neurones, la fonction de décision *softmax* (équation 3.1) est couramment utilisée dans la dernière couche afin de permettre la classification des données.

$$f(x_m) = \frac{e^{x_m}}{\sum_{j=1}^M e^{x_j}}, \quad m = 1, \dots, M \quad (3.1)$$

L’image de cette fonction est interprétée comme une estimation des probabilités d’un événement  $m$  parmi  $M$ . Durant l’entraînement du réseau, la sortie de la fonction *softmax* est utilisée comme entrée de la fonction de perte à minimiser. L’entropie croisée, donnée par l’équation 3.2, figure parmi les fonctions de perte les plus couramment utilisées.

$$\mathcal{L}_{CE} = -\frac{1}{n} \sum_{i=1}^n \sum_{m=1}^M \mathbb{1}_{(y_i=m)} \log(f(x_{im})) \quad (3.2)$$

où  $y_i \in \{1, 2, \dots, M\}$  est la classe réelle de l’observation  $i$  et  $f(x_{im})$  est le score de sortie du réseau correspondant à l’observation  $i$  pour la classe  $m$ .

Nous verrons ci-dessous les différents algorithmes d’optimisation généralement employés pour atteindre la minimisation de cette fonction de perte.

### 3.2.2 Algorithmes d’optimisation

Plusieurs algorithmes d’optimisation itératifs ont été développés afin d’entraîner les réseaux de neurones. Ces algorithmes peuvent être classés en deux catégories [168] :

1. Algorithmes d’optimisation de premier ordre : ces algorithmes utilisent le gradient de la fonction de perte par rapport aux paramètres du réseau. Le gradient nous indique la direction d’accroissement, nous faisant évoluer en sens inverse, pour diminuer ainsi la valeur de la fonction de perte. Parmi les méthodes du premier ordre, la descente de gradient est sans doute la plus couramment utilisée.

2. Algorithmes d'optimisation de second ordre : dans cette catégorie, la dérivée seconde est utilisée (matrice hessienne) afin d'accélérer la minimisation de la fonction de perte. Le calcul de cette dérivée est plus complexe que dans le cas de la dérivée première, ce qui induit une augmentation de temps de calcul et de la mémoire utilisée. Toutefois, cela peut être compensé dans certains cas par une réduction du nombre d'itérations.

Il faut souligner que les problèmes d'optimisation relatifs aux réseaux de neurones profonds sont de nature non convexe, ce qui engendre la présence de nombreux minima locaux et des points où le gradient est nul [169]. De ce fait, le choix de l'algorithme d'optimisation, ainsi que des hyper-paramètres associés, sont fondamentaux pour trouver une bonne solution au problème [61].

Nous nous focaliserons par la suite sur les algorithmes d'optimisation du premier ordre en raison de leur popularité dans l'entraînement des réseaux de neurones profonds.

### La descente de gradient

La méthode d'optimisation le plus couramment utilisé pour l'entraînement des réseaux de neurones profonds est la descente de gradient, mise en œuvre par l'algorithme 1.

---

#### Algorithme 1 : La descente de gradient

---

Initialisation aléatoire des paramètres du réseau :  $\theta_0$   
 Taux d'apprentissage :  $\alpha$   
 $t = 0$   
**tant que** condition d'arrêt non atteinte **faire**  
   **pour** chaque donnée d'entraînement étiquetée  $(x_i, y_i)$  avec  $i = 1, \dots, n$  **faire**  
      $y'_i =$  Propagation en avant de  $x_i$   
      $\mathcal{J}_i(\theta_t) =$  Calcul de l'erreur entre  $y'_i$  et  $y_i$   
      $\nabla_{\theta_t} \mathcal{J}_i(\theta_t) =$  Rétro-propagation du gradient de la fonction de perte  
   **fin pour**  
    $\nabla_{\theta_t} \mathcal{J}(\theta_t) = \frac{1}{n} \sum_{i=1}^n \nabla_{\theta_t} \mathcal{J}_i(\theta_t)$   
    $\theta_{t+1} = \theta_t - \alpha \nabla_{\theta_t} \mathcal{J}(\theta_t)$   
    $t = t + 1$   
**fin tant que**

---

Cet algorithme d'optimisation itératif met à jour les paramètres du réseau dans la direction opposée au gradient afin de minimiser la fonction de perte :

$$\theta_{t+1} = \theta_t - \alpha \nabla_{\theta_t} \mathcal{J}(\theta_t) \quad (3.3)$$

où  $\theta_t$  représente l'ensemble des paramètres du réseau à l'instant  $t$ ,  $\nabla_{\theta_t} \mathcal{J}(\theta_t)$  le gradient de la fonction de perte  $\mathcal{J}$  par rapport aux paramètres  $\theta_t$  et  $\alpha$  le pas ou taux d'apprentissage

(*learning rate*) qui permet de définir la vitesse de convergence. Le choix de  $\alpha$  est primordial pour l'entraînement du réseau : un pas trop petit conduit à une convergence trop lente et dans le cas contraire, un pas trop élevé provoquera de forts changements dans les paramètres, ce qui peut entraîner des oscillations et donc empêcher la convergence. Dans le but d'améliorer la convergence, il est possible de définir un taux d'apprentissage différent pour chaque couche du réseau et de faire varier le taux d'apprentissage au fur à mesure que l'apprentissage progresse.

Le calcul du gradient de la fonction de perte par rapport aux paramètres du réseau se fait grâce à la technique de rétro-propagation [170]. Étant donné qu'un réseau de neurones est une composition de fonctions, la règle de dérivation en chaîne est utilisée afin de calculer ce gradient. Soient  $f$  et  $g$  deux fonctions dérivables, tel que  $g$  est dérivable au point  $x$  et  $f$  est dérivable au point  $g(x)$ , alors la composée  $z = f \circ g$  est dérivable au point  $x$  et :

$$z'(x) = (f \circ g)'(x) = f'(g(x)) \times g'(x) \quad (3.4)$$

Cette formule peut également être écrite en utilisant la notation de Leibniz de la manière suivante :

$$\frac{\partial z(x)}{\partial x} = \frac{\partial z(x)}{\partial g(x)} \times \frac{\partial g(x)}{\partial x} \quad (3.5)$$

Afin d'éclaircir comment la technique de rétro-propagation du gradient est appliquée dans l'entraînement d'un réseau de neurones, nous allons présenter un exemple. L'architecture du réseau présentée à ce propos est illustrée dans la figure 3.2. Le réseau est formé par trois

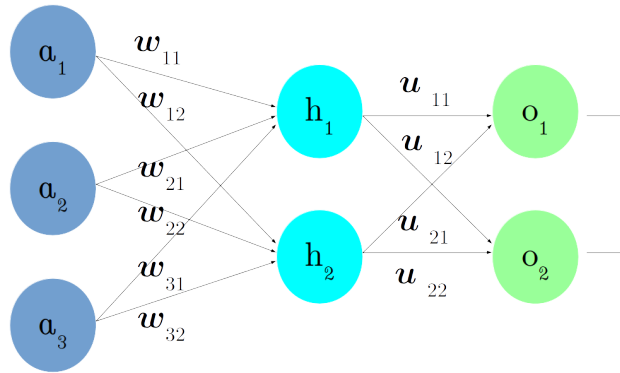


FIGURE 3.2 – Perceptron multicouche avec une seule couche cachée.

couches : une couche d'entrée avec trois neurones, une couche cachée avec deux neurones et finalement, une couche de sortie constituée également de deux neurones. La fonction d'activation sigmoïde est utilisée comme fonction d'activation dans la couche cachée et la couche de sortie :

$$f(x) = \frac{1}{1 + e^{-x}} \quad (3.6)$$

La dérivé de cette fonction sigmoïde est donnée par :

$$\frac{\partial f(x)}{\partial x} = f(x)(1 - f(x)) \quad (3.7)$$

Dans l'objectif d'apprendre les paramètres du réseau, les étapes suivantes sont effectuées :

1. Les paramètres du réseau  $w_{ij}$  et  $u_{jk}$  sont initialisés de manière aléatoire.
2. La propagation vers l'avant est effectuée. En effet, les entrées  $a_i$  se propagent à travers les neurones de chaque couche afin d'obtenir la sortie  $o_k$ .  
En passant à travers la première couche, on obtient les activations suivantes :

$$z_{h_j} = \sum_i w_{ij} a_i \quad (3.8)$$

$$h_j = f(z_{h_j}) \quad (3.9)$$

Puis, la sortie de chaque neurone du réseau est calculée comme suit :

$$z_{o_k} = \sum_j u_{jk} h_j \quad (3.10)$$

$$o_k = f(z_{o_k}) \quad (3.11)$$

3. Nous calculons l'erreur de prédiction en utilisant l'erreur quadratique moyenne comme fonction de perte. Son expression pour une observation est :

$$E = \sum_k \frac{1}{2} (y_k - o_k)^2 \quad (3.12)$$

4. Pour une observation, nous calculons ensuite l'erreur pour chaque neurone  $k$  de la couche de sortie ( $\delta_k$ ) :

$$\delta_k = \frac{\partial E}{\partial o_k} \frac{\partial o_k}{\partial z_{o_k}} = -(y_k - o_k) o_k (1 - o_k) \quad (3.13)$$

Puis, l'erreur est rétro-propagée à la couche intermédiaire :

$$\delta_j = \frac{\partial E}{\partial h_j} \frac{\partial h_j}{\partial z_{h_j}} = \left( \sum_k u_{jk} \delta_k \right) h_j (1 - h_j) \quad (3.14)$$

5. L'erreur totale et le gradient totale s'estiment en faisant la somme des termes  $E$ ,  $\delta_k$  et  $\delta_j$  sur l'ensemble des observations d'apprentissage. Finalement, les erreurs ainsi obtenues sont utilisées afin de mettre à jour les paramètres (ou poids) du réseau :

$$u_{jk} = u_{jk} - \alpha \delta_k h_j = u_{jk} - \alpha \frac{\partial E}{\partial u_{jk}} \quad (3.15)$$

$$w_{ij} = w_{ij} - \alpha \delta_j a_i = w_{ij} - \alpha \frac{\partial E}{\partial w_{ij}} \quad (3.16)$$

Malgré l'efficacité de la descente de gradient, l'utilisation de cette méthode présente quelques inconvénients. Tout d'abord, l'estimation du gradient en prenant en compte toutes les données d'entraînement est très coûteuse. De plus, une limitation liée à la configuration matérielle existe. En effet, plus le nombre d'observations utilisées pour estimer le gradient est important, plus la taille de la mémoire nécessaire pour traiter tous les échantillons en parallèle augmente.

En raison de ces inconvénients, la descente de gradient a été initialement remplacé par une variante qui utilise un seul échantillon à la fois pour estimer le gradient (nommée descente de gradient stochastique SGD). Dans cette dernière, pour chaque itération le gradient de la fonction de coût est estimé à l'aide d'un seul exemple d'entraînement choisi aléatoirement. Cette approche est préférée à la descente de gradient standard car il a été démontré que la précision gagnée en utilisant la totalité des observations pour estimer le gradient n'est pas suffisante pour compenser la complexité calculatoire engendrée.

### La descente de gradient stochastique

La descente de gradient stochastique (SGD) réalise la mise à jour des paramètres en utilisant uniquement un seul exemple choisi aléatoirement dans l'ensemble d'entraînement (voir algorithme 2).

Cette stratégie a comme principaux avantages la rapidité de convergence et la diminution de la complexité calculatoire. Cependant, le fait d'estimer le gradient avec une seule observation génère des fluctuations importantes. Pour cette raison, le chemin emprunté par l'algorithme pour atteindre les minima est habituellement plus bruité que pour la descente de gradient standard.

Une approche intermédiaire désignée par descente de gradient par mini-lots (ou SGD par mini-lots) permet un compromis entre les deux précédentes en proposant un apprentissage reposant sur des lots.

---

**Algorithme 2** : La descente de gradient stochastique

---

Initialisation aléatoire des paramètres du réseau :  $\theta_0$

Taux d'apprentissage :  $\alpha$

$t = 0$

**tant que** condition d'arrêt non atteinte **faire**

**pour** chaque donnée d'entraînement étiquetée  $(x_i, y_i)$  avec  $i = 1, \dots, n$  **faire**

$y'_i =$  Propagation en avant de  $x_i$

$\mathcal{J}_i(\theta_t) =$  Calcul de l'erreur entre  $y'_i$  et  $y_i$

$\nabla_{\theta_t} \mathcal{J}_i(\theta_t) =$  Rétro-propagation du gradient de la fonction de perte

$\theta_{t+1} = \theta_t - \alpha \nabla_{\theta_t} \mathcal{J}_i(\theta_t)$

$t = t + 1$

**fin pour**

**fin tant que**

---

### La descente de gradient par mini-lots

Dans cette stratégie, la mise à jour des paramètres est faite à l'aide de l'estimation du gradient obtenue avec  $n_b$  observations, (voir algorithme 3). Cela se traduit par une convergence plus stable et plus rapide.

---

#### Algorithme 3 : La descente de gradient stochastique par mini-lots

---

Initialisation aléatoire des paramètres du réseau :  $\theta_0$

Taux d'apprentissage :  $\alpha$

$n_b$  nombre d'observations dans un mini-lot

$t = 0$

**tant que** condition d'arrêt non atteinte **faire**

Extraire  $n_b$  exemples de l'ensemble d'entraînement étiqueté  $(x_i, y_i)$  tirés au hasard

**pour** chaque couple  $(x_i, y_i)$  avec  $i = 1, \dots, n_b$  **faire**

$y'_i =$  Propagation en avant de  $x_i$

$\mathcal{J}_i(\theta_t) =$  Calcul de l'erreur entre  $y'_i$  et  $y_i$

$\nabla_{\theta_t} \mathcal{J}_i(\theta_t) =$  Rétro-propagation du gradient de la fonction de perte

**fin pour**

$\nabla_{\theta_t} \mathcal{J}(\theta_t) = \frac{1}{n_b} \sum_{i=1}^{n_b} \nabla_{\theta_t} \mathcal{J}_i(\theta_t)$

$\theta_{t+1} = \theta_t - \alpha \nabla_{\theta_t} \mathcal{J}(\theta_t)$

$t = t + 1$

**fin tant que**

---

Bien que les techniques d'optimisation basées sur la descente de gradient décrit ci-dessus soient les plus employées pour l'entraînement des réseaux de neurones, certaines difficultés doivent être considérées :

- Le choix d'un taux d'apprentissage  $\alpha$  adéquat.
- Le choix de la technique de mise à jour du taux d'apprentissage, par exemple réduction à partir d'un certain nombre d'itérations.

De nombreuses améliorations ont été proposées pour l'algorithme de la descente de gradient [136, 171, 172, 173, 174], parmi lesquelles on peut citer la méthode d'accélération (ou momentum) [136] et Adam [174].

### Accélération

La méthode d'accélération (ou momentum) a été proposée par Rumelhart et al. [136] afin d'accélérer l'apprentissage (voir algorithme 4). Dans cette méthode, une nouvelle variable  $\nu$  est introduite, communément appelée vitesse, qui représente la direction et la vitesse que prennent les paramètres pour se déplacer dans l'espace de paramètres. Cette variable

$\nu$  est définie en combinant le gradient estimé des étapes passées et le gradient de l'étape actuelle. La règle de mise à jour des paramètres est définie comme suit :

$$\begin{aligned}\nu_{t+1} &= \gamma\nu_t + \alpha\nabla_{\theta_t}\mathcal{J}(\theta_t) \\ \theta_{t+1} &= \theta_t - \nu_{t+1}\end{aligned}\tag{3.17}$$

où  $\gamma \in [0, 1)$  est connu comme l'hyper-paramètre de friction (ou de momentum) communément fixé à 0,9 [168]. En effet, plus  $\gamma$  est grand par rapport au taux d'apprentissage  $\alpha$ , plus les gradients précédents influencent la nouvelle direction prise par la méthode. En regardant la nouvelle règle de mise à jour (équations 3.17), nous pouvons observer que la taille du pas ne dépend plus uniquement du taux d'apprentissage et de l'estimation du gradient, mais également de la direction et de la taille d'une séquence (ou suite) de gradients. En effet, un pas plus grand sera effectué lorsque la direction de plusieurs gradients successifs sera la même.

---

**Algorithme 4** : La descente de gradient stochastique par mini-lots avec momentum

---

Initialisation aléatoire des paramètres du réseau :  $\theta_0$   
 Initialisation de la vitesse :  $\nu_0$   
 Taux d'apprentissage :  $\alpha$   
 Hyper-paramètre de momentum :  $\gamma$   
 $n_b$  nombre d'observations dans un mini-lot  
 $t = 0$   
**tant que** condition d'arrêt non atteinte **faire**  
     Extraire  $n_b$  exemples de l'ensemble d'entraînement étiqueté  $(x_i, y_i)$  tirés au hasard  
     **pour** chaque couple  $(x_i, y_i)$  avec  $i = 1, \dots, n_b$  **faire**  
          $y'_i =$  Propagation en avant de  $x_i$   
          $\mathcal{J}_i(\theta_t) =$  Calcul de l'erreur entre  $y'_i$  et  $y_i$   
          $\nabla_{\theta_t}\mathcal{J}_i(\theta_t) =$  Rétro-propagation du gradient de la fonction de perte  
     **fin pour**  
      $\nabla_{\theta_t}\mathcal{J}(\theta_t) = \frac{1}{n_b} \sum_{i=1}^{n_b} \nabla_{\theta_t}\mathcal{J}_i(\theta_t)$   
      $\nu_{t+1} = \gamma\nu_t + \alpha\nabla_{\theta_t}\mathcal{J}(\theta_t)$  (mis à jour de la vitesse)  
      $\theta_{t+1} = \theta_t - \nu_{t+1}$   
      $t = t + 1$   
**fin tant que**

---

## Adam

Proposé par Kingma et al. [174], Adam est un algorithme d'optimisation à taux d'apprentissage adaptatif très utilisé pour l'entraînement des réseaux de neurones profonds (voir algorithme 5). Le taux d'apprentissage adaptatif fait référence au fait que le taux d'apprentissage est différent pour chaque paramètre du réseau. La première étape de l'algorithme consiste à réaliser l'estimation du premier et second moment du gradient. Pour ce faire, Adam utilise la moyenne mobile exponentielle du gradient et du gradient au carré.

Ensuite, des corrections de biais sur les estimations obtenues sont appliquées pour prendre en considération le fait que ces estimations sont initialisées à zéro. Enfin, les paramètres du réseau sont mis à jour.

---

**Algorithme 5 : Adam**


---

Nombre d'observations dans un mini-lot :  $n_b$   
 Taux de dégradation exponentielle pour l'estimation du premier moment :  $\beta_1$   
 Taux de dégradation exponentielle pour l'estimation du second moment /  $\beta_2$   
 Constante pour stabilisation numérique :  $\delta$   
 Taux d'apprentissage :  $\alpha$   
 Initialisation aléatoire des paramètres du réseau :  $\theta_0$   
 Initialisation premier variable de momentum :  $m_0 = 0$   
 Initialisation deuxième variable de momentum :  $v_0 = 0$   
 $t = 0$   
**tant que** condition d'arrêt non atteinte **faire**  
   Extraire  $n_b$  exemples de l'ensemble d'entraînement étiqueté  $(x_i, y_i)$  tirés au hasard  
   **pour** chaque couple  $(x_i, y_i)$  avec  $i = 1, \dots, n_b$  **faire**  
      $y'_i =$  Propagation en avant de  $x_i$   
      $\mathcal{J}_i(\theta_t) =$  Calcul de l'erreur entre  $y'_i$  et  $y_i$   
      $\nabla_{\theta_t} \mathcal{J}_i(\theta_t) =$  Rétro-propagation du gradient de la fonction de perte  
   **fin pour**  
    $t = t + 1$   
    $g_t = \frac{1}{n_b} \sum_{i=1}^{n_b} \nabla_{\theta_{t-1}} \mathcal{J}_i(\theta_{t-1})$   
    $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$  (Mis à jour la première estimation biaisée du moment)  
    $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$  (Mis à jour la second estimation biaisée du moment)  
    $\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$  (Correction de biais du premier moment)  
    $\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$  (Correction de biais du second moment)  
    $\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \delta}}$  (Mise à jour des paramètres)  
**fin tant que**

---

### 3.2.3 Auto-encodeur

Comme nous l'avons déjà vu dans le chapitre 2, l'auto-encodeur (AE) est un réseau de neurones non supervisé fondé sur la reconstruction des données d'entrée. La clé de ce réseau est l'apprentissage d'un espace de variables latentes de dimension réduites qui contient suffisamment d'informations pour permettre une reconstruction appropriée [155]. Le fait qu'il ne nécessite pas d'étiquettes pour être entraîné le rend polyvalent et son usage est répandu dans de nombreux domaines. Comme nous pouvons le voir dans la figure 3.3, l'architecture de l'AE est composée de deux modules principaux :

- Tout d'abord nous avons l'encodeur, où une fonction d'activation  $f$  est appliquée pour projeter la donnée d'entrée  $X$  et obtenir une représentation latente  $Z$  :

$$Z = f(WX + b_1) \quad (3.18)$$



où les poids  $W$  et les biais  $b_1$  représentent les paramètres de l'encodeur à apprendre.

- Dans la deuxième partie, nous avons le décodeur, lequel utilise une fonction d'activation  $g$  pour déployer la représentation latente  $Z$  vers  $Y$  une reconstruction de la donnée de départ dans l'espace initial :

$$Y = g(W'Z + b_2) \quad (3.19)$$

où  $W'$  et  $b_2$  représentent les paramètres entraînaables du décodeur. Dans le but de réduire la complexité du modèle, la technique de poids liés (*tied-weights*) est généralement utilisée, où les poids du décodeur sont définis comme la transposée des poids de l'encodeur, c'est-à-dire  $W' = W^T$  [175].

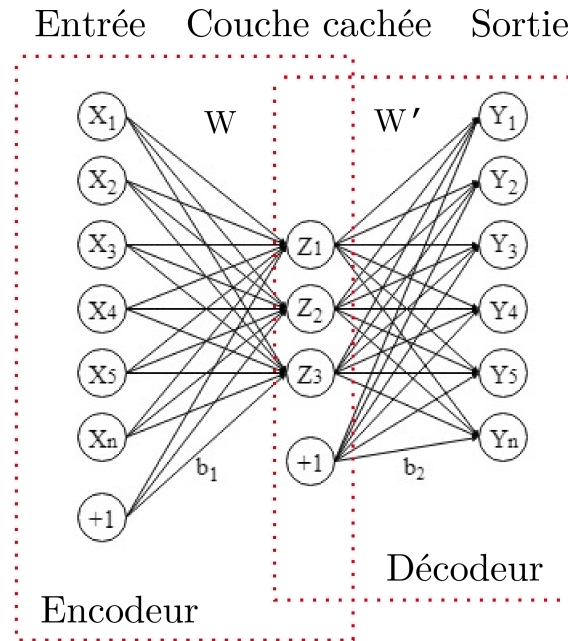


FIGURE 3.3 – Exemple de l'auto-encodeur avec une seule couche cachée.

Une des fonctions d'activation les plus utilisées dans les AEs est la fonction sigmoïde (figure 3.4) donnée par :

$$f(x) = \frac{1}{1 + e^{-x}} \quad (3.20)$$

La fonction sigmoïde est généralement utilisée comme fonction d'activation de sortie quand l'objectif est de prédire la valeur d'une variable binaire (valeurs de sortie entre 0 et 1). Bien que cette fonction soit couramment utilisée dans les architectures des AEs, elle n'est pas si répandue dans les réseaux de neurones profonds. En effet, la fonction sigmoïde sature sur la majorité de son domaine : proche de zéro pour une valeur de  $x$  très négatif et proche de 1 pour une valeur de  $x$  très positive. Cette saturation se traduit par un gradient presque nul et par conséquent, l'apprentissage fondé sur le gradient peut être très lent.

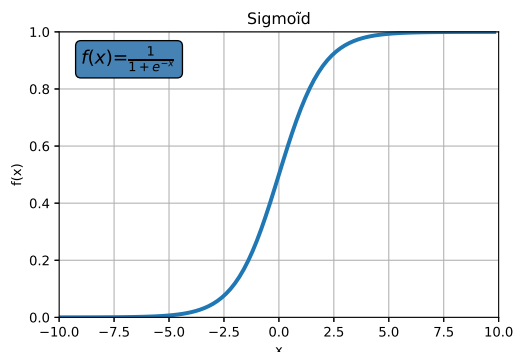


FIGURE 3.4 – Fonction sigmoïde.

Une deuxième fonction d'activation couramment utilisée est la fonction tangente hyperbolique (figure 3.5) défini comme suit :

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (3.21)$$

Cette fonction est centrée sur 0, avec des valeurs de sortie comprises entre -1 et 1. Le fait que cette fonction soit similaire à la fonction identité (pour des valeurs d'entrée proche de 0) facilite l'entraînement du réseau et par conséquent, elle est généralement plus performante que la fonction sigmoïde décrite précédemment.

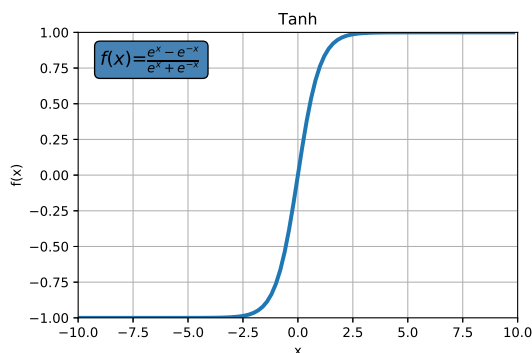


FIGURE 3.5 – Fonction tangente hyperbolique.

En plus de permettre d'obtenir des valeurs de sortie dans un intervalle borné raisonnable, l'utilisation de ces fonctions d'activation est en lien avec le théorème d'approximation universelle [176]. Ce théorème établit que toute fonction continue  $f(x)$  peut être approchée par un réseau de neurones avec une couche de sortie linéaire et au moins une couche cachée avec une fonction d'activation « écrasante », telle que la fonction tangente hyperbolique ou la fonction sigmoïde.

L'apprentissage de l'AE consiste à minimiser l'erreur de reconstruction. Cette fonction est conçue pour exprimer une pénalité d'autant plus grande que la sortie  $Y$  est différente

de l'entrée  $X$ . Dans le cas où la sortie est composée de valeurs réelles, l'erreur quadratique moyenne (*Mean Squared Error* - MSE) est couramment utilisé :

$$\mathcal{J}_{MSE}(X; \theta) = \frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2 \quad (3.22)$$

où  $\theta = (W, W', b_1, b_2)$  est l'ensemble des paramètres du réseau et  $n$  le nombre d'observations. Lorsque la sortie est binaire, la fonction de perte d'entropie croisée binaire (*Binary Cross-Entropy* - BCE) est appliquée :

$$\mathcal{J}_{BCE}(X; \theta) = -\frac{1}{n} \sum_{i=1}^n [x_i \log(y_i) + (1 - x_i) \log(1 - y_i)] \quad (3.23)$$

Dans le but de pénaliser les grands poids ( $W$ ) et ainsi éviter le sur-apprentissage, un terme de régularisation peut également être ajouté à la fonction de perte. Cette variante est connu sous le nom de dégradation des pondérations (*weight decay*) [177] :

$$\mathcal{J}(X; \theta) = \mathcal{J}(X; \theta) + \frac{\lambda}{2} \|W\|^2 \quad (3.24)$$

où  $\mathcal{J}$  représente la fonction de perte (MSE ou CE) et  $\lambda$  l'hyper-paramètre de régularisation à définir.

### 3.2.4 Machine à vecteurs de support

Les machines à vecteurs de support (SVM) sont un ensemble de méthodes d'apprentissage supervisé proposées initialement par Cortes et Vapnik [130] pour résoudre des problèmes de classification et de régression.

Étant donné un ensemble de  $n$  observations  $x$  de l'espace d'entrée  $\mathcal{X}$ , et  $y \in \{-1, 1\}$  les étiquettes de sortie relatives à ces données, le SVM est exploité pour définir l'hyperplan optimal qui sépare les données en deux classes ( $-1$  et  $1$ ). Cet hyperplan peut être défini comme l'ensemble des  $x$  qui vérifie la relation suivante :

$$w^T x + b = 0 \quad (3.25)$$

Où  $w$  et  $b$  sont les paramètres de l'hyperplan à apprendre.

Dans le cas de classes séparables linéairement, la somme des distances minimales des observations de chaque classe à leurs projections orthogonales sur l'hyperplan détermine la marge (voir figure 3.6). Plus la marge est grande, plus la capacité de généralisation du modèle est importante. L'objectif d'apprentissage du SVM est donc de maximiser cette distance. Dans le cas de classes linéairement séparables, ce classifieur peut-être obtenu en résolvant le problème d'optimisation quadratique sous contraintes linéaires :

$$\min_{w,b} \left( \frac{1}{2} \|w\|^2 \right) \quad (3.26)$$

sous la contrainte

$$y_i (w^T x_i + b) \geq 1, \quad \forall i = 1, \dots, n$$

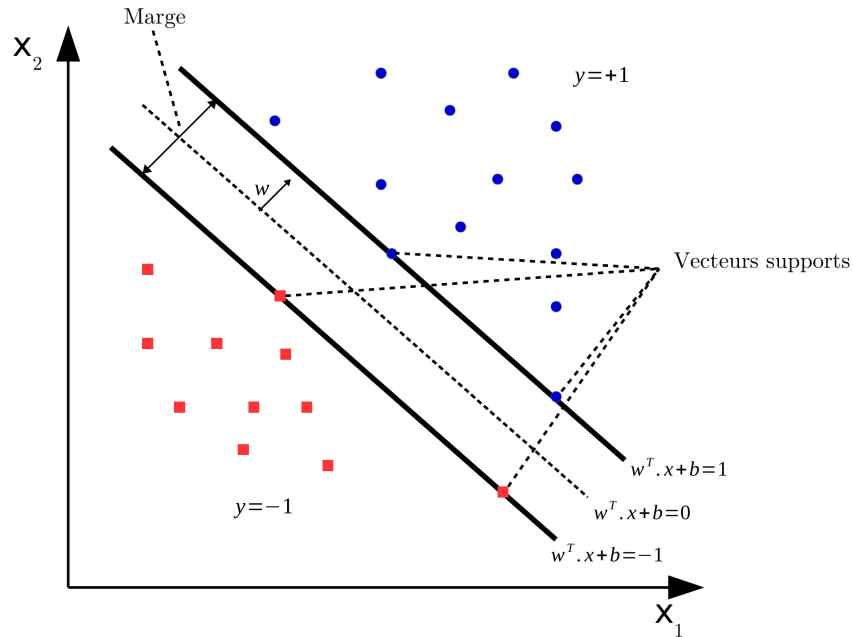


FIGURE 3.6 – Une illustration de l’hyperplan séparant les observations appartenant à deux classes dans un espace bidimensionnel (cas de classes séparables).

Cette formulation se généralise dans le cas de classes non séparables en introduisant un terme de pénalisation pour les échantillons mal placés par rapport à l’hyperplan (voir exemple de la figure 3.7). De ce fait, le problème d’optimisation quadratique sous contraintes linéaires prend la forme suivante :

$$\min_{w,b,\xi} \left( \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i \right), \quad C \geq 0 \quad (3.27)$$

sous les contraintes

$$y_i (w^T x_i + b) \geq 1 - \xi_i, \quad \forall i = 1, \dots, n$$

$$\xi_i \geq 0, \quad \forall i = 1, \dots, n$$

où  $C$  est l’hyper-paramètre de pénalisation qui peut être modifié pour accepter plus ou moins de classifications incorrectes, et  $\xi_i$  une variable de relâchement (*slack variable*).

En appliquant la méthode des multiplicateurs de Lagrange le problème dual du SVM s’exprime comme suit :

$$\max_{\alpha} \left( \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j \right) \quad (3.28)$$

sous les contraintes

$$\sum_{i=1}^n \alpha_i y_i = 0,$$

$$0 \leq \alpha_i \leq C, \quad \forall i = 1, \dots, n$$

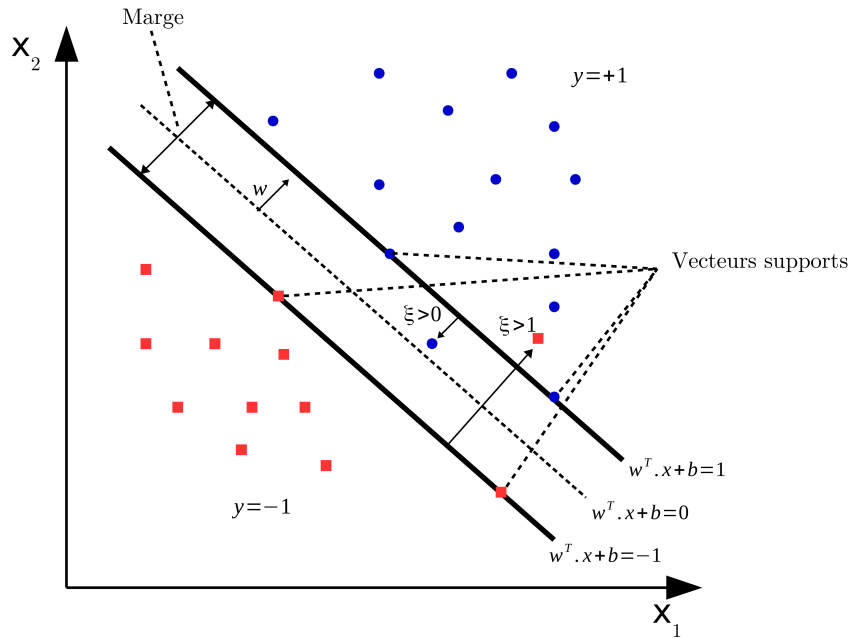


FIGURE 3.7 – Une illustration de l’hyperplan séparant les observations appartenant à deux classes dans un espace bidimensionnel (cas de classes non séparables).

où  $\alpha_i$  sont les multiplicateurs de Lagrange.

Une fois l’hyperplan optimal défini, les nouvelles observations sont classées selon la fonction de décision  $f$  donnée par :

$$f(x) = \text{sign} \left( \sum_{i \in SV} \alpha_i y_i x^T x_i + b \right) \quad (3.29)$$

où  $SV$  sont les vecteurs de support, c’est-à-dire les observation pour lesquelles l’expression  $0 < \alpha_i < C$  est vérifiée. À partir de l’équation 3.29, une observation  $x$  est affectée à la classe  $+1$  si  $f(x) \geq 0$  et à la classe  $-1$  dans le cas contraire.

Les problèmes de classification simples peuvent être résolus grâce à des séparateurs linéaires. Cependant, dans la plupart de cas réels, les données ne sont pas séparables linéairement. Afin de traiter ce type de problème, les données sont projetées vers un espace de dimension supérieure où une solution linéaire existe. La transformation d’espace est faite à partir d’une fonction de projection  $\phi(x)$  (*mapping function*), qui projette les données de l’espace d’entrée  $\mathcal{X}$  vers un nouvel espace de caractéristiques  $\mathcal{H}$ , couramment appelé espace de redescription. En appliquant ce principe, l’équation 3.28 peut être récrit sous la forme :

$$\max_{\alpha} \left( \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \langle \phi(x_i), \phi(x_j) \rangle \right) \quad (3.30)$$

Dans cette équation, la fonction à maximiser dépend du produit scalaire  $\langle \phi(x_i), \phi(x_j) \rangle$ .

Ce produit scalaire est réalisé dans un espace de grande dimension, ce qui rend son calcul impraticable. Les auteurs Aizerman et al. [178] ont montré que sous certaines conditions des fonctions bivariées peuvent correspondre à un produit scalaire dans un espace de redescription  $\mathcal{H}$ . L'utilisation de fonctions noyaux ( $K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$ ) permet donc d'exprimer le problème et sa solution sans avoir à expliciter la fonction de projection  $\phi$  (astuce du noyau). Le tableau 3.1 regroupe les fonctions noyaux les plus couramment utilisées dans la littérature.

TABLEAU 3.1 – Les fonctions à noyaux les plus couramment utilisées.

Noyaux	Fonction à noyau $K(x_i, x_j)$
Linéaire	$\langle x_i, x_j \rangle$
Polynomial (de degré $p$ )	$(\langle x_i, x_j \rangle + d)^p$
Gaussien	$e^{-\frac{\ x_i - x_j\ _2^2}{2\sigma^2}}$

### Machine à vecteurs de support mono-classe

Le SVM a initialement été proposé pour résoudre des tâches de classification bi-classe. Cependant, une adaptation a été proposée par Schölkopf et al. [179, 180] afin d'utiliser le SVM dans le contexte d'une classification à classe unique. Cette variante du SVM est désignée par *One-class SVM* (OC-SVM). Comme nous pouvons le voir dans la figure 3.8, la stratégie adoptée consiste à séparer les données de l'origine avec une marge maximale, donnée par  $\frac{\rho}{\|w\|}$ . La maximisation de la marge peut alors se faire en minimisant  $-\rho$  et  $\|w\|$

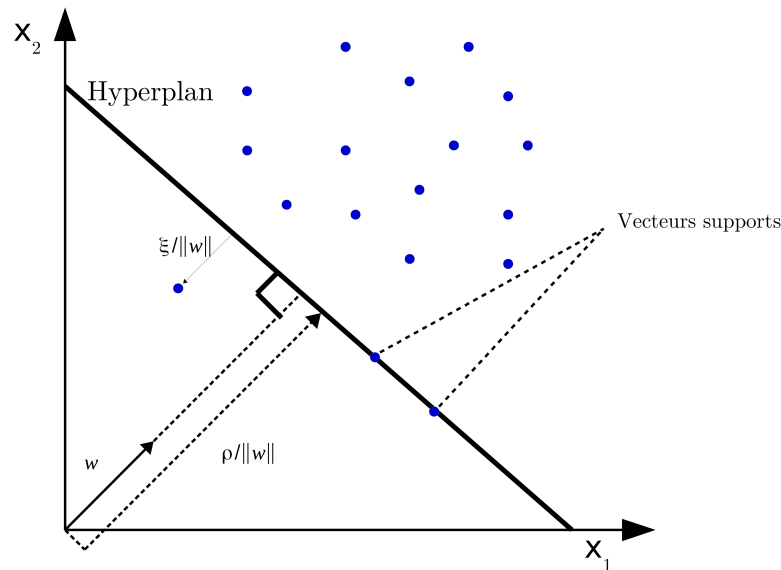


FIGURE 3.8 – Une illustration de l'hyperplan séparant les observations de l'origine dans un espace bidimensionnel.

à partir du problème quadratique sous contraintes suivant :

$$\min_{w, \xi, \rho} \left( \frac{1}{2} \|w\|^2 + \frac{1}{\nu n} \sum_{i=1}^n \xi_i - \rho \right) \quad (3.31)$$

sous les contraintes

$$\begin{aligned} \langle w, \phi(x_i) \rangle &\geq \rho - \xi_i, \quad \forall i = 1, \dots, n \\ \xi_i &\geq 0, \quad \forall i = 1, \dots, n \end{aligned}$$

L'hyper-paramètre  $\nu \in [0, 1]$ , qui doit être défini, représente la borne supérieure de la fraction des observations anormales et la borne inférieure de la proportion de vecteurs support.

En utilisant la méthode des multiplicateurs de Lagrange et en remplaçant le produit scalaire par une fonction noyau, on obtient le problème d'optimisation dual suivant :

$$\min_{\alpha} \left( \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j K(x_i, x_j) \right) \quad (3.32)$$

sous les contraintes

$$\begin{aligned} \sum_{i=1}^n \alpha_i &= 1, \\ 0 &\leq \alpha_i \leq \frac{1}{\nu n}, \quad \forall i = 1, \dots, n \end{aligned}$$

Finalement, la fonction de décision est donnée par :

$$f(x) = \text{sign} \left( \sum_{i=1}^n \alpha_i K(x, x_i) - \rho \right) \quad (3.33)$$

À partir de cette fonction, une nouvelle observation  $x$  est classée selon la valeur de  $f(x)$  obtenue :

- $x$  est normale si  $f(x) \geq 0$ .
- $x$  est anormale si  $f(x) < 0$ .

Lorsque la fonction noyau utilisée dans le processus d'apprentissage satisfait  $K(x, x) = \text{constante}$  (par exemple le noyau Gaussien avec  $K(x, x) = 1$ ), la méthode OC-SVM devient identique à la méthode SVDD (*Support Vector Data Description*) proposée par Tax et Duin [181]. Dans ce contexte, la frontière de décision calculée prend la forme d'une hypersphère avec un rayon minimum qui englobe la plupart des échantillons d'apprentissage.

### 3.3 Création de la base de données

Afin de définir notre système d'analyse, nous avons créé une base de données (BDD) contenant des pommes de terre réparties en six classes différentes, c'est-à-dire saines, endommagées, vertes, dartrose, gale commune et rhizoctone. Le système décrit dans les sections 1.1 et 2.2.1 a été utilisé pour acquérir les images constituant la base : une caméra couleur avec un capteur CMOS placée dans un dôme qui contient deux panneaux LEDs pour éclairer la scène. Ce système a été employé pour obtenir 4 images pour chaque pomme de terre, une image par face ou côté. Durant la prise de vues, les pommes de terre sont placées sur un tapis roulant de couleur noire qui les transporte et les fait tourner. La figure 3.9 montre un exemple d'images fournies par la machine. La résolution des images obtenues est de  $1602 \times 882$ . Plusieurs variétés de pommes de terre jaunes, telles que



FIGURE 3.9 – Exemple d'image fournie par le système d'acquisition d'images.

l'Agata, la Monalisa, la Gourmandine, l'Annabelle, la Caesar, la Charlotte et la Marilyn ont été incluses pour générer un ensemble de données aussi varié que possible. Au total, 2422 tubercules ont été inclus dans la base. En tenant compte du fait qu'il y a 4 images par tubercule, on obtient un ensemble de données final de 9688 images. Afin d'entraîner et de tester les méthodes, un expert de l'entreprise et moi-même avons annoté manuellement toutes les images de deux manières différentes.

Dans le premier cas de figure, une classification par pomme de terre a été effectuée, en considérant les images des 4 faces provenant du même tubercule. Dans ce contexte, les images ont été divisées en 8 classes : saine, endommagée légère, endommagée grave, verte légère, verte grave, dartrose, gale commune et rhizoctone. La décision de ne diviser que les images endommagées et vertes par gravité est une conséquence du manque de données pour les autres classes de défauts. Le tableau 3.2 donne le nombre d'images pour chaque classe.

Dans le second cas, chaque image représentant une face de pomme de terre a été annotée individuellement pour être classée dans l'une des 6 classes différentes, sans distinction entre les niveaux de gravité. Comme le montre le tableau 3.3, l'ensemble de données



TABLEAU 3.2 – Base de données des images de pommes de terre comprenant les quatre faces du même tubercule en considération (voir figure 3.9).

Classe	Nombre d'images
Saine	831
Endommagée légère	341
Endommagée grave	159
Verte légère	161
Verte grave	349
Dartrose	151
Gale commune	359
Rhizoctone	71
Total	2422

d'images par face a été divisé en 5325 images de pommes de terre saines, 984 d'endommagées, 1263 de vertes, 597 de dartrose, 1276 de gale commune et 243 de rhizoctone. Afin de tester les différentes méthodes proposées, 30% de l'ensemble de données a été sélectionné de manière aléatoire pour construire un ensemble de test. Les pommes de terre qui composent cet ensemble de test peuvent appartenir à plusieurs classes simultanément (1 classe par face). Le reste de la base de données a servi à l'apprentissage et à la validation des modèles. Il est nécessaire de noter que les 4 images provenant de la même pomme de terre ont été intégrées dans le même ensemble d'entraînement ou de test afin de conserver l'indépendance des ensembles.

TABLEAU 3.3 – Base de données constituée d'images de faces de pommes de terre (chaque face est annotée séparément).

Classe	Nombre de images
Saine	5325
Endommagée	984
Verte	1263
Dartrose	597
Gale commune	1276
Rhizoctone	243
Total	9688

Étant donné que les images proviennent de différentes machines avec des étalonnages qui ne sont pas identiques, une étape de prétraitement a été introduite afin de normaliser les images. Tout d'abord, une image correspondant à une face de pomme de terre a été choisie et considérée comme « modèle ». Cette image modèle a ensuite été utilisée pour calculer la moyenne de l'intensité des canaux R, V et B, sans tenir compte de l'arrière-plan. Ces valeurs moyennes ( $R_m$ ,  $V_m$  et  $B_m$ ) ont ensuite été utilisées pour réaliser une normalisation sur le reste des images. Les canaux de chaque nouvelle image I ( $I_{R'}$ ,  $I_{V'}$  et

$I_{B'}$ ) sont alignés par rapport aux valeurs moyennes  $R_m$ ,  $V_m$  et  $B_m$  par mise à l'échelle :

$$I_{R'}^{norm} = I_{R'} \times \frac{R_m}{R'_m} \quad (3.34)$$

$$I_{V'}^{norm} = I_{V'} \times \frac{V_m}{V'_m} \quad (3.35)$$

$$I_{B'}^{norm} = I_{B'} \times \frac{B_m}{B'_m} \quad (3.36)$$

où  $R'_m$ ,  $V'_m$  et  $B'_m$  sont les moyennes de chaque canal de l'image à normaliser  $I$ .

Notre méthode permet également de localiser les régions de défauts des pommes de terre classées endommagées ou vertes afin de les catégoriser par gravité. Pour accomplir cette tâche de localisation des défauts, deux bases de données annotées par patches ont été créées. La première contient 29657 patches de taille  $16 \times 16$  extraits de manière aléatoire à partir de 168 images normales (ne contenant aucun défaut). Tous les patches contenant des pixels de l'arrière-plan, c'est-à-dire des pixels noirs, ont été exclus de cette base. En effet, des expérimentations ont montré que ces patches sont confondus avec des patches affectés de défauts à cause de la ressemblance des pixels noirs de l'arrière plans avec des pixels très sombres représentant certains défauts.

La deuxième base, quant à elle, est formée de deux sous ensembles de patches. Le premier sous ensemble est défini en séparant les patches de 100 images provenant de la classe verte en deux catégories : patches verts et patches non verts. Ce sous ensemble contient 1271 patches verts et 7722 patches non verts. Le deuxième sous ensemble est obtenu, quant à lui, en séparant les patches de 115 images de la classe endommagée en deux catégories : patches endommagés et non endommagés. Ce second sous ensemble contient 3962 patches endommagés et 14249 patches non endommagés. Dans la figure 3.10 nous pouvons observer une image divisée par patch. Dans ce cas, la face de la pomme de terre a été divisée en 122 patches, sans compter les patches contenant des pixels de fond. Il est important de noter que l'étiquetage par patch est beaucoup plus rapide et facile que l'étiquetage par pixel.

## 3.4 Système proposé reposant sur l'apprentissage supervisé

La figure 3.11 présente un aperçu de la méthode proposée. La méthode se compose de trois phases principales :

1. Tout d'abord, un CNN est entraîné pour classer les images représentant les faces des pommes de terre (figure 3.11 A).
2. Deuxièmement, une combinaison d'un auto-encodeur et d'un classifieur SVM est appliquée afin de localiser les régions affectées par les défauts grâce à une classification des patches de l'image. Cette étape de localisation est uniquement appliquée

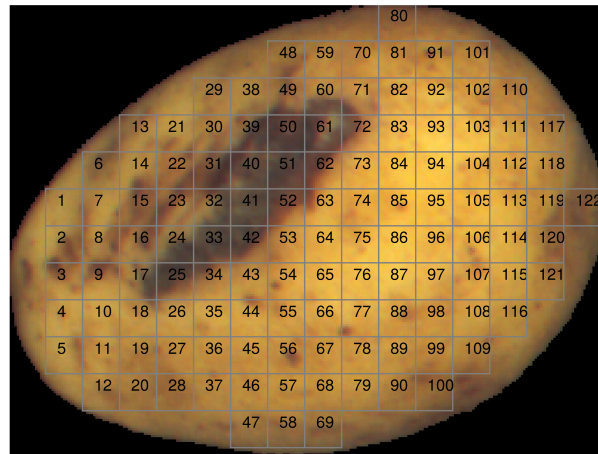


FIGURE 3.10 – Image divisée par patches afin d’aider à l’annotation manuelle.

aux images classées endommagées où vertes lors de la première étape (figure 3.11 B).

3. Enfin, dans la troisième phase, les résultats de localisation de la phase précédente sont utilisés pour entraîner deux SVM qui classent les pommes de terre endommagées et vertes par gravité (figure 3.11 C).

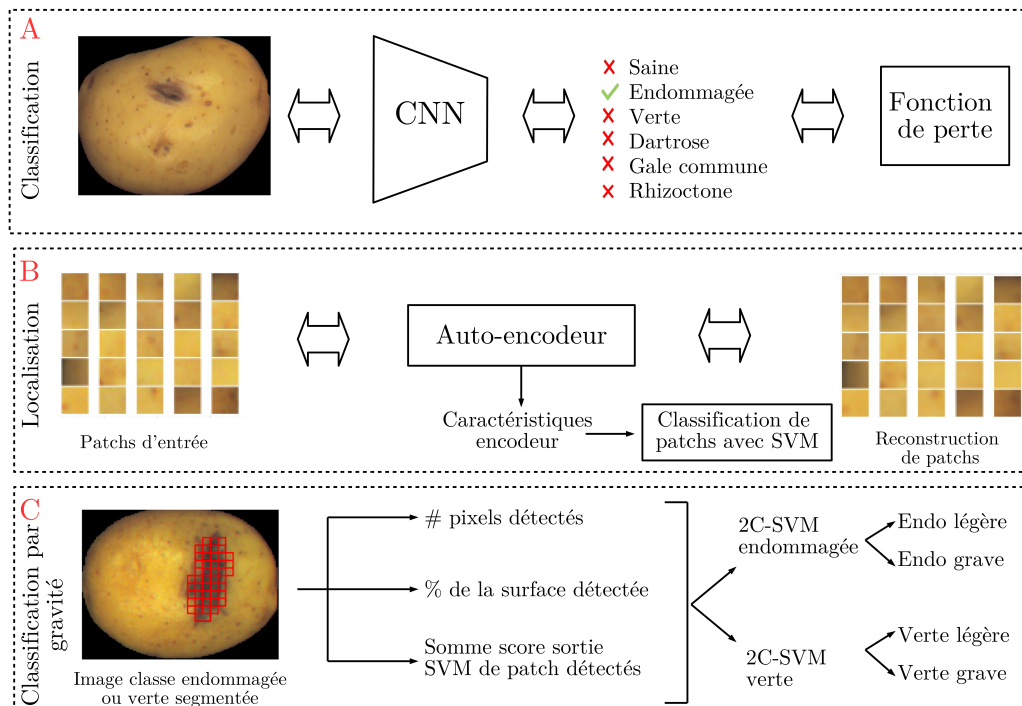


FIGURE 3.11 – Schéma de la méthode proposée basée sur l’apprentissage supervisé.

Le plan de cette section suit ces trois étapes de traitement en approfondissant leur présentation.

### 3.4.1 Classification des défauts par face de pomme de terre

Dans la première phase, un CNN pré-entraîné est modifié et affiné (*finetuned*) grâce à notre ensemble de données d'apprentissage pour classer les images de faces de pommes de terre en 6 classes distinctes : saine, endommagée, verte, dartrose, gale commune et rhizoctone. Trois réseaux de neurones profonds pré-entraînés sont testés afin de sélectionner celui qui convient le mieux à notre problème : AlexNet [3], VGG-16 [47] et GoogLeNet [5]. Tous ces réseaux ont été initialement entraînés sur la base de données ImageNet [141], une base de donnée de plus de 1 million d'images appartenant à 1000 classes différentes. AlexNet, dont l'architecture peut être observée dans la figure 3.12, est composé de 5 couches de convolution, à chaque couche de convolution est appliquée une fonction d'activation ReLU. Le réseau contient également 3 couches de *max-pooling*. Dans le bloc de classification, trois couches entièrement connectées et une fonction *softmax* sont utilisées. Le réseau compte 62,3 millions de paramètres, dont la majorité sont contenus dans les couches entièrement connectées.

Le réseau nommé VGG-16, présenté dans la figure 3.13, a été développé par le «*Visual*

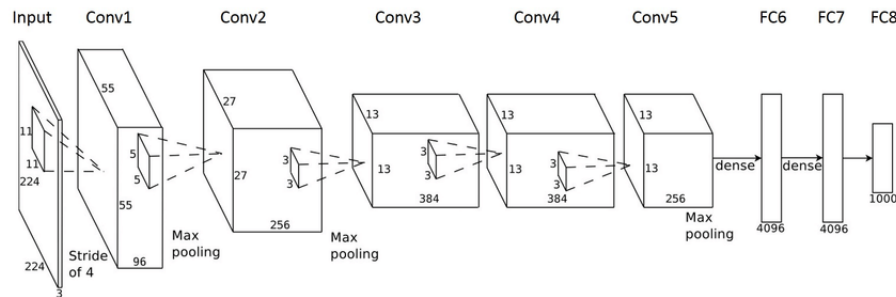


FIGURE 3.12 – Architecture du réseau AlexNet [3] (image extraite de [4]).

*Geometry Group*» de l'université d'Oxford. Grâce à son architecture, les auteurs ont démontré l'impact de la profondeur du réseau sur ses performances. Le VGG-16 compte 13 couches de convolution (chacune liée à une fonction ReLU), 5 couches *max-pooling* et 3 couches entièrement connectées, avec une couche *softmax* comme fonction de sortie. Tous les filtres de convolution ont une taille de  $3 \times 3$  et utilisent un pas (*stride*) de 1. Malgré d'excellents résultats obtenus dans le concours ILSVRC 2014, son grand nombre de paramètres (environ 138 millions) complique considérablement son apprentissage.

Le troisième réseau, GoogLeNet, possède quant à lui, 22 couches de profondeur pour un total de 6,8 millions de paramètres (figure 3.14). Ce réseau est réputé grâce à l'introduction des 9 modules d'inception présentés dans la figure 3.15. Un des intérêts de ces modules est de couvrir une plus grande surface de l'image avec des filtres de convolution de taille plus élevée, mais également de conserver des résolutions fines pour les petits motifs en utilisant des filtres de taille réduite. L'idée consiste donc à appliquer plusieurs opérations de convolution en parallèle en utilisant des filtres de tailles différentes ( $1 \times 1$ ,  $3 \times 3$  et  $5 \times 5$ ). Des couches de convolution avec des filtres de taille  $1 \times 1$  et un nombre de filtres inférieur au nombre de cartes de caractéristiques de la couche précédente sont également insérées avant les couches de convolution  $3 \times 3$  et  $5 \times 5$ . Cela permet de définir

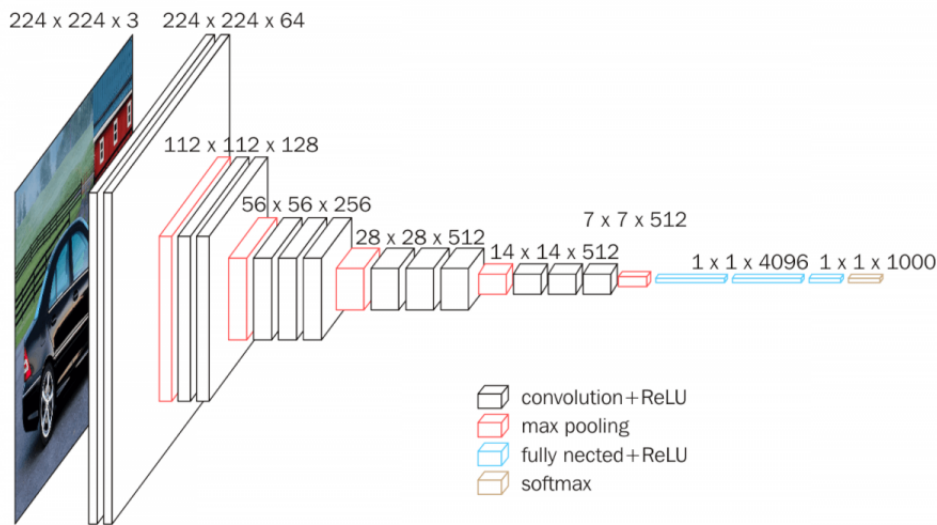


FIGURE 3.13 – Architecture du réseau VGG-16 (image obtenue de <https://neurohive.io/en/popular-networks/vgg16/>).

de nouvelles cartes de caractéristiques par combinaison des précédentes et ainsi d'en réduire le nombre. En réduisant le nombre de cartes de caractéristiques le coût calculatoire est réduit pour la suite du réseau.

D'autre part, une couche de convolution  $1 \times 1$  avec un nombre de filtres supérieur au nombre de cartes de caractéristiques de la couche précédente est introduite après la couche de *max-pooling*. Le nombre de cartes de caractéristiques après le regroupement est alors augmenté, créant artificiellement plus de projections du contenu des cartes de caractéristiques sous-échantillonnées.

Outre les modules d'inception, un autre aspect important de l'architecture de ce réseau est la substitution des couches entièrement connectées du bloc de classification par une couche de *Global Average Pooling*. Cela permet d'obtenir un réseau très profond tout en limitant le nombre de paramètres entraînaibles.

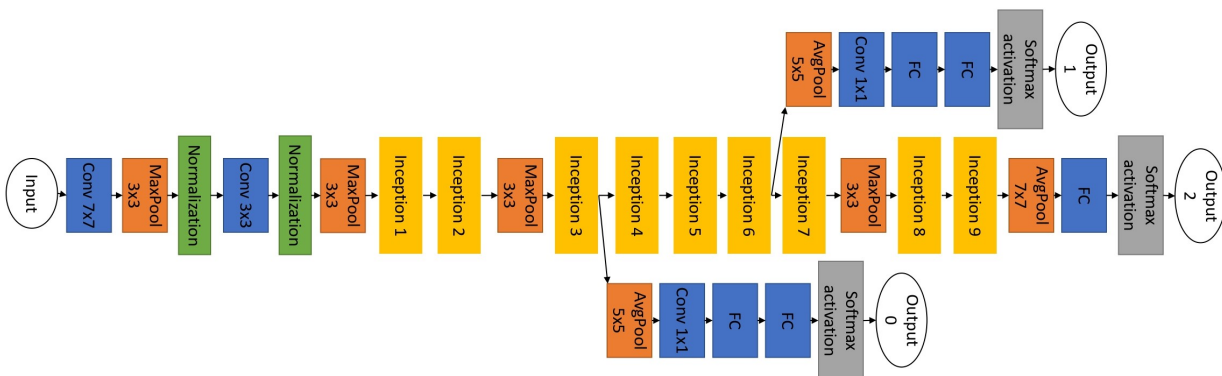


FIGURE 3.14 – Architecture du réseau GoogLeNet [5].

Afin d'entraîner chaque réseau (AlexNet, VGG-16 et GoogLeNet), nous avons remplacé la dernière couche entièrement connectée, définie à l'origine pour les 1000 classes de la

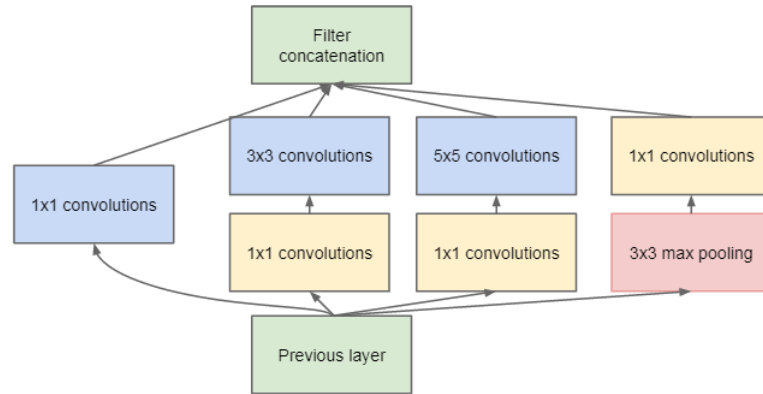


FIGURE 3.15 – Module d'inception introduit dans GoogLeNet [5].

base de données ImageNet, par une nouvelle couche qui permet de classer les images en 6 catégories : saine, endommagée, verte, dartrose, gale commune et rhizoctone. Le CNN est non seulement utilisé pour classer les images des faces de pommes de terre, mais également sélectionner les images qui passeront à l'étape suivante qui consiste à localiser des défauts. Un des avantages de l'utilisation du CNN est que l'image est analysée de manière globale, ce qui nous permet de prendre en compte les motifs décisifs pour l'attribution d'une classe mais également le contexte global autour de ces motifs. Cela n'aurait pas été possible dans le cas d'une analyse de l'image reposant uniquement sur des patches (cette solution a été testée, mais avec des performances insatisfaisantes). De plus, cette classification avec le CNN permet également de réduire considérablement le nombre d'images qui seront traitées dans la deuxième étape de localisation. En effet, la classification nous permet de distinguer les pommes de terre endommagées et vertes des autres catégories et de cibler uniquement ces deux classes pour la localisation des défauts.

### Analyse comparative

Afin d'évaluer les performances du CNN, nous avons proposé comme éléments de comparaison des approches conventionnelles reposant sur une extraction ciblée de caractéristiques. Les caractéristiques de couleur et de texture sont extraites des images de pommes de terre afin d'être utilisées comme données d'entrée d'un classifieur SVM. Trois espaces couleur ont été utilisés : RVB, TSV et CIELAB. La moyenne, la variance, l'asymétrie et l'entropie sont extraites à partir de neuf canaux différents. La matrice de co-occurrence des niveaux de gris (GLCM) [126] de chaque canal est également utilisée. Pour une image  $I$  de taille  $(n, m)$  avec  $g$  niveaux de gris, la matrice de co-occurrence de taille  $g \times g$  paramétrée par un voisinage  $(\Delta u, \Delta v)$  est définie comme suit :

$$C_{[\Delta u, \Delta v]}(i, j) = \sum_{u=1}^n \sum_{v=1}^m \mathbb{1}[I(u, v) = i \ \& \ I(u + \Delta u, v + \Delta v) = j] \quad (3.37)$$

À partir de cette matrice, les caractéristiques de texture incluant le contraste, la corrélation, l'énergie et l'homogénéité ont été extraits.

La définition analytique des attributs est donnée dans le tableau 3.4.

TABLEAU 3.4 – Caractéristiques de couleur et de texture extraites à partir des images RVB, TSV et CIELAB. Remarques :  $C$  = Matrice de co-occurrence des niveaux de gris de taille  $(g \times g)$ ,  $I$  image en niveaux de gris de taille  $(n \times m)$  et  $p_k$  la probabilité estimée d'un pixel d'être associé au niveau d'intensité  $k$ , avec  $k = 0, \dots, 255$  pour une image de 8-bits.

Caractéristique	Calcul
Moyenne	$\mu = \frac{1}{n \times m} \sum_{u=1}^n \sum_{v=1}^m I_{u,v}$
Variance	$\sigma^2 = \frac{1}{n \times m} \sum_{u=1}^n \sum_{v=1}^m (I_{u,v} - \mu)^2$
Asymétrie	$A = \frac{\frac{1}{n \times m} \sum_{u=1}^n \sum_{v=1}^m (I_{u,v} - \mu)^3}{\left(\frac{1}{n \times m} \sum_{u=1}^n \sum_{v=1}^m (I_{u,v} - \mu)^2\right)^{3/2}}$
Entropie	$E = - \sum_{k=0}^{255} p_k \log(p_k)$
Contraste	$Co = \sum_{i,j=1}^g C_{i,j} (i - j)^2$
Corrélation ( $Corr$ )	$Corr = \sum_{i,j=1}^g C_{i,j} \left[ \frac{(i - \mu_x)(j - \mu_y)}{\sigma_x \sigma_y} \right]$ $C_x(i) = \sum_{j=1}^g C(i, j) \quad C_y(j) = \sum_{i=1}^g C(i, j)$ $\mu_x = \sum_{i=1}^g i \cdot C_x(i) \quad \mu_y = \sum_{j=1}^g j \cdot C_y(j)$ $\sigma_x^2 = \sum_{i=1}^g (i - \mu_x)^2 C_x(i) \quad \sigma_y^2 = \sum_{j=1}^g (j - \mu_y)^2 C_y(j)$
Énergie	$En = \sum_{i,j=1}^g C_{i,j}^2$
Homogénéité	$Ho = \sum_{i,j=1}^g \frac{C_{i,j}}{1 + (i - j)^2}$

### 3.4.2 Localisation de défauts

Afin de classer les images de pommes de terre vertes et endommagées par gravité, l'étendue de la surface affectée par le défaut doit être estimée. Des patches de taille  $16 \times 16$  sont extraits des images afin de les utiliser comme entrée d'un auto-encodeur. Ce réseau est entraîné dans le but d'apprendre une représentation plus compacte, robuste et pertinente des patches. Une fois l'AE entraîné, les caractéristiques extraites de la partie encodage sont utilisées comme entrée d'un classifieur binaire SVM ( $2C-SVM_{Dét}$ ). Pour les deux cas, c'est-à-dire détection des patches endommagés et détection des patches verts, un classifieur  $2C-SVM_{Dét}$  est entraîné.

Dans le but d'obtenir une localisation précise de chaque défaut, les patches extraits se chevauchent. En effet, lors de l'extraction d'un nouveau patch la fenêtre est translatée de 8 pixels à partir de sa position précédente (soit la moitié de sa taille). La complexité calculatoire élevée générée par le chevauchement des patches est limitée par le fait que seules les images classées par le CNN comme endommagées ou vertes à l'étape précédente sont traitées.

### 3.4.3 Classification de défauts par gravité

Après les étapes précédentes de classification et de localisation, des informations relatives aux patches identifiés comme abîmés (endommagés ou verts) sont extraites et utilisées pour entraîner deux classifieurs  $2C-SVM_{Grav}$ , un pour chaque type de défaut. L'objectif de ces classifieurs est de classer les images de pommes de terre endommagées et vertes par gravité, à savoir défaut léger ou grave. Les données d'entrée des classifieurs  $2C-SVM_{Grav}$  sont :

1. Le nombre de patches détectés comme étant affectés par un défaut. Nous ne prenons pas en compte les patches affectés qui sont isolés, c'est-à-dire sans patches voisins également affectés.
2. Le pourcentage de la surface affectée par rapport à la surface totale de la pomme de terre ( $\frac{NPD}{NPT}$ , où  $NPD$  = nombre de patches détectés et  $NPT$  = nombre total de patches).
3. La somme des scores de sortie du classifieur  $2C-SVM_{Dét}$  pour les patches détectés.

Étant donné que dans la base de données, l'information relative à la gravité du défaut est disponible au niveau de la pomme de terre (une seule étiquette relative à la gravité du défaut pour les 4 faces de chaque pomme de terre) et que notre algorithme analyse séparément chaque face, les attributs utilisés sont ceux de la face présentant le plus grand défaut (le défaut qui occupe le plus de surface relative). Cette décision est justifiée par le fait qu'un même défaut peut être visible sur plus d'une face. Dans la figure 3.16, on peut apercevoir une image de pomme de terre où un même défaut est visible sur deux faces différentes (en haut à gauche et en bas à droite). Dans ce cas précis, considérer les défauts présents sur toutes les faces pour caractériser la gravité du défaut sur la totalité d'une



pomme de terre serait erroné puisque le même défaut serait comptabilisé deux fois (une fois sur chacune des deux faces). Ce phénomène est notamment dû au fait que la capture des faces n'est pas totalement contrôlée. En effet, les formes irrégulières des pommes de terres ainsi que le fait qu'elles soient transportées par un tapis roulant au moment de la prise des images, peut expliquer la présence de chevauchement entre les images des différentes faces.

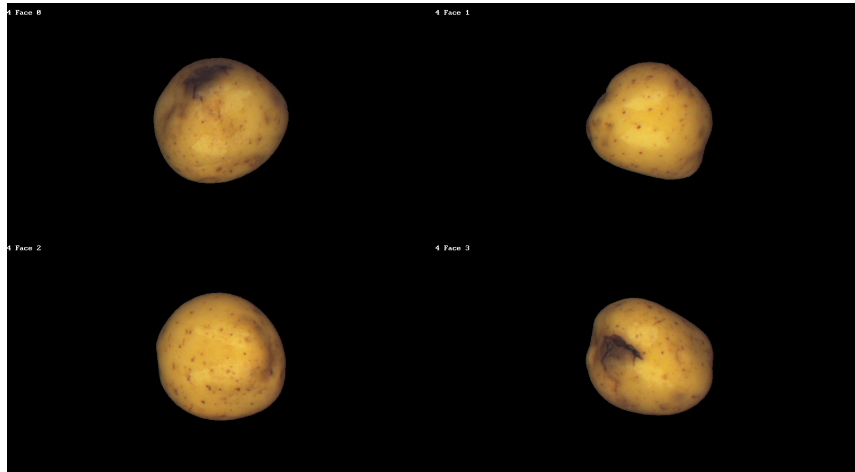


FIGURE 3.16 – Pomme de terre endommagée où le même défaut est visible sur deux faces.

Dans la section qui va suivre, section 3.5, nous allons évaluer la méthode proposée. Pour rappel, cette méthode comporte les étapes suivantes :

Tout d'abord un CNN est utilisé afin de classer chaque face de pomme de terre en 6 classes (saine, endommagée, verte, dartrose, gale commune et rhizoctone). Une fois cette classification obtenue, les faces catégorisées en vertes et endommagées sont sélectionnées pour la seconde étape qui consiste à localiser les défauts (verdissement et endommagement) présents sur chacune des faces. Compte tenu de la taille réduite de notre ensemble de données d'apprentissage pour cette étape, nous avons écarté l'utilisation des CNNs et nous avons opté pour une combinaison d'un auto-encodeur et de deux SVMs. Une fois les verdissements et les endommagements détectés, une dernière étape de classification par gravité de ces défauts est effectuée en utilisant là aussi deux SVM (un pour chaque défaut). Ici encore l'utilisation du SVM est justifiée par la taille réduite de l'ensemble d'apprentissage ainsi que la dimension réduite des données d'entrée (chaque vecteur est constitué uniquement de trois attributs).

### 3.5 Résultats expérimentaux

Nous allons juger de la pertinence de la méthode proposée en évaluant chacune des trois étapes détaillées dans les sections précédentes. Nous allons également comparer la méthode que nous proposons avec des méthodes fondées sur une extraction ciblée de caractéristiques, en terme de classification des défauts et des maladies affectant les pommes de terres. L'implémentation de la méthode a été faite en Matlab R2017b. En termes de

ressources matérielles, un GPU NVIDIA GeForce GTX 1050 Ti (4 Go de mémoire) a été utilisé.

### 3.5.1 Critères d'évaluation

Les critères d'évaluation utilisés dans ce travail de thèse ont été choisis afin de tenir compte de la nature déséquilibrée de l'ensemble de données [182]. En effet, les différentes classes ne sont pas représentées avec le même nombre d'échantillons d'entraînement dans nos bases de données. Ces critères d'évaluation sont décrits dans ce qui suit :

- Matrice de confusion : matrice carrée utilisée pour comparer les étiquettes prédites par le modèle avec les étiquettes réelles. Dans notre cas, chaque colonne représente la classe réelle et chaque ligne représente la prédiction du classifieur.
- Précision<sub>*m*</sub> : est le rapport entre le nombre d'observations prédites correctement comme appartenant à la classe *m* et le nombre total d'observations prédites comme appartenant classe *m* :

$$P_m = \frac{VP_m}{VP_m + FP_m} \quad (3.38)$$

avec  $VP_m$  le nombre de vrai positifs de la classe *m* et  $FP_m$  le nombre de faux positifs de la classe *m*.

- Rappel<sub>*m*</sub> : est le rapport entre le nombre d'observations prédites correctement comme appartenant à la classe *m* et le nombre réel d'observations de la classe *m*.

$$R_m = \frac{VP_m}{VP_m + FN_m} \quad (3.39)$$

avec  $FN_m$  le nombre de faux négatifs de la classe *m*.

- F<sub>1</sub>-mesure<sub>*m*</sub> [183] : est la moyenne harmonique de la précision et du rappel de la classe *m*.

$$F_1\text{-mesure}_m = 2 * \frac{P_m * R_m}{P_m + R_m} \quad (3.40)$$

Dans la phase de localisation, nous utilisons le taux de fausses alarmes (TFA) et le taux de non détection (TND) calculés comme suit :

$$TFA = \frac{\text{Nombre de faux positifs}}{\text{Nombre total de négatifs}} \quad (3.41)$$

$$TND = \frac{\text{Nombre de faux négatifs}}{\text{Nombre total de positifs}} \quad (3.42)$$

### 3.5.2 Résultats de la classification

Nous avons entraîné trois réseaux CNNs : AlexNet [3], VGG-16 [47] et GoogLeNet [5] à l'aide du *finetuning*. Les images d'entrée ont été redimensionnées à une taille fixe adaptée à l'architecture du réseau utilisée ( $227 \times 227$  pour AlexNet et  $224 \times 224$  pour VGG-16 et GoogLeNet). Afin d'accroître la variabilité et le nombre d'observations utilisées pour entraîner les réseaux, des techniques d'augmentation de données telles que la rotation et le retournement horizontal et vertical ont été appliquées de façon aléatoire. La descente du gradient stochastique par mini-lots avec l'hyper-paramètre d'accélération fixé à 0,9 et la fonction de perte d'entropie croisée ont été utilisées pour entraîner les trois réseaux. La taille du mini-batch a été fixée à 10 en raison de la limitation de nos ressources matérielles, notamment la taille de la mémoire du GPU. L'hyper-paramètre  $\lambda$  de *weight decay* a été fixé à  $1 \times 10^{-4}$ . Le taux d'apprentissage de la nouvelle couche entièrement connectée a été défini 20 fois supérieur au taux d'apprentissage global fixé quant à lui à  $1 \times 10^{-4}$ . Cette stratégie a été adoptée du fait que les couches pré-entraînées sont plus susceptibles d'avoir appris des caractéristiques génériques utiles à une large variété de tâches de reconnaissance de formes. Par conséquent, les paramètres de ces couches ne nécessitent pas de modifications majeures, à l'inverse de la couche entièrement connectée rajoutée par nos soins et dont les poids ont été initialisés aléatoirement.

L'apprentissage a été programmé pour s'arrêter automatiquement si la perte de validation venait à augmenter sur dix *epochs*<sup>1</sup> successives afin d'éviter le sur-apprentissage. La technique de validation croisée à 5 «*folds*» a été utilisée. Nous avons divisé l'ensemble d'entraînement en cinq parties égales et nous avons entraîné le réseau en utilisant quatre parties, laissant le reste pour valider les résultats. Le processus a été répété cinq fois pour obtenir la moyenne et l'écart-type de la précision, du rappel et de la  $F_1$ -mesure introduites dans le tableau 3.5.

Nous pouvons constater que les meilleurs résultats ont été obtenus avec le réseau GoogLeNet atteignant une valeur moyenne de  $F_1$ -mesure de 0,94 contre 0,92 et 0,88 pour AlexNet et VGG-16 respectivement. La classe verte a obtenu la meilleure valeur de  $F_1$ -mesure avec toutes les architectures : 0,96, 0,95 et 0,98 avec AlexNet, VGG-16 et GoogLeNet respectivement. Les performances les plus faibles ont été obtenues pour la classe dartrose avec les trois architectures différentes, ce qui peut naturellement s'expliquer par la similitude entre les symptômes de la dartrose et la peau rugueuse de certaines pommes de terre saines.

Nous pouvons analyser les confusions entre classes à l'aide des matrices de confusion présentées dans le tableau 3.6. Comme nous l'avions présagé, la classe dartrose a été principalement confondue avec la classe saine. Ce type de confusion existe également lorsque les opérateurs classent les tubercules manuellement.

---

1. un *epoch* est accompli quand tout l'ensemble de données d'apprentissage est propagé une seule fois vers l'avant puis vers l'arrière dans le réseau.

TABLEAU 3.5 – Moyenne et écart-type de la précision, le rappel et la  $F_1$ -mesure obtenues par classe à partir de la validation croisée sur les trois réseaux. Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

Critères	Classes	AlexNet	VGG-16	GoogLeNet
Précision	S	$0,93 \pm 0,03$	$0,91 \pm 0,02$	<b><math>0,95 \pm 0,02</math></b>
	E	$0,94 \pm 0,03$	$0,93 \pm 0,02$	<b><math>0,96 \pm 0,02</math></b>
	V	<b><math>0,99 \pm 0,01</math></b>	$0,98 \pm 0,01$	$0,98 \pm 0,01$
	D	$0,86 \pm 0,05$	$0,86 \pm 0,08$	<b><math>0,93 \pm 0,06</math></b>
	GC	$0,96 \pm 0,02$	$0,94 \pm 0,03$	<b><math>0,97 \pm 0,02</math></b>
	R	$0,91 \pm 0,06$	$0,85 \pm 0,03$	<b><math>0,92 \pm 0,04</math></b>
Rappel	S	<b><math>0,98 \pm 0,01</math></b>	<b><math>0,98 \pm 0,01</math></b>	<b><math>0,98 \pm 0,01</math></b>
	E	$0,89 \pm 0,05$	$0,84 \pm 0,04$	<b><math>0,92 \pm 0,04</math></b>
	V	$0,93 \pm 0,06$	$0,92 \pm 0,02$	<b><math>0,97 \pm 0,04</math></b>
	D	<b><math>0,79 \pm 0,04</math></b>	$0,68 \pm 0,10$	$0,78 \pm 0,06$
	GC	$0,90 \pm 0,06$	$0,86 \pm 0,02$	<b><math>0,95 \pm 0,02</math></b>
	R	$0,96 \pm 0,03$	$0,86 \pm 0,07$	<b><math>0,97 \pm 0,04</math></b>
$F_1$ -mesure	S	$0,95 \pm 0,02$	$0,94 \pm 0,01$	<b><math>0,97 \pm 0,01</math></b>
	E	$0,92 \pm 0,03$	$0,88 \pm 0,03$	<b><math>0,94 \pm 0,03</math></b>
	V	$0,96 \pm 0,04$	$0,95 \pm 0,01$	<b><math>0,98 \pm 0,02</math></b>
	D	$0,82 \pm 0,04$	$0,75 \pm 0,03$	<b><math>0,85 \pm 0,06</math></b>
	GC	$0,93 \pm 0,03$	$0,90 \pm 0,01$	<b><math>0,96 \pm 0,02</math></b>
	R	$0,93 \pm 0,04$	$0,86 \pm 0,04$	<b><math>0,95 \pm 0,04</math></b>

### Résultats comparatifs

Afin d'évaluer notre méthode, nous comparons ses résultats avec ceux obtenus par des méthodes basées sur l'extraction ciblée de caractéristiques développées à partir de notre expertise pour permettre une comparaison pertinente. Un classifieur SVM a été entraîné avec les 72 caractéristiques de couleur et de texture (cités dans la section 3.4.1) extraites de chaque face de la pomme de terre. Une validation croisée de type  $k$ -folds ( $k = 5$ ) a été employé pour estimer les performances. Les expériences sont effectuées à partir de l'ensemble de données déséquilibrées original ( $SVM_{Déséquilibré}$ ), mais également avec deux versions équilibrées et obtenues comme suit :

Pour obtenir la première version équilibrée (nommée  $SVM_{Équilibré}$ ), nous appliquons une technique de sous-échantillonnage qui nous donne 174 images par classe, soit le nombre d'images d'entraînement de la classe minoritaire (rhizoctone). De cette façon, chaque classe est représentée par le même nombre d'observations, ce qui donne une base de données équilibrée pour former le SVM.

De plus, pour éviter de réduire le nombre d'échantillons d'apprentissage, nous proposons une deuxième version équilibrée dans lequel nous ajustons les poids des échantillons de chaque classe lors de l'entraînement du SVM. Cette version est désignée dans ce travail par  $SVM_{Pondéré}$ . En nous inspirant des travaux de Huang et al. [184], nous ajustons le

TABLEAU 3.6 – Matrices de confusion obtenues après l’entraînement d’AlexNet, de VGG-16 et de GoogLeNet. Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone. Les valeurs sont en %.

		Classe réelle(%)					
		S	E	V	D	GC	R
Classe préd.(%)	AlexNet						
	S	<b>97,5</b>	6,6	5,9	20,0	7,9	2,9
	E	0,8	<b>89,4</b>	0,1	0,2	0,5	0,0
	V	0,0	0,6	<b>93,3</b>	0,2	0,0	0,0
	D	1,2	0,7	0,7	<b>78,8</b>	0,2	0,0
	GC	0,4	2,2	0,0	0,7	<b>90,2</b>	1,2
	R	0,1	0,4	0,0	0,0	1,2	<b>96,0</b>
Classe préd.(%)	VGG-16						
	S	<b>97,6</b>	11,5	7,0	28,8	9,6	8,1
	E	0,6	<b>83,7</b>	0,1	0,7	1,1	2,3
	V	0,3	0,4	<b>91,9</b>	0,7	0,1	0,0
	D	0,7	1,3	0,7	<b>68,4</b>	1,2	0,0
	GC	0,7	2,4	0,2	1,2	<b>86,2</b>	3,5
	R	0,1	0,6	0,1	0,2	1,7	<b>86,2</b>
Classe préd.(%)	GoogLeNet						
	S	<b>98,1</b>	6,3	2,8	20,5	3,7	1,1
	E	0,6	<b>92,4</b>	0,0	0,2	0,3	0,6
	V	0,2	0,1	<b>96,9</b>	0,5	0,1	0,0
	D	0,6	0,0	0,2	<b>78,4</b>	0,2	0,0
	GC	0,4	1,0	0,1	0,5	<b>94,5</b>	1,1
	R	0,1	0,1	0,0	0,0	1,1	<b>97,1</b>

paramètre de pénalité de l’erreur de classification du SVM ( $C$ ) en l’affectant de facteurs de pondération pour chaque classe. De cette manière, le problème d’optimisation quadratique sous contraintes linéaires présenté dans l’équation 3.27 peut être ré-écrit comme suit :

$$\min_{w,b,\xi} \left( \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n a_i \xi_i \right), \quad C \geq 0 \quad (3.43)$$

sous les contraintes

$$y_i (w^T x_i + b) \geq 1 - \xi_i, \quad \forall i = 1, \dots, n$$

$$\xi_i \geq 0, \quad \forall i = 1, \dots, n$$

où  $a_i$  est le facteur de pondération affectant l’observation  $i$ . L’objectif de cette pondération est de compenser les effets négatifs de la sous représentation des classes minoritaires afin que les contributions de chaque classe soient équilibrées.

Les résultats comparatifs sont présentés dans le tableau 3.7. Nous pouvons observer que les méthodes fondées sur l'apprentissage de caractéristiques ont surpassée largement les classifieurs conventionnels entraînés à partir de caractéristiques prédéfinies. La moyenne de la  $F_1$ -mesure obtenue avec le classifieur  $SVM_{Déséquilibré}$  est de 0,78 contre 0,94 obtenue avec GoogLeNet.

TABLEAU 3.7 – Comparaison de la moyenne de la précision, du rappel et de la  $F_1$ -mesure obtenues avec des méthodes d'apprentissage de caractéristiques et d'extraction ciblée de caractéristiques.

Méthode	Précision moyenne	Rappel moyenne	$F_1$ -mesure moyenne
AlexNet	0,93	0,91	0,92
VGG-16	0,91	0,86	0,88
GoogLeNet	<b>0,95</b>	<b>0,93</b>	<b>0,94</b>
$SVM_{Équilibré}$	0,66	0,76	0,69
$SVM_{Pondéré}$	0,75	0,78	0,76
$SVM_{Déséquilibré}$	0,83	0,74	0,78

Les résultats détaillés obtenus à partir des approches reposant sur l'extraction ciblée de caractéristiques sont présentés dans le tableau 3.8. Comme prévu, la performance de la classe saine a été affectée lorsque l'ensemble de données équilibré a été utilisé. En effet, la bonne détection de cette classe évolue de 0,94 avec  $SVM_{Déséquilibré}$  à 0,71 et 0,84 avec  $SVM_{Équilibré}$  et  $SVM_{Pondéré}$  respectivement. Cela peut s'expliquer par le fait que la classe saine est la classe majoritaire dans l'ensemble de données original.

Les résultats les plus faibles ont été obtenus avec la technique de sous-échantillonnage  $SVM_{Équilibré}$  ( $F_1$ -mesure moyenne de 0,69). Dans ce contexte, plusieurs observations ne sont pas utilisées pour entraîner le classifieur, ce qui a un impact négatif sur la performance. Tout comme les résultats obtenus avec le CNN, la  $F_1$ -mesure moyenne la plus faible obtenue par toutes les méthodes est celle de la dartrose.

Afin de mieux comprendre la confusion existant entre les différentes classes, nous présentons les matrices de confusion dans le tableau 3.9. Comme attendu, nous pouvons observer que la classe dartrose a été souvent confondue avec la classe saine en raison de leur similitude. De manière globale, nous pouvons observer que les meilleures performances ont été obtenues avec le  $SVM_{Pondéré}$  où la contribution de chaque classe a été équilibrée sans avoir à réduire le nombre d'échantillons. En effet, avec cette méthode, toutes les classes obtiennent un taux de bonne détection plus équilibré, ce qui n'est pas le cas du  $SVM_{Déséquilibré}$  où le taux de bonne détection de la classe majoritaire (classe saine) est bien plus élevé que celui des autres classes.

En raison des résultats pertinents obtenus avec l'architecture GoogLeNet, nous avons décidé d'utiliser ce réseau comme première brique de notre traitement. En effet, GoogLeNet est utilisé pour classer chaque face de la pomme de terre, et uniquement les images endommagées et vertes passeront à la phase de localisation.

TABLEAU 3.8 – Moyenne et écart-type de la précision, le rappel et la  $F_1$ -mesure par classe obtenus par les méthodes basées sur l'extraction ciblée de caractéristiques. Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

Critères	Classe	SVM <sub>Équilibré</sub>	SVM <sub>Pondéré</sub>	SVM <sub>Déséquilibré</sub>
Précision	S	<b>0,91 ± 0,01</b>	0,90 ± 0,01	0,85 ± 0,01
	E	0,46 ± 0,05	0,57 ± 0,02	<b>0,74 ± 0,06</b>
	V	0,74 ± 0,02	0,89 ± 0,02	<b>0,93 ± 0,01</b>
	D	0,40 ± 0,04	0,51 ± 0,03	<b>0,69 ± 0,08</b>
	GC	0,83 ± 0,03	0,83 ± 0,02	<b>0,86 ± 0,03</b>
	R	0,61 ± 0,06	0,81 ± 0,07	<b>0,90 ± 0,07</b>
Rappel	S	0,71 ± 0,01	0,84 ± 0,02	<b>0,94 ± 0,01</b>
	E	<b>0,72 ± 0,03</b>	0,70 ± 0,02	0,51 ± 0,08
	V	0,88 ± 0,03	<b>0,89 ± 0,03</b>	<b>0,89 ± 0,02</b>
	D	<b>0,63 ± 0,03</b>	0,61 ± 0,03	0,48 ± 0,06
	GC	0,77 ± 0,01	<b>0,85 ± 0,04</b>	<b>0,85 ± 0,04</b>
	R	<b>0,86 ± 0,02</b>	0,77 ± 0,06	0,78 ± 0,08
$F_1$ -mesure	S	0,80 ± 0,01	0,87 ± 0,01	<b>0,89 ± 0,01</b>
	E	0,56 ± 0,03	<b>0,63 ± 0,01</b>	0,60 ± 0,06
	V	0,81 ± 0,01	0,89 ± 0,02	<b>0,91 ± 0,01</b>
	D	0,49 ± 0,03	0,55 ± 0,03	<b>0,56 ± 0,07</b>
	GC	0,80 ± 0,01	0,84 ± 0,01	<b>0,85 ± 0,02</b>
	R	0,71 ± 0,04	0,79 ± 0,02	<b>0,83 ± 0,04</b>

### 3.5.3 Résultats de la localisation de défauts : endommagés et verts

L'architecture de l'auto-encodeur utilisé pour l'extraction de caractéristiques est présentée dans la figure 3.17. Le réseau prend en entrée des patches de taille  $16 \times 16$  en RVB, soit  $(16 \times 16 \times 3) = 768$  attributs. La couche cachée est composée de 50 neurones et le réseau est entièrement connecté. L'algorithme d'optimisation de la descente de gradient conjugué [185] a été employé pour l'entraînement de l'AE. L'erreur quadratique moyenne (MSE) a été utilisée comme fonction de perte et l'hyper-paramètre  $\lambda$  de *weight decay* a été fixé à  $3 \times 10^{-6}$ . Le nombre maximal d'*epochs* a été fixé à 1000. La fonction sigmoïde a été employée comme fonction d'activation dans l'encodeur et le décodeur. Dans la figure

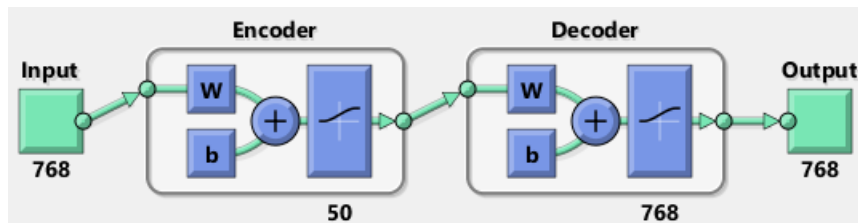


FIGURE 3.17 – Architecture de l'auto-encodeur proposé.

TABLEAU 3.9 – Matrices de confusion obtenues à partir des méthodes basées sur l'extraction ciblée de caractéristiques. Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone. Les valeurs sont en %.

		Classe réelle(%)					
		S	E	V	D	GC	R
Classe préd.(%)	SVM <sub>Équilibré</sub>						
	S	<b>71,4</b>	11,2	6,7	15,8	6,2	3,5
	E	12,6	<b>71,6</b>	2,6	8,4	4,1	2,3
	V	5,7	3,9	<b>88,5</b>	5,1	0,9	0,6
	D	7,6	8,2	1,7	<b>62,6</b>	5,6	2,3
	GC	2,3	3,5	0,4	4,6	<b>77,4</b>	5,7
	R	0,4	1,6	0,1	3,5	5,8	<b>85,6</b>
Classe préd.(%)	SVM <sub>Pondéré</sub>						
	S	<b>84,2</b>	17,2	7,1	23,0	6,2	1,2
	E	7,5	<b>70,1</b>	1,6	6,3	3,7	3,5
	V	1,8	2,1	<b>88,9</b>	2,8	0,5	0,0
	D	4,5	5,2	2,1	<b>60,9</b>	3,3	1,1
	GC	1,8	4,5	0,2	6,3	<b>84,9</b>	17,2
	R	0,2	0,9	0,1	0,7	1,4	<b>77,0</b>
Classe préd.(%)	SVM <sub>Déséquilibré</sub>						
	S	<b>94,5</b>	40,1	10,2	39,8	10,2	5,2
	E	1,9	<b>50,5</b>	0,5	3,5	2,4	3,4
	V	0,9	1,6	<b>88,6</b>	2,3	0,3	0,0
	D	1,4	3,0	0,6	<b>47,9</b>	1,4	0,6
	GC	1,2	4,2	0,1	6,5	<b>84,9</b>	13,2
	R	0,1	0,6	0,0	0,0	0,8	<b>77,6</b>

3.18 nous montrons la reconstruction de patches effectuée par l'AE. Comme nous pouvons l'observer, la reconstruction est réalisée avec succès, ce qui nous permet de conclure que le réseau est bien entraîné et que les 50 attributs extraits pour décrire chaque patch constituent une représentation robuste et pertinente avec une réduction significative de la dimension (de 768 à 50).

Afin de détecter les patches endommagés et verts, nous avons entraîné deux classifieurs 2C-SVM<sub>Dét</sub>, un pour chaque tâche de classification. En effet, un premier 2C-SVM<sub>DétE</sub> est utilisé pour dissocier les patches endommagés des patches non endommagés et un second 2C-SVM<sub>DétV</sub> pour séparer les patches verts des patch non verts. Ces classifieurs prennent en entrée les représentations extraites par l'auto-encodeur. La validation croisée avec 5 *folds* a également été appliquée pour estimer les performances. De plus, une recherche par grille a été utilisée afin de régler les hyper-paramètres, c'est-à-dire choisir la combinaison du paramètre du noyau gaussien  $\sigma$  et du poids de pénalisation  $C$ .

Nous comparons les résultats entre un SVM binaire (2C-SVM<sub>DétE</sub> et 2C-SVM<sub>DétV</sub>) et un SVM mono-classe (OC-SVM<sub>Dét</sub>). L'intérêt principal d'utiliser le OC-SVM réside



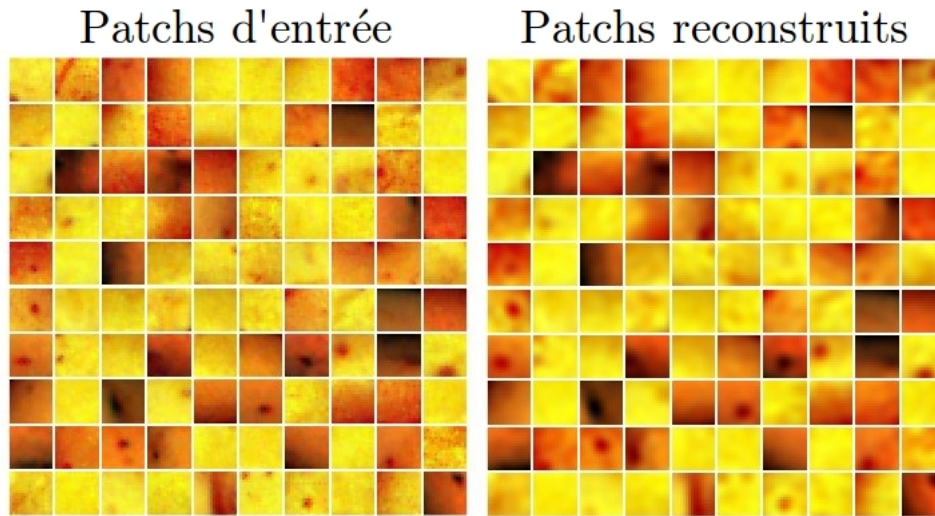


FIGURE 3.18 – Comparaison des patches de l’ensemble de test reconstruits par l’auto-encodeur.

dans la facilité d’obtenir uniquement des patches normaux sans avoir besoin de créer manuellement des étiquettes par patches. En effet, les patches normaux peuvent être extraits directement des images étiquetées comme étant saines, qui ne doivent théoriquement pas contenir de patches avec défauts. Pour le cas des classifieurs  $2C-SVM_{DétE}$  et  $2C-SVM_{DétV}$ , nous utilisons des patches étiquetés comme endommagés et non endommagés, et verts et non verts pour faire l’entraînement de chaque classifieur. Le tableau 3.10 montre les résultats sur l’ensemble de données endommagées et le tableau 3.11 montre, quant à lui, les résultats sur les données vertes.

Comme prévu, nous avons constaté une supériorité des résultats lors de l’utilisation des

TABLEAU 3.10 – Résultats de la classification de patches endommagés versus non-endommagés. TFA = Taux de Fausses Alarmes et TND = Taux de Non Détection.

	TFA(%)	TND(%)
OC-SVM <sub>Dét</sub>	4,23	27,66
2C-SVM <sub>DétE</sub>	<b>4,19</b>	<b>14,46</b>

TABLEAU 3.11 – Résultats de la classification des patches verts versus non-verts. TFA = Taux de Fausses Alarmes et TND = Taux de Non Détection.

	TFA(%)	TND(%)
OC-SVM <sub>Dét</sub>	<b>4,91</b>	39,11
2C-SVM <sub>DétV</sub>	5,53	<b>28,11</b>

classifieurs supervisés bi-classes  $2C-SVM_{DétE}$  et  $2C-SVM_{DétV}$ . On constate que pour un taux de fausse alarme (TFA) similaire, le 2C-SVM permet une diminution considérable du taux de non détection (TND) par rapport au OC-SVM pour les deux tâches de classification.

Nous pouvons observer que le taux de non détection des patchs verts est un peu plus élevé que celui des patchs endommagés. Cela peut s'expliquer par le fait que les taches causées par le verdissement ne sont pas homogènes et la frontière entre la surface affectée par le défaut et la surface saine est difficile à définir, même pour un humain. D'après les résultats obtenus, nous pouvons confirmer que le taux de non détection, ainsi que le taux de fausses alarmes, des patchs endommagés et verts est satisfaisant. Cette conclusion se fonde sur les résultats de classification par gravité présentés dans la section suivante, qui s'appuie sur les résultats de localisation. Toutefois, une analyse qualitative des résultats conforte cette conclusion (voir figures 3.19 et 3.20).

### 3.5.4 Résultats de la classification de défauts par gravité

L'étape suivante consiste à faire une synthèse des résultats obtenus avec les patchs détectés sur les faces des pommes de terre en notant l'importance du défaut. En effet, après la classification de défauts sur les patchs, il est possible de localiser les zones affectées et donc définir leurs étendues. Les figures 3.19 et 3.20 montrent un exemple de la localisation effectuée par l'AE+2C-SVM<sub>DétE</sub> et l'AE+2C-SVM<sub>DétV</sub> pour les pommes de terre endommagées et vertes respectivement. Comme on peut le voir, la localisation a été effectuée avec succès pour les deux défauts. Nous pouvons donc conclure qu'il est pertinent d'utiliser les résultats de la localisation pour catégoriser les défauts par gravité. À cet effet, les informations relatives à la localisation ont été utilisées comme entrée de deux classifieurs 2C-SVM<sub>GravE</sub> et 2C-SVM<sub>GravV</sub> afin de définir la gravité des endommagements et verdissements respectivement.

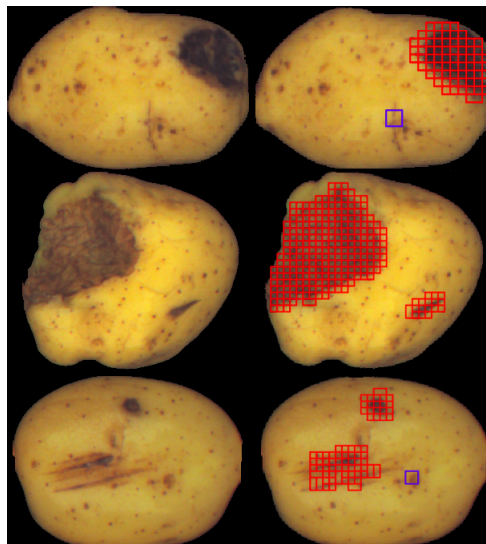


FIGURE 3.19 – Résultats obtenus lors de la localisation au moyen de l'AE+2C-SVM<sub>DétE</sub> pour les pommes de terre endommagées. Le patch bleu représente une irrégularité isolée où aucun patch adjacent n'a été détecté. Dans ce cas, le patch bleu est ignoré afin de minimiser les fausses alarmes et d'éviter la détection des petites imperfections non pertinentes.

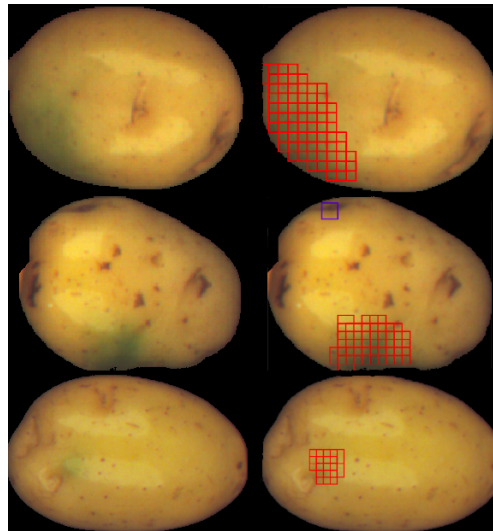


FIGURE 3.20 – Résultats obtenus lors de la localisation au moyen de l’AE+2C-SVM<sub>DétV</sub> pour les pommes de terre vertes. Le patch bleu représente une irrégularité isolée où aucun patch adjacent n’a été détecté. Dans ce cas, le patch bleu est ignoré afin de minimiser les fausses alarmes et d’éviter la détection des petites imperfections non pertinentes.

Jusqu’à présent nous avons réalisé une classification par face (ou par côté) de la pomme de terre. Néanmoins, comme nous l’avons expliqué dans la section 3.4.3, les étiquettes relatives à la gravité sont disponibles uniquement au niveau de la pomme de terre (4 faces, 1 étiquette). Par conséquent, pour réaliser la classification par gravité nous avons retenu pour chaque défaut et chaque pomme de terre uniquement les résultats de la face où le défaut localisé occupe la surface relative la plus importante. Enfin, les pommes de terre sont classées dans les catégories suivantes : Endommagée Légère (EL), Endommagée Grave (EG), Verte Légère (VL) ou Verte Grave (VG). Comme il a été précisé dans la section 3.4.3, le nombre de patches détectés par l’AE+2C-SVM<sub>DétE</sub> ou l’AE+2C-SVM<sub>DétV</sub>, le pourcentage de la surface affectée, ainsi que la somme des scores de sortie du classifieur 2C-SVM<sub>DétE</sub> ou 2C-SVM<sub>DétV</sub> pour les patches détectés, sont utilisés comme attributs d’entrée de deux classifieurs 2C-SVM<sub>GravE</sub> et 2C-SVM<sub>GravV</sub>, un pour chaque classe : endommagée et verte. La validation croisée et la recherche par grille ont été appliquées pour sélectionner les hyper-paramètres des classifieurs.

Le tableau 3.12 et le tableau 3.13 montrent les résultats de la classification par gravité des pommes de terre endommagées et vertes respectivement. Afin de mesurer l’utilité de l’étape de classification du CNN, nous comparons les résultats obtenus avec et sans le CNN comme première étape de classification. Lorsque le CNN n’est pas utilisé, une image est classée saine si moins de deux patches defectueux sont détectées. Dans le cas contraire, elle est classée comme defectueuse et elle sera ensuite classée selon sa gravité.

Les meilleurs résultats globaux ont été obtenus avec l’utilisation du CNN comme première étape de classification (notamment pour les pommes de terre vertes), ce qui a permis de réduire le nombre de pommes de terre saines classées comme endommagées ou vertes, c’est-à-dire que nous avons diminué les fausses alarmes. Un autre avantage d’une

TABLEAU 3.12 – Résultats obtenus lors de la classification par gravité de pommes de terre endommagées en utilisant les deux méthodes :  $AE+2C-SVM_{DétE}+2C-SVM_{GravE}$  («sans CNN») et  $GoogLeNet+AE+2C-SVM_{DétE}+2C-SVM_{GravE}$  («avec CNN»). Les classes sont : Saine (S), Endommagée Légère (EL) et Endommagée Grave (EG).

		sans CNN	avec CNN
Précision	S	0,96	<b>0,97</b>
	EL	0,80	<b>0,94</b>
	EG	<b>0,95</b>	0,94
Rappel	S	0,92	<b>0,99</b>
	EL	0,88	<b>0,91</b>
	EG	<b>0,93</b>	0,90
F <sub>1</sub> -mesure	S	0,94	<b>0,98</b>
	EL	0,84	<b>0,92</b>
	EG	<b>0,94</b>	0,92

TABLEAU 3.13 – Résultats obtenus lors de la classification par gravité de pommes de terre vertes en utilisant les deux méthodes :  $AE+2C-SVM_{DétV}+2C-SVM_{GravV}$  («sans CNN») et  $GoogLeNet+AE+2C-SVM_{DétV}+2C-SVM_{GravV}$  («avec CNN»). Les classes sont : Saine (S), Verte Légère (VL) et Verte Grave (VG).

		sans CNN	avec CNN
Précision	S	0,99	0,99
	VL	0,37	<b>0,86</b>
	VG	0,88	<b>0,96</b>
Rappel	S	0,84	<b>1</b>
	VL	0,62	<b>0,85</b>
	VG	<b>0,98</b>	0,95
F <sub>1</sub> -mesure	S	0,91	<b>0,99</b>
	VL	0,47	<b>0,85</b>
	VG	0,93	<b>0,95</b>

première étape de classification par CNN est la réduction du temps de calcul. La prédiction du CNN est deux fois plus rapide que la méthode de localisation des défauts par patch avec l' $AE+2C-SVM_{DétE}$  ou l' $AE+2C-SVM_{DétV}$ . Par conséquent, l'analyse des patches issus uniquement des images classées endommagées ou vertes durant l'étape de classification initiale (classification par le CNN) réduit significativement le temps de traitement.

Afin d'analyser les confusions qui se sont produites lors du processus de classification, les matrices de confusion de chacune des deux tâches de classification, sans et avec utilisation du CNN, sont présentées dans les tableaux 3.14 et 3.15. Comme le montre le premier tableau, seulement 0,91% des pommes de terre gravement endommagées (1 observation) ont été classées comme saine. Cette erreur de classification s'est produite avec une pomme de terre sectionnée, dont la partie endommagée n'était pas foncée mais jaune clair, ce qui peut expliquer la confusion (voir figure 3.21). On constate une large réduction du taux de

fausses alarmes grâce à l'utilisation du CNN : de 7,55% à 0,69% et de 15,78% à 0% pour les images de pommes de terre endommagées et vertes respectivement.

TABLEAU 3.14 – Matrices de confusion correspondant à la classification des images de pommes de terre en Saine (S), Endommagée Légère (EL) et Endommagée Grave (EG). Deux méthodes sont comparées : AE+2C-SVM<sub>DétE</sub>+2C-SVM<sub>GravE</sub> («sans CNN») et GoogLeNet+AE+2C-SVM<sub>DétE</sub>+2C-SVM<sub>GravE</sub> («avec CNN»)

		Classe réelle(%)		
		S	EL	EG
Classe préd.(%)	sans CNN			
	S	<b>92,3</b>	10,5	0
	EL	7,5	<b>87,8</b>	7,3
	EG	0,2	1,7	<b>92,7</b>
Classe préd.(%)	avec CNN			
	S	<b>99,3</b>	6,7	0,9
	EL	0,7	<b>90,8</b>	9,1
	EG	0	2,5	<b>90</b>

Nous concluons d'après ces résultats que les caractéristiques extraites à partir de l'étape de localisation présentée dans la section 3.5.3 sont pertinentes pour la classification par gravité des images de pomme de terre endommagées et vertes.

TABLEAU 3.15 – Matrices de confusion correspondant à la classification des images de pommes de terre en Saine (S), Verte Légère (VL) et Verte Grave (VG). Deux méthodes sont comparées : AE+2C-SVM<sub>DétV</sub>+2C-SVM<sub>GravV</sub> («sans CNN») et GoogLeNet+AE+2C-SVM<sub>DétV</sub>+2C-SVM<sub>GravV</sub> («avec CNN»)

		Classe réelle(%)		
		S	VL	VG
Classe préd.(%)	sans CNN			
	S	<b>83,70</b>	4,30	0
	VL	15,78	<b>62,37</b>	2,28
	VG	0,51	33,33	<b>97,72</b>
Classe préd.(%)	avec CNN			
	S	<b>100</b>	3,23	0
	VL	0	<b>84,95</b>	4,94
	VG	0	11,83	<b>95,06</b>

### 3.5.5 Résultats de la classification multi-classes multi-étiquettes

Pour les résultats finaux, un ensemble de données de test multi-classes et multi-étiquettes de 722 tubercules a été utilisé, voir exemple de la figure 3.22. Dans cette phase, les quatre étiquettes de sortie obtenues lors des étapes précédentes, une pour chaque face, sont considérées pour caractériser la pomme de terre de manière globale. Les résultats



FIGURE 3.21 – Image d’une pomme de terre endommagée grave détectée comme saine par la méthode  $\text{GoogLeNet}+\text{AE}+2\text{C-SVM}_{\text{DétE}}+2\text{C-SVM}_{\text{GravE}}$ .



FIGURE 3.22 – Exemple d’un pomme de terre avec deux faces classe verte et deux faces classe dartrose.

finaux obtenus en utilisant la méthode  $\text{GoogLeNet}+\text{AE}+2\text{C-SVM}_{\text{Dét}}+2\text{C-SVM}_{\text{Grav}}$ <sup>2</sup> sont présentés dans le tableau 3.16. De plus, nous montrons dans le tableau 3.17 les résultats obtenus sans réaliser la distinction des défauts par gravité afin d’analyser uniquement la performance du réseau de neurones convolutifs.

Nous pouvons constater que malgré la similitude entre certaines classes et la grande variabilité au sein d’une même classe, l’ensemble du système fonctionne de manière satisfaisante. Les meilleures performances ont été atteintes sur la classe saine avec une précision de 0,98 et donc un nombre de fausses alarmes extrêmement faible. Ceci est particulièrement important étant donné que cette classe est la plus présente dans les cas réels et que la confusion entre une pomme de terre saine et une pomme de terre abîmée ne se produit pas souvent lorsque l’analyse est effectuée par un humain. La classe dartrose, quant à elle, a obtenu les résultats de détection les plus bas (0,82) en raison de la confusion entre les pommes de terre affectées par cette maladie et les pommes de terres saines, comme cela a été expliqué dans la section 3.5.2. En effet, la dartrose est difficilement détectable en

2. Afin de simplifier la notation, nous utilisons  $2\text{C-SVM}_{\text{Dét}}$  pour désigner les classifieurs  $2\text{C-SVM}_{\text{DétE}}$  et  $2\text{C-SVM}_{\text{DétV}}$  et  $2\text{C-SVM}_{\text{Grav}}$  pour les classifieurs  $2\text{C-SVM}_{\text{GravE}}$  et  $2\text{C-SVM}_{\text{GravV}}$ .

TABLEAU 3.16 – Résultats obtenus sur la base de données de test multi-classes multi-étiquettes en utilisant la méthode GoogLeNet+AE+2C-SVM<sub>Dét</sub>+2C-SVM<sub>Grav</sub>. Les classes sont : S=Saine, EL=Endommagée Légère, EG=Endommagée Grave, VL=Verte Légère, VG=Verte Grave, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

	Précision	Rappel	F <sub>1</sub> -mesure
S	0,98	0,98	0,98
EL	0,90	0,94	0,92
EG	0,86	0,88	0,87
VL	0,88	0,91	0,89
VG	0,98	0,95	0,96
D	0,92	0,82	0,87
GC	0,95	0,87	0,91
R	0,88	0,95	0,91
Moy	0,92	0,91	0,91

TABLEAU 3.17 – Résultats obtenus en utilisant le réseau GoogLeNet pour classer les images appartenant à la base de données de test multi-classes multi-étiquettes. Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

	Précision	Rappel	F <sub>1</sub> -mesure
S	0,98	0,98	0,98
E	0,93	0,97	0,95
V	1	0,99	0,99
D	0,92	0,82	0,87
GC	0,95	0,87	0,91
R	0,88	0,95	0,91
Moy	0,94	0,93	0,94

utilisant seulement des attributs visuels. Compte tenu de cet aspect, les résultats obtenus sur cette classe sont satisfaisants.

La classification des défauts par gravité (tableau 3.16) est très performante : la F<sub>1</sub>-mesure la plus basse obtenue est de 0,87 pour la classe endommagée grave. Néanmoins, les résultats sont bien meilleurs lorsque la distinction en fonction de la gravité est omise (voir tableau 3.17). Ceci est compréhensible en considérant la difficulté de définir clairement la limite entre les défauts légers et les défauts graves, même pour un expert humain.

### 3.6 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle méthode de contrôle qualité des pommes de terre reposant sur l'apprentissage profond, ainsi que sur des techniques de classification conventionnelles. La méthode proposée permet non seulement de classer les faces de pommes de terre en 6 classes différentes (saine, endommagée, verte, dartrose,

gale commune et rhizoctone), mais également de localiser de manière précise les défauts (verdissements et endommagements) et de les classer ensuite en fonction de leur gravité.

Les résultats expérimentaux ont montré la précision de notre méthode pour ce qui est de la classification des faces de pommes de terre. Les résultats ont également mis en avant la supériorité du CNN pour cette tâche comparé aux approches conventionnelles basées sur une extraction ciblée de caractéristiques. Dans l'étape de localisation des défauts affectant les pommes de terre endommagées et vertes, nous avons comparé deux approches fondées respectivement sur un apprentissage supervisé et non supervisé. Nous avons sans surprise observé la supériorité de l'approche supervisée. Bien que cette dernière implique la création d'une base de données annotée par patches, le gain de performance justifie dans ce cas l'investissement que nécessite la constitution de cette base de données.

Les bonnes performances de notre méthode ainsi que sa complexité calculatoire réduite, ont permis son intégration dans un environnement industriel. Bien que dans la littérature d'autres travaux ont été proposés pour la classification des pommes de terre, ces travaux utilisent généralement des bases de données limitées en termes de nombre d'exemples et/ou de nombre de défauts à classer (voir tableau 3.18), ce qui rend difficile une comparaison appropriée avec notre méthode.

TABLEAU 3.18 – Travaux de la littérature proposés pour la classification de pommes de terre catégorisés selon la taille de la base de données utilisée et selon le nombre de catégories à classer.

Auteur	Taille base de données	Nb. de catégories à classes
[18]	78	2
[21]	102	2
[89]	234	2
[22]	500	2
[23]	182	2
[138]	600	2
[63]	100	3

Les performances satisfaisantes de notre méthode ont permis son intégration dans un système industriel fonctionnel. Cependant, la nécessité d'utiliser une base de donnée étiquetée au niveau du patch pour l'apprentissage de la localisation constitue une limitation majeure. En effet, se défaire de l'apprentissage supervisé à l'aide de bases de données spécifiques pour la localisation constituerait une amélioration non négligeable. Afin de répondre à cette problématique, le chapitre suivant sera focalisé sur les méthodes de classification reposant sur un apprentissage faiblement supervisé.

La variabilité des conditions d'utilisation imposées par une exploitation au niveau industriel, nécessite aussi la mise en place de techniques permettant l'adaptabilité de la méthode à de nouvelles situations, telles que l'introduction d'une nouvelle variété de pommes de terre ou l'évolution des conditions d'acquisition des images. Ce point sera



abordé ultérieurement dans le chapitre 5.

# Chapitre 4

## Apprentissage faiblement supervisé pour la classification et la localisation de défauts externes

### Sommaire

---

<b>4.1</b>	<b>Introduction</b>	<b>94</b>
<b>4.2</b>	<b>Système proposé reposant sur l'apprentissage faiblement supervisé</b>	<b>95</b>
4.2.1	Classification des images par défauts	97
4.2.2	Localisation des défauts	98
4.2.3	Segmentation fine des défauts	100
4.2.4	Classification des défauts de pommes de terre par gravité	101
<b>4.3</b>	<b>Résultats expérimentaux</b>	<b>102</b>
4.3.1	Résultats de l'étape de classification	102
4.3.2	Résultats de localisation des défauts	103
4.3.3	Résultats de segmentation des défauts	104
4.3.4	Résultats de la classification des défauts par gravité	105
4.3.5	Évaluation globale du tubercule	108
<b>4.4</b>	<b>Réseau entièrement convolutif pour la segmentation sémantique d'images</b>	<b>111</b>
<b>4.5</b>	<b>Système proposé pour la segmentation sémantique d'images</b>	<b>115</b>
4.5.1	Étiquetage automatique au niveau du pixel d'une base de données	115
4.5.2	Segmentation sémantique d'images de pommes de terre	116
4.5.3	Classification de défauts par gravité	118
<b>4.6</b>	<b>Résultats expérimentaux</b>	<b>119</b>
<b>4.7</b>	<b>Conclusion</b>	<b>122</b>

---

## 4.1 Introduction

L'apprentissage de caractéristiques à l'aide de réseaux de neurones convolutifs nous a permis de mettre au point une méthode de classification et de localisation des défauts susceptibles d'affecter les pommes de terre. Cette méthode nous a notamment permis de vérifier la supériorité en terme de performance des réseaux de neurones convolutifs (CNNs) par rapport à des méthodes fondées sur une extraction ciblée de caractéristiques pour ce qui est de la classification des défauts de pommes de terre. Afin de réaliser la localisation des défauts, un système combinant un auto-encodeur avec un classifieur SVM a été utilisé. Dans cette approche, des patchs extraits des images ont été analysés et classés individuellement en fonction de la présence ou absence de défauts. Deux modèles de classification de patchs ont été explorés, un modèle non-supervisé (OC-SVM) et un modèle supervisé (2C-SVM), avec des résultats significativement meilleurs dans le cas de l'approche supervisée. En effet, la mise en œuvre de cette approche nécessite la constitution de bases de données étiquetées au niveau du patch pour accomplir la tâche de localisation des défauts.

Bien que la création de ce type de bases de données soit plus simple et moins chronophage que la création de bases étiquetées au niveau du pixel, elle demeure plus complexe et plus onéreuse que la création de bases de données étiquetées au niveau de l'image. En tenant compte de cet aspect, le développement d'une méthode donnant des résultats satisfaisants en utilisant un ensemble de données étiqueté uniquement au niveau de l'image est d'une importance fondamentale sur le plan opérationnel. Les méthodes reposant sur un apprentissage faiblement supervisé constituent une solution potentielle afin d'atteindre cet objectif et ainsi éliminer la dépendance de notre méthode vis-à-vis des bases de données étiquetées par patchs.

Comme il a été mentionné dans le chapitre 2, les approches fondées sur l'apprentissage faiblement supervisé se sont avérées efficaces dans de nombreuses tâches de localisation et de segmentation et ce en utilisant uniquement des bases de données étiquetées au niveau de l'image [146, 147, 148, 149, 150]. Dans ce type d'apprentissage, l'objectif est de créer des modèles prédictifs capables d'apprendre à partir d'une supervision minimale. Ce type de supervision est notamment nécessaire lorsque l'on est confrontés à un problème de segmentation d'images, mais que la base de données d'entraînement ne contient que des annotations au niveau de l'image.

Dans le contexte de notre problématique, l'apprentissage faiblement supervisé est utilisé afin d'obtenir une segmentation précise des défauts et ce en utilisant uniquement un ensemble de données étiquetés au niveau de l'image (voir exemple de la figure 4.1). Dans ce chapitre nous allons donc étudier des stratégies reposant sur l'apprentissage faiblement supervisé afin de segmenter les défauts sur les pommes de terre. Nous allons également examiner la possibilité d'appliquer des méthodes de segmentation d'images reposant sur l'apprentissage supervisé, en utilisant une base de données étiquetée par pixels construite de manière faiblement supervisée. Cette approche supervisée a deux principaux avantages : la phase d'apprentissage de la méthode est moins complexe compte tenu du fait qu'un seul réseau entraînable de bout en bout est utilisé. L'utilisation de ce réseau unique

permet un temps de traitement considérablement réduit durant la phase de prédiction, ce qui est fondamental dans notre contexte applicatif.

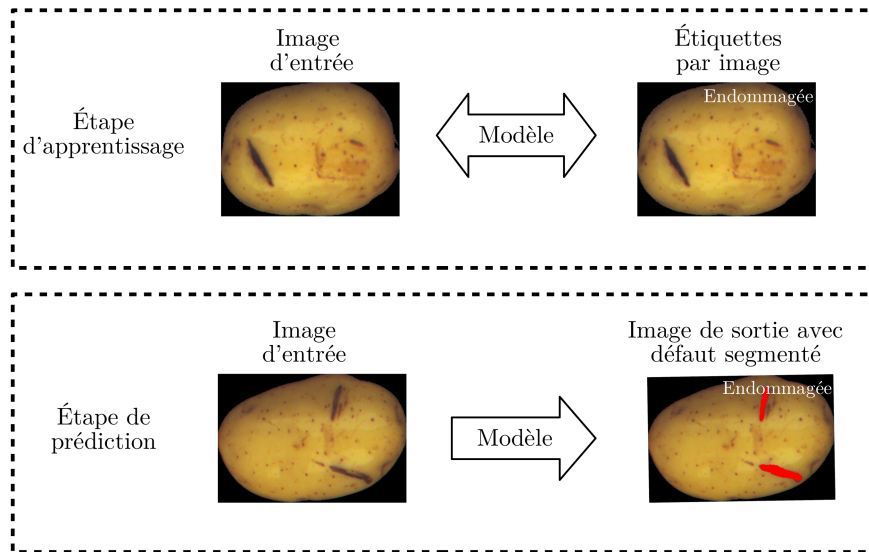


FIGURE 4.1 – Illustration des étapes d'apprentissage et de prédiction dans le contexte d'un apprentissage faiblement supervisé. L'objectif est d'obtenir une classification de défauts au niveau du pixel en utilisant une base de données d'entraînement uniquement étiquetée par image.

Le chapitre est structuré comme suit : dans un premier temps, nous allons présenter une méthode fondée sur l'apprentissage faiblement supervisé pour la classification, localisation et segmentation de défaut sur des pommes de terre. Les différentes étapes de cette méthode seront détaillées et les résultats de chaque étape seront présentés et discutés. Dans un second temps, nous allons présenter une deuxième méthode de segmentation d'images de pommes de terre reposant sur un apprentissage supervisé et tirant profit d'une base de données construite à l'aide de la méthode précédente. Les résultats obtenus à l'aide de cette méthode seront également exposés et comparés avec les méthodes proposées antérieurement.

## 4.2 Système proposé reposant sur l'apprentissage faiblement supervisé

Dans la figure 4.2 nous présentons le schéma global de la méthode proposée. Cette dernière se compose de quatre étapes :

1. Tout d'abord, une étape de classification d'images de pommes de terre est effectuée grâce à un CNN (figure 4.2 A).
2. Ensuite, les défauts sont localisés à l'aide d'une carte d'activation des défauts (appelé *Defect Activation Map* - DAM), où les régions clés de l'image qui ont conduit le réseau à prédire une classe particulière sont isolées. Bien entendu, cette étape de localisation ne s'applique pas aux images classées comme saines (figure 4.2 B).

3. Dans la troisième étape, une méthode de segmentation fine est appliquée afin d'obtenir une segmentation plus précise du défaut. Un classifieur OC-SVM est donc utilisé pour détecter tous les pixels anormaux appartenant aux régions abimées préalablement définis dans l'étape précédente (figure 4.2 C).
4. Enfin, dans la dernière étape, les résultats de la segmentation sont utilisés pour entraîner deux classifieurs 2C-SVMs dans le but de diviser par gravité les pommes de terre endommagées et vertes (figure 4.2 D).

Dans les sections suivantes, nous allons détailler les différentes étapes constitutives de notre méthode.

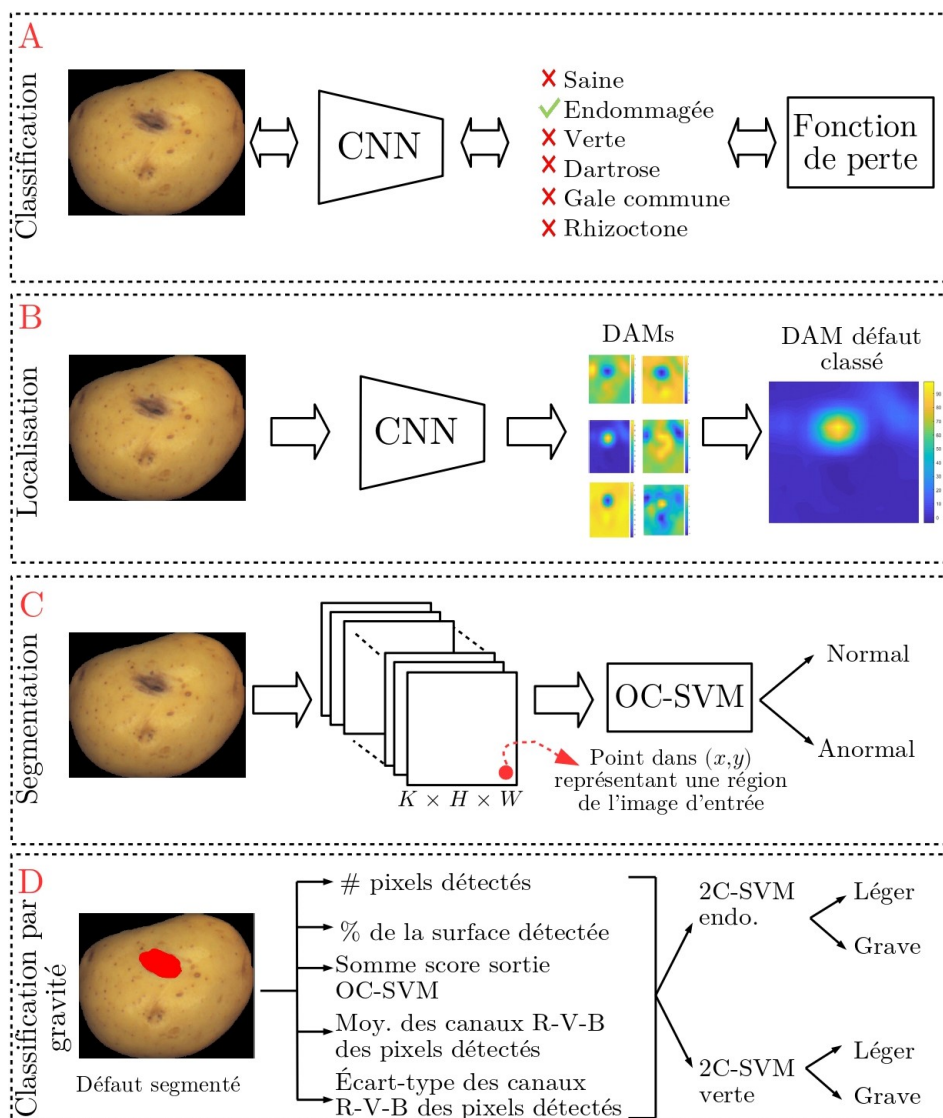


FIGURE 4.2 – Schéma de la méthode proposée fondée sur l'apprentissage faiblement supervisé.

### 4.2.1 Classification des images par défauts

La première phase de classification est similaire à celle présentée au chapitre 3, section 3.4.1 : les paramètres d'un CNN pré-entraîné sont adaptés à notre problème grâce à du *finetuning* et ce afin d'obtenir un réseau capable de classer chaque face de pomme de terre en six classes distinctes : saine, endommagée, verte, dartrose, gale commune et rizoctone. En plus d'évaluer l'architecture donnant les meilleurs résultats de classification dans le chapitre 3 (GoogLeNet), nous proposons également une nouvelle architecture. Ce nouveau réseau, nommé dans le contexte de cette thèse GNet-Modif, est dérivé du réseau GoogLeNet. Il consiste en une version moins profonde du réseau original. L'objectif de sa construction est l'obtention d'une meilleure résolution spatiale à la sortie de la dernière couche de convolution, la capacité de localisation du réseau devrait s'en trouver améliorée [149]. Afin de créer le réseau GNet-Modif, seules les couches en amont du sixième module d'inception ont été conservées (cf. figure 3.14). En effet, les 3 derniers modules d'inception ont été supprimés. Ainsi, la résolution de la dernière carte de caractéristiques obtenue par le réseau du GoogLeNet original a été doublée, en passant d'une taille de  $7 \times 7$  à  $14 \times 14$ . Après les 6 modules d'inception, une nouvelle couche de convolution avec 1024 filtres de taille  $3 \times 3$  et un *padding* de 1 et un pas de 1 a été ajoutée avant de remettre les deux dernières couches du GoogLeNet original (un *Global Average Pooling* (GAP) et la dernière couche entièrement connectée avec 6 neurones suivie d'une fonction d'activation *softmax*).

La base de données utilisée afin d'entraîner et de tester la méthode proposée est la même que celle décrite dans la section 3.3 du chapitre 3. Dans cette base de données, chaque image est affecté à une classe unique. De ce fait, l'utilisation de la fonction *softmax* dans la couche de classification finale est justifiée. Cependant, dans une application réelle, chaque face d'une pomme de terre peut contenir plus d'un défaut. Pour cette raison, nous proposons d'appliquer, dans l'étape de prédiction, en conjonction avec la fonction *softmax*, une fonction sigmoïde. Contrairement à la fonction *softmax*, la fonction sigmoïde est utilisée dans la couche de classification comme fonction d'activation lorsque les classes ne sont pas mutuellement exclusives, c'est-à-dire dans le cas d'un problème multi-label.

Étant donné une image  $I$ ,  $f_k \in \mathbb{R}^{W_l \times H_l}$  est sa  $k^{\text{ième}}$  carte de caractéristiques de la dernière couche  $l$  de convolution de taille  $H_l \times W_l$ , avec  $k = 1, 2, \dots, K$ , où  $K$  le nombre total de filtres de la couche de convolution  $l$ . Nous notons chaque carte de caractéristiques par  $f_k(x, y)$ , où  $(x, y)$  est chaque valeur qui compose la carte avec  $x = 1, \dots, W_l$  et  $y = 1, \dots, H_l$ . En appliquant un *Global Average Pooling* (GAP) à chaque carte de caractéristiques  $k$  on obtient  $F_k = \frac{1}{H_l \times W_l} \sum_{x,y} f_k(x, y)$ ,  $F_k \in \mathbb{R}$ . La sortie de la couche GAP est ensuite suivie par une dernière couche linéaire, entièrement connectée, qui génère un score  $S_m$  pour chaque classe  $m$  :

$$S_m = \sum_{k=1}^K w_k^m F_k \quad (4.1)$$

où  $w_k^m$  représente le poids appris pour la classe  $m$  dans l'unité  $k$ . Le score  $S_m$  de chaque

classe  $m$  est ensuite réévalué à l'aide de la fonction *softmax* :

$$O_m = \frac{e^{S_m}}{\sum_m e^{S_m}} \quad (4.2)$$

Dans la phase d'apprentissage, les valeurs de sortie pour chaque classe  $m$  obtenues à partir d'une observation quelconque sont utilisées pour calculer l'erreur de prédiction à l'aide de l'entropie croisée :

$$\mathcal{L}_{CE} = - \sum_{m=1}^M Y_m \log(O_m) \quad (4.3)$$

où  $M$  est le nombre de classes,  $Y_m$  est égal à 1 si l'observation d'entrée appartient à la classe  $m$  et égal à 0 sinon.

Dans la phase de prédiction, la fonction *softmax* de l'équation 4.2 est utilisée en conjonction avec la fonction sigmoïde définie par :

$$Q_m = \frac{1}{1 + e^{-S_m}} \quad (4.4)$$

Enfin, pour décider à quelle(s) classe(s) appartient l'image d'entrée, la règle de décision suivante est appliquée :

1. Première classe de sortie  $m^*$  tel que :

$$m^* = \arg \max_m O_m \quad (4.5)$$

2. Autre classe de sortie possible : tout  $m'$  tel que

$$Q_{m'} \geq h_{sigmoid} \quad (4.6)$$

où  $h_{sigmoid}$  est un seuil sélectionné de manière expérimental pour décider si l'image d'entrée  $I$  appartient à la classe  $m'$  ou pas. En définitive, cette règle de décision équivaut à définir un seuil sur le score de sortie  $S_m$ . Cependant, l'interprétation des sorties est plus facile en utilisant la fonction *softmax* et la fonction sigmoïde.

## 4.2.2 Localisation des défauts

La seconde étape consiste à localiser les défauts sur les faces des pommes de terre classées par le CNN selon la règle décrite dans la section précédente. Contrairement à la méthode présentée au chapitre 3, la localisation est effectuée pour tous les défauts et maladies de notre base de données (endommagements, verdissements, dartrose, gale commune et rhizoctone). Inspirés des travaux proposés par Zhou et al. [149], nous proposons de générer une carte d'activation des défauts (appelé *Defect Activation Map* - DAM), qui estime l'emplacement des défauts présents sur l'image classée. Comme nous pouvons l'observer dans la figure 4.3, la DAM est générée en utilisant une somme pondérée des cartes de caractéristiques de la dernière couche de convolution avec les poids appris dans la dernière

couche entièrement connectée. De manière générale, les filtres de convolution des couches moins profondes du CNN jouent le rôle de détecteurs de motifs simples. Puis, les filtres des couches de convolution les plus profondes se focalisent sur les motifs les plus complexes et discriminants, spécifiques à la tâche de classification ciblée. En remplaçant les couches entièrement connectées, qui sont couramment employées à la fin des réseaux, par une couche de GAP, nous pouvons isoler les motifs qui ont été les plus importants pour la prise de décision du CNN lors de la classification, tout en conservant les informations relatives à la localisation de ces motifs. Contrairement à d'autres méthodes faiblement supervisées que nous avons présentées dans l'état de l'art (chapitre 2) [152, 153, 150], la DAM est générée grâce à une seule propagation en avant (*forward pass*), ce qui rend cette méthode plus rapide et efficace durant la phase de prédiction.

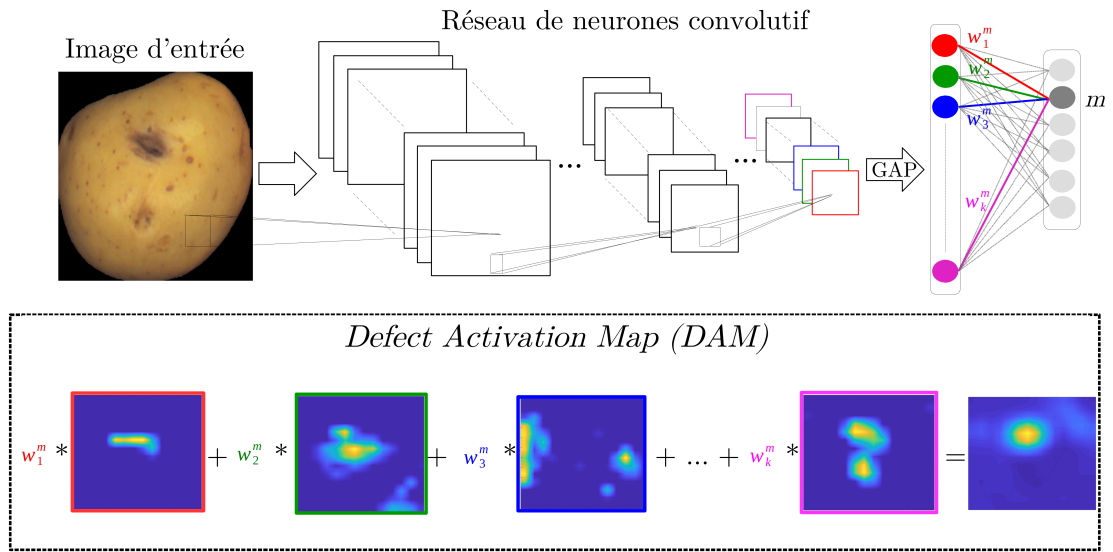


FIGURE 4.3 – Schéma de la génération de la Defect Activation Map dans le but de localiser le défaut classé par le CNN.

Étant donné une image  $I$ , le score de sortie pour chaque classe  $m$  ( $S_m$ ) est calculé par l'équation 4.1 en utilisant les poids  $w_k^m$  appris dans la dernière couche entièrement connectée. En effet, ces poids mesurent la contribution de chaque carte de caractéristiques  $f_k(x, y)$  à la classe de sortie  $m$ . Ainsi, la DAM est générée en appliquant une somme pondérée des cartes de caractéristiques comme suit :

$$DAM_m(x, y) = \sum_{k=1}^K w_k^m f_k(x, y) \quad (4.7)$$

Une interpolation bilinéaire est appliquée à chaque DAM afin de faire correspondre sa taille à celle de l'image d'entrée. De cette façon, nous pouvons identifier les régions clés qui ont permis au CNN de prédire une classe en particulier. Dans le but d'obtenir une localisation plus précise, une technique simple de seuillage est ensuite appliquée aux DAMs obtenues. Nous conservons les points  $(x, y)$  de la DAM qui vérifient la condition suivante :



$$DAM_m(x, y) \geq h_{DAM} \times \max DAM_m \quad (4.8)$$

où  $0 < h_{DAM} < 1$  est un seuil défini de manière expérimentale.

Dans la phase de prédiction, nous utilisons également les cartes d'activation des défauts pour décider si l'image appartient à plus d'une classe. Si une deuxième classe  $m'$  est prédite après application de la règle décrite par la relation 4.5, nous définissons si elle est retenue ou pas en fonction de la superposition des DAMs obtenues. En effet, nous calculons l'intersection entre la DAM de chaque classe de sortie et vérifions s'il existe une intersection entre elles. Si le chevauchement des DAMs est inférieur à 20%, les deux classes sont retenues. Sinon, seule la classe dominante est conservée. Cette stratégie est appliquée afin de réduire le nombre de fausses alarmes causées pas les fortes similitudes entre certains défauts ou maladies, telles que la gale commune et le rhizoctone.

Les résultats obtenus par la DAM nous donnent une estimation peu précise de la localisation spatiale des défauts, une étape de segmentation plus précise est donc nécessaire pour obtenir des résultats permettant de classer les défauts par gravité.

### 4.2.3 Segmentation fine des défauts

La troisième phase de notre approche vise à obtenir une segmentation précise des défauts classés. Pour ce faire, nous proposons une méthode qui n'utilise aucune information préalable sur l'emplacement ou l'étendue du défaut, mais tire parti des caractéristiques apprises par le CNN entraîné. En effet, une fois le CNN entraîné avec nos images de pommes de terre, il est utilisé comme extracteur de caractéristiques, de telle sorte que pour chaque image d'entrée  $I$ , nous obtenons à la sortie de la  $l^{\text{ème}}$  couche de convolution  $K$  cartes de caractéristiques de taille  $H_l \times W_l$ . Ainsi, une image d'entrée est représentée par  $(H_l \times W_l)$  vecteurs de caractéristiques de dimension  $K$  (voir figure 4.4). En fait, chacun de ces vecteurs est lié à une région spécifique de l'image d'entrée, de sorte qu'il est en mesure de décrire les motifs présents dans cette région.

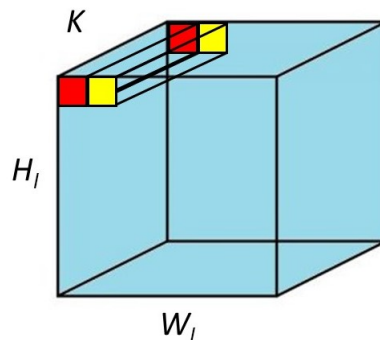


FIGURE 4.4 – Carte de caractéristiques de sortie de la couche  $l$ .

À partir de  $n_h$  images classées par le CNN comme saines, une base d'apprentissage de  $s_i \in \mathbb{R}^K$  est constituée, avec  $i = 1, \dots, N_h$  où  $N_h = (n_h \times H_l \times W_l)$ . Cette base de données

est ensuite utilisée pour entraîner un classifieur OC-SVM<sub>Seg</sub>. Le principal avantage de ce type de classifieur est leur aptitude à utiliser uniquement des échantillons normaux durant la phase d'apprentissage. Dans notre situation, ces échantillons normaux sont facilement obtenus grâce aux images préalablement classées comme saines par le CNN. Ce classifieur est également apprécié pour son efficacité calculatoire dans la phase de prédiction puisque seuls les vecteurs de support sont utilisés pour le calcul de la solution (voir chapitre 3 section 3.2.4).

Une fois l'apprentissage réalisé, le classifieur OC-SVM<sub>Seg</sub> est utilisé pour dissocier dans les régions délimitées par la DAM les pixels normaux et des pixels anormaux. De cette manière, une segmentation plus fine des défauts localisés précédemment est obtenue. Cette segmentation fine est essentielle pour la phase suivante, dans laquelle les pommes de terre endommagées et vertes sont classées par gravité.

#### 4.2.4 Classification des défauts de pommes de terre par gravité

Dans la dernière phase, nous classons les images des pommes de terre endommagées et vertes par gravité : défaut grave ou léger. Les résultats de segmentation des défauts obtenus dans la section 4.2.3 sont utilisés comme attributs d'entrée pour entraîner deux classifieurs 2C-SVM<sub>GravE</sub> et 2C-SVM<sub>GravV</sub>, le premier pour les endommagements et le deuxième pour la classe verte. Afin de simplifier la notation, nous utiliserons par la suite 2C-SVM<sub>Grav</sub> pour désigner les classifieurs 2C-SVM<sub>GravE</sub> et 2C-SVM<sub>GravV</sub>. Finalement, les attributs décrivant l'image de pomme de terre sont les suivants :

1. Nombre de pixels détectés comme anormaux.
2. Pourcentage de la surface de la pomme de terre détectée comme anormale.
3. Somme des scores de sortie du OC-SVM<sub>Seg</sub> pour les pixels détectés comme anormaux.
4. Moyenne des canaux R, V et B des pixels détectés anormaux.
5. Écart-type des canaux R, V et B des pixels détectés anormaux.

Comme nous l'avons expliqué dans la section 3.4.3 du chapitre 3, pour classer les défauts endommagés et verts par gravité, seuls les attributs de la face où le défaut détecté (verte et/ou endommagé) occupe la plus grande surface relative (pourcentage de la surface affectée) sont retenus.

Dans la section qui va suivre, nous allons évaluer la méthode proposée. Pour rappel, cette méthode comporte les quatre étapes suivantes :

Tout d'abord, un CNN est utilisé afin de classer chaque face de pomme de terre en 6 catégories (saine, endommagée, verte, dartrose, gale commune et rhizoctone). Une fois cette classification accomplie, les faces (sauf celles classées en saines) passent à la seconde étape qui consiste à estimer les régions clés qui sont à l'origine de la prédiction du réseau. Ces régions sont obtenues à l'aide des cartes d'activations de défauts (DAMs) qui sont générées à partir de la somme pondérée des cartes de caractéristiques de la dernière

couche de convolution. De cette manière, nous identifions parmi les motifs détectés dans la dernière couche de convolution ceux qui ont le plus contribué à la prise de décision du réseau.

Une fois les défauts localisés, une troisième étape de segmentation fine est effectuée en utilisant un détecteur OC-SVM entraîné avec des caractéristiques extraites du CNN à partir des faces saines. En supposant que les faces saines sont constituées uniquement de pixels normaux, l'OC-SVM détecte les pixels anormaux dans les régions délimitées par la DAM. Finalement, dans la quatrième étape, la segmentation des défauts endommagés et verts est utilisée pour effectuer une dernière classification par gravité. Pour ce faire, deux 2C-SVMs (un pour chaque défaut) sont utilisés. Le choix d'utiliser le SVM est lié à la taille de l'ensemble d'apprentissage ainsi que la dimension réduite des données d'entrée.

## 4.3 Résultats expérimentaux

Cette section, synthétise les résultats obtenus pour chaque phase de la méthode proposée. Nous montrons ainsi l'importance d'une segmentation précise des défauts afin d'effectuer une classification par gravité avec des résultats satisfaisants. Tout comme la première méthode proposée, l'implémentation a été réalisée en Matlab R2017b en utilisant un GPU NVIDIA GeForce GTX 1050Ti avec 4 Go de mémoire.

### 4.3.1 Résultats de l'étape de classification

L'étape de classification d'images, ainsi que la base de données utilisée, sont identiques à celles décrites dans la section 3.3 du chapitre précédent, nous allons donc nous focaliser sur les résultats de classification obtenus à partir du nouveau réseau GNet-Modif et les comparer avec ceux obtenus à partir de l'architecture originale GoogLeNet. Cette comparaison permet de analyser l'importance de la profondeur du réseau sur le résultat final. L'apprentissage du réseau GNet-Modif a été réalisé à l'aide d'un algorithme de descente du gradient stochastique avec un mini-batch (ou mini-lot) de 10. L'hyper-paramètre d'accélération a été fixé à 0,9 et la fonction d'entropie croisée a été utilisée comme fonction de perte. Le terme de régularisation de *weight decay* a été ajouté avec un  $\lambda$  égal à  $1 \times 10^{-4}$ . Enfin, le taux d'apprentissage a été fixé à  $1 \times 10^{-4}$  pour les couches pré-entraînées et à  $2 \times 10^{-3}$  pour les couches nouvellement rajoutées.

Nous pouvons observer dans le tableau 4.1 que le réseau GoogLeNet a été plus performante dans la phase de classification que le réseau GNet-Modif, avec des résultats supérieurs sur la totalité des 6 classes. Cela démontre que la profondeur du réseau est d'une importance capitale pour la classification, ce qui a été également attesté par différents travaux de la littérature [5, 92, 48, 25].

Nous avons également comparé les matrices de confusion obtenues avec GoogLeNet et GNet-Modif, lesquelles sont présentées dans le tableau 4.2. Comme nous pouvons le constater, l'utilisation du réseau moins profond augmente la confusion entre les classes qui se ressemblent. On remarque notamment, qu'un nombre plus important d'images appartenant à la classe endommagée ont été confondues avec la classe gale commune, passant de 1% avec GoogLeNet à 3% avec GNet-Modif. En revanche, les performances

TABLEAU 4.1 – Moyenne et écart-type de la précision, le rappel et la  $F_1$ -mesure obtenus en utilisant la validation croisée sur les réseaux GoogLeNet et GNet-Modif. Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

Critère	Classes	GoogLeNet	GNet-Modif
Précision	S	<b><math>0.95 \pm 0.02</math></b>	$0.94 \pm 0.01$
	E	<b><math>0.96 \pm 0.02</math></b>	$0.95 \pm 0.02$
	V	<b><math>0.98 \pm 0.01</math></b>	$0.93 \pm 0.04$
	D	<b><math>0.93 \pm 0.06</math></b>	$0.86 \pm 0.03$
	GC	<b><math>0.97 \pm 0.02</math></b>	$0.90 \pm 0.02$
	R	<b><math>0.92 \pm 0.04</math></b>	$0.84 \pm 0.07$
Rappel	S	<b><math>0.98 \pm 0.02</math></b>	$0.95 \pm 0.01$
	E	<b><math>0.92 \pm 0.04</math></b>	$0.85 \pm 0.05$
	V	<b><math>0.97 \pm 0.04</math></b>	<b><math>0.97 \pm 0.02</math></b>
	D	<b><math>0.78 \pm 0.06</math></b>	$0.71 \pm 0.06$
	GC	<b><math>0.95 \pm 0.02</math></b>	$0.92 \pm 0.04$
	R	<b><math>0.97 \pm 0.04</math></b>	$0.93 \pm 0.04$
$F_1$ -mesure	S	<b><math>0.97 \pm 0.01</math></b>	$0.95 \pm 0.01$
	E	<b><math>0.94 \pm 0.03</math></b>	$0.90 \pm 0.03$
	V	<b><math>0.98 \pm 0.02</math></b>	$0.95 \pm 0.01$
	D	<b><math>0.85 \pm 0.06</math></b>	$0.78 \pm 0.03$
	GC	<b><math>0.96 \pm 0.02</math></b>	$0.91 \pm 0.01$
	R	<b><math>0.95 \pm 0.04</math></b>	$0.88 \pm 0.03$

pour la classe verte sont similaires avec les deux réseaux, probablement en raison de la différence significative de cette classe avec les autres.

### 4.3.2 Résultats de localisation des défauts

Les résultats de localisation obtenus à l'aide des cartes d'activation des défauts (DAMs) ont été évalués qualitativement (figure 4.5). Une technique de seuillage a été appliquée à chaque DAM afin d'obtenir une carte de chaleur segmentée. La valeur du seuil  $h_{DAM}$  a été fixé à 0,4 de manière expérimentale. Nous avons comparé les résultats de localisation de GoogLeNet avec ceux de GNet-Modif pour vérifier si l'augmentation de la résolution de la dernière carte de caractéristiques avait un impact positif sur la localisation des défauts. La figure 4.5 montre les DAMs obtenus avec les deux réseaux. Comme prévu, de meilleurs résultats de localisation ont été obtenus avec le réseau moins profond GNet-Modif (dernière ligne de la figure 4.5).

Différentes expériences ont été également réalisées avec différentes valeurs de seuil  $h_{DAM}$  (voir figure 4.6). Les résultats obtenus confirment dans tous les cas que le réseau le moins profond GNet-Modif surpasse le réseau original GoogLeNet dans la phase de localisation.

TABLEAU 4.2 – Matrice de confusion obtenue après l’entraînement de GoogLeNet et GNet-Modif. Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone. Les valeurs sont en %.

		Classe réelle(%)					
		S	E	V	D	GC	R
Classe préd.(%)	GoogLeNet						
	S	<b>98,1</b>	6,3	2,8	20,5	3,7	1,1
	E	0,6	<b>92,4</b>	0,0	0,2	0,3	0,6
	V	0,2	0,1	<b>96,9</b>	0,5	0,1	0,0
	D	0,6	0,0	0,2	<b>78,4</b>	0,2	0,0
	GC	0,4	1,0	0,1	0,5	<b>94,5</b>	1,1
	R	0,1	0,1	0,0	0,0	1,1	<b>97,1</b>
Classe préd.(%)	GNet-Modif						
	S	<b>95,1</b>	7,9	2,2	26,3	3,9	2,3
	E	0,7	<b>85,5</b>	0,0	0,2	0,4	0,0
	V	1,4	1,1	<b>97,3</b>	1,4	0,3	0,0
	D	0,8	1,2	0,3	<b>71,2</b>	0,7	1,7
	GC	1,8	3,0	0,2	0,9	<b>92,8</b>	2,9
	R	0,2	1,3	0,0	0,0	1,9	<b>93,1</b>

### 4.3.3 Résultats de segmentation des défauts

Les caractéristiques de la 2ème couche de convolution du CNN entraîné pour classer les images de pommes de terre ont été extraites. La taille de la carte de caractéristiques obtenue est de  $56 \times 56 \times 192$ . Cette carte peut être interprétée comme 3136 vecteurs de caractéristiques de dimension 192, où chaque vecteur caractérise une petite région carré de l’image initiale. Les attributs extraits des images classées comme saines par le réseau ont été utilisés pour entraîner un détecteur OC-SVM<sub>Seg</sub> avec un noyau gaussien. Ce détecteur a été ensuite utilisé pour localiser les pixels anormaux des images classées comme défectueuses par le CNN. Cette étape de détection des anomalies (ou défauts) donne une carte de chaleur, où les valeurs négatives représentent les pixels défectueux. Une opération d’interpolation bilinéaire a été ensuite effectuée sur la carte de chaleur obtenue par le OC-SVM<sub>Seg</sub>, afin d’obtenir une carte de la même taille que l’image d’entrée et que la DAM ( $224 \times 224$ ). Un exemple de la sortie du OC-SVM<sub>Seg</sub>, ainsi que de la localisation obtenue à partir de la DAM, est présenté à la figure 4.7.

Durant l’étape de classification, pour chaque image testée et classée non saine, la DAM est calculée afin d’estimer la localisation des défauts présents sur l’image. Le classifieur OC-SVM<sub>Seg</sub> est ensuite utilisé pour détecter les pixels anormaux. Afin de montrer la pertinence de cette étape de segmentation, nous présentons dans la figure 4.8 des résultats obtenus selon deux scénarios. Dans le premier cas, la localisation du défaut est effectuée

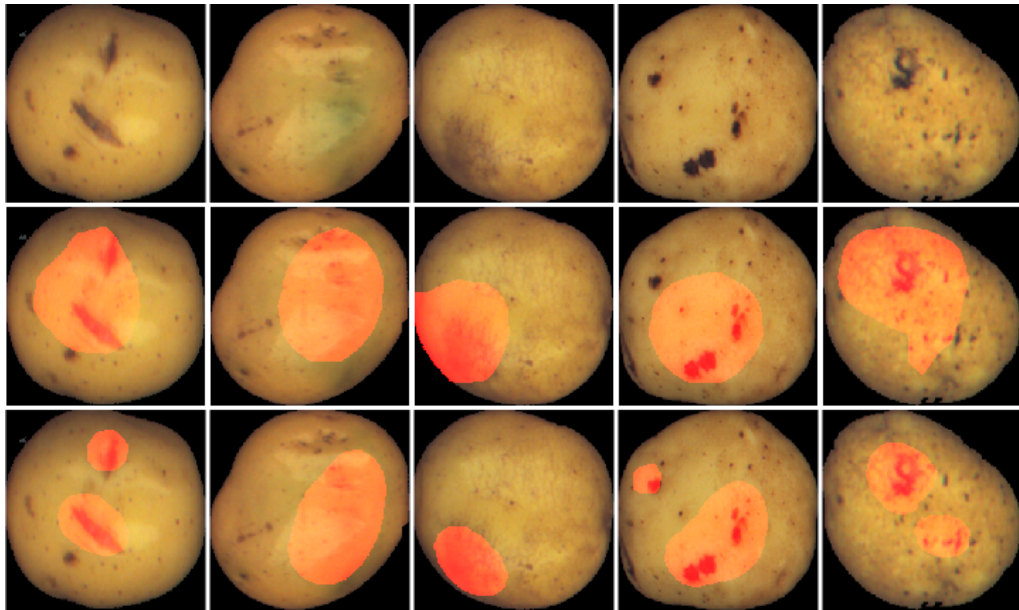


FIGURE 4.5 – Exemple des DAMs générées en utilisant le réseau GoogLeNet et GNet-Modif. Par lignes : image d’entrée, DAM obtenue avec GoogLeNet, DAM obtenue avec GNet-Modif. Par colonne différentes classes : endommagée, verte, dartoise, gale commune et rhizoctone.

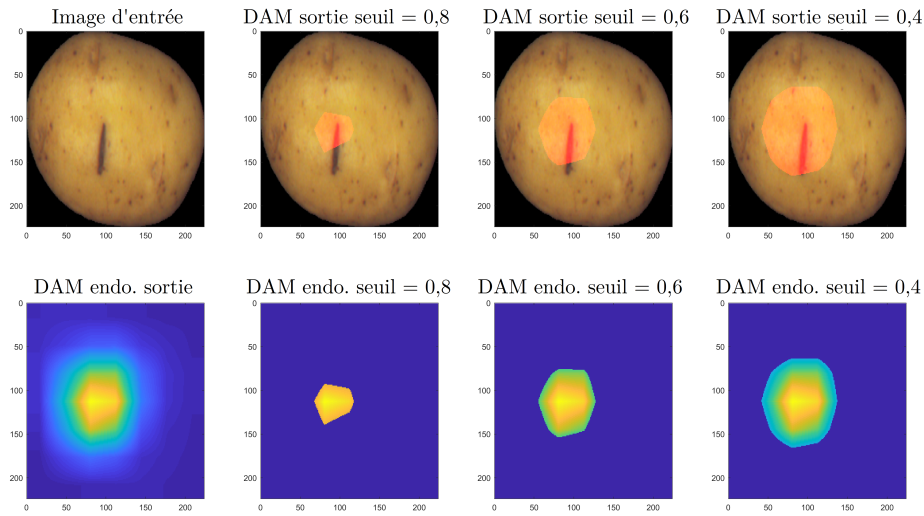
seulement à partir de la DAM. Dans le second cas, le défaut localisé par la DAM est ensuite segmenté finement à l’aide du classifieur  $OC-SVM_{Seg}$ .

À partir de ces résultats, nous pouvons voir que la segmentation fine des défauts est réalisée avec succès. Nous constatons également l’importance fondamentale de cette étape pour l’estimation précise de l’étendue de chaque défaut. En effet, les résultats de localisation obtenus à partir de la DAM combinée avec la segmentation fine sont bien plus précis que ceux obtenus en appliquant uniquement la DAM.

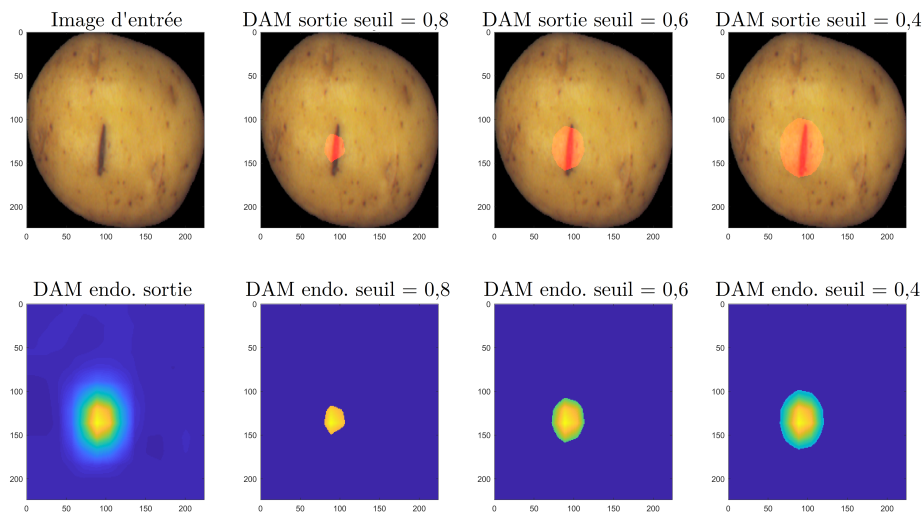
La pertinence de cette combinaison sera corroborée dans la section suivante, où les résultats de segmentation seront utilisés pour classer les défauts, endommagements et verdissements, par gravité.

#### 4.3.4 Résultats de la classification des défauts par gravité

Dans la phase finale, les images de pommes de terre endommagées et vertes sont classées par gravité. Seules les pommes de terre issues de ces deux classes ont été prises en compte pour l’apprentissage des modèles. Comme nous l’avons vu dans la section 3.4.3 du chapitre précédent, les étiquettes relatives à la gravité sont disponibles uniquement au niveau de la pomme de terre (4 faces, 1 étiquette). De ce fait, nous avons utilisé comme entrée des deux classifieurs  $2C-SVM_{Grav}$  (un pour les endommagements et l’autre pour la classe verte) uniquement la face où le défaut détecté et segmenté occupe la plus grande surface relative. Les deux classifieurs ont été entraînés pour classer les tubercules endommagés et verts par gravité : endommagée légère (EL) ou endommagée grave (EG), ainsi que verte



(a)



(b)

FIGURE 4.6 – Exemple des DAMs générées avec des valeurs de seuils  $h_{DAM}$  différentes. Par colonne : image originale ; seuil = 0,8 ; seuil = 0,6 ; seuil = 0,4. a) GoogLeNet et b) GNet-Modif.

légère (VL) ou verte grave (VG).

Afin de pouvoir évaluer l'importance de la phase de segmentation, nous avons également analysé le scénario où nous n'utilisons que les résultats de la localisation à partir des DAMs pour classer les pommes de terre par gravité, c'est-à-dire sans appliquer la phase de segmentation fine. La validation croisée de *5-folds*, ainsi que la recherche par grille, ont été appliquées pour sélectionner les hyper-paramètres  $\sigma$  et  $C$  de chaque classifieur  $2C-SVM_{Grav}$ .

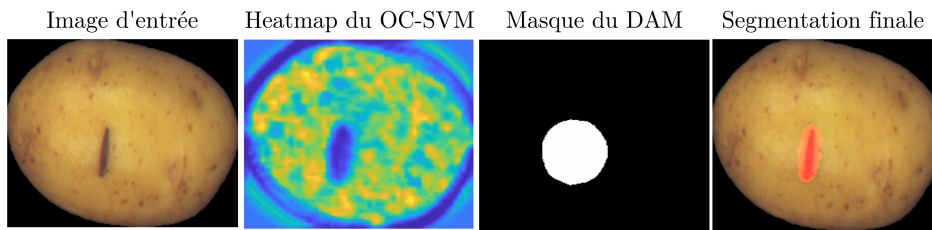


FIGURE 4.7 – Exemple des sorties intermédiaires de la méthode proposée jusqu’au résultat final. De gauche à droite : image d’entrée, sortie de la carte de chaleur du classifieur OC-SVM<sub>Seg</sub>, DAM obtenue ( $h_{DAM} = 0,4$ ) et image finale segmentée.

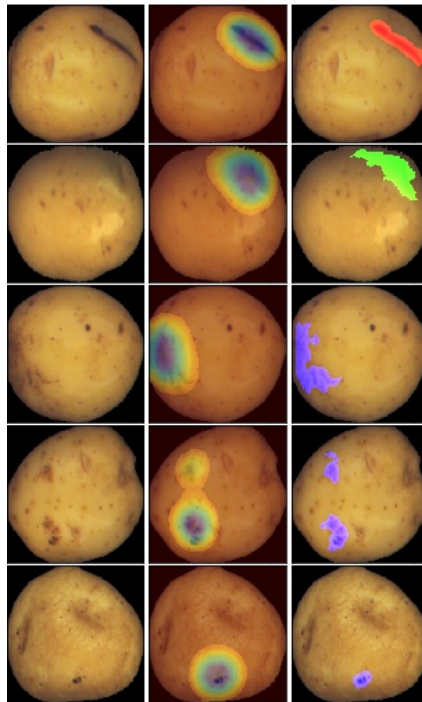


FIGURE 4.8 – Résultats de la localisation et de la segmentation des défauts sur échantillons appartenant à l’ensemble de test (colonne de gauche). Par ligne, les différentes classes : endommagée, verte, dartrose, gale commune et rhizoctone. Nous pouvons observer que les DAMs (au centre) donnent les localisations des défauts classés, localisations utilisées pour la segmentation de grossière à fine (troisième colonne).

Les matrices de confusion (tableaux 4.3 et 4.4) montrent les résultats obtenus pour les pommes de terre endommagées et vertes respectivement. Comme nous pouvons l’observer, le GNet-Modif a été plus précis pour différencier les défauts légers des défauts graves, ce qui confirme que la localisation et la segmentation faites par ce réseau ont été plus précises. Nous pouvons également observer que la phase de segmentation fine est indispensable pour différencier correctement les défauts par gravité. La confusion entre les classes endommagée grave et endommagée légère a été réduite de plus de moitié lorsqu’une segmentation fine du défaut a été effectuée. En effet, 37,3% de pommes de terre endommagées graves ont été classées comme « endommagées légères » en utilisant unique-



TABLEAU 4.3 – Classification des pommes de terre en S=Saines, EL=Endommagée Légère et EG=Endommagée Grave. Matrices de confusion obtenues avec trois configurations différentes : GoogLeNet+DAM+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub> (nommé « GNet+DAM » pour simplifier), GNet-Modif+DAM+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub> (nommé « GNet-Mod+DAM ») et GoogLeNet+DAM+2C-SVM<sub>Grav</sub> sans l'étape de segmentation de grossière à fine (nommé « Sans segmentation »).

		Classes	Classe réelle(%)		
			S	EL	EG
Classe préd.(%)	GNet+DAM				
	S		<b>99,3</b>	6,7	0,9
	EL	0,7	<b>89,1</b>	15,5	
	EG	0,0	4,2	<b>83,6</b>	
Classe préd.(%)	GNet-Mod+DAM				
	S		<b>99,7</b>	10,5	1,8
	EL	0,3	<b>87,4</b>	10,0	
	EG	0,0	2,1	<b>88,2</b>	
Classe préd.(%)	Sans segmentation				
	S		<b>99,3</b>	6,7	0,9
	EL	0,7	<b>85,7</b>	37,3	
	EG	0,0	7,6	<b>61,8</b>	

ment la localisation, contre 15,5% avec la segmentation fine. Une situation similaire s'est produite avec les classes verte grave et légère : 50,5% de pommes de terre vertes légères classées vertes graves sans utiliser la segmentation fine. Ce pourcentage a été réduite à 15,1% grâce à la segmentation fine.

En comparant les résultats obtenus à l'aide de « GNet+DAM » et « GNet-Mod+DAM » nous pouvons constater que les performances sont légèrement meilleures avec « GNet-Mod+DAM ». Nous pouvons conclure que l'étape de segmentation fine est indispensable pour obtenir des résultats de classification par gravité satisfaisants. En revanche, garder une meilleure résolution à la sortie du CNN (GNet versus GNet-Modif) n'est pas décisif pour définir la gravité des défauts.

### 4.3.5 Évaluation globale du tubercule

L'ensemble de données test multi-labels multi-classes contient 722 tubercules. Il a été utilisé afin de tester l'ensemble du système proposé. La caractérisation de la pomme de terre entière a été réalisée en tenant compte, pour chaque face du tubercule, des étiquettes de sortie obtenues lors des étapes précédentes. Étant donné que dans l'ensemble de test, chaque images peut appartenir à plus d'une classe, nous avons appliqué la règle de décision introduite dans les sections 4.2.1 et 4.2.2 pour décider des classes de sortie finales (voir exemple dans la figure 4.9).

Trois configurations différentes de la méthode ont été évaluées : GoogLeNet, GNet-Modif et une combinaison des deux réseaux (nommé « Combinaison » dans la suite),

TABLEAU 4.4 – Classification des pommes de terre en S=Saines, VL=Verte Légère et VG=Verte Grave. Matrices de confusion obtenues avec trois configurations différentes : GoogLeNet+DAM+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub> (nommé « GNet+DAM » pour simplifier), GNet-Modif+DAM+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub> (nommé « GNet-Mod+DAM ») et GoogLeNet+DAM+2C-SVM<sub>Grav</sub> sans l'étape de segmentation de grossière à fine (nommé « Sans segmentation »).

		Classes	Classe réelle(%)		
			S	VL	VG
Classe préd.(%)	GNet+DAM				
	S		<b>100</b>	3,2	0,0
	VL	0,0	<b>81,7</b>	6,1	
	VG	0,0	15,1	<b>93,9</b>	
Classe préd.(%)	GNet-Mod+DAM				
	S	<b>99,8</b>	2,1	0,0	
	VL	0,2	<b>83,9</b>	5,3	
	VG	0,0	14,0	<b>94,7</b>	
Classe préd.(%)	Sans segmentation				
	S	<b>100</b>	3,3	0,0	
	VL	0,0	<b>46,2</b>	7,2	
	VG	0,0	50,5	<b>92,8</b>	



FIGURE 4.9 – Exemple d'une image représentant une face de pomme de terre contenant deux défauts (endommagé et vert) segmentée par la méthode GNet-Modif+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub>. De gauche à droite : image d'entrée, sortie du DAM et sortie de l'étape de segmentation.

où nous avons classé les images avec GoogLeNet et nous avons localisé et segmenté les défauts grâce au GNet-Modif. L'objectif principal de cette combinaison est de tirer profit des points forts de chaque réseau : GoogLeNet est plus performant pour la classification des images et GNet-Modif est, quant à lui, plus efficace pour la localisation et segmentation des défauts. Les résultats obtenus avec la première méthode introduite au chapitre 3 sont également présentés afin de réaliser une analyse comparative complète. Pour ce qui est de la première méthode, nous analysons deux configurations. Dans le premier cas, nous avons utilisé GoogLeNet pour classer les images et une combinaison d'un auto-encodeur et d'un OC-SVM<sub>Dét</sub> pour classer les patches verts et endommagés. Dans le deuxième cas, nous avons remplacé l'OC-SVM<sub>Dét</sub> par un 2C-SVM<sub>Dét</sub> pour comparer la classification des

patches après un apprentissage non supervisée et supervisée. Les résultats de ces deux configurations seront comparés à ceux de l'approche faiblement supervisée proposée dans ce chapitre.

TABLEAU 4.5 – Précision, rappel et  $F_1$ -mesure des méthodes appliquées sur la base de données de test. Configurations de la méthode du chapitre précédent : GoogLeNet+AE+OC-SVM<sub>Dét</sub>+2C-SVM<sub>Grav</sub> (« GNet+AE+OC-SVM ») et GoogLeNet+AE+2C-SVM<sub>Dét</sub>+SVM<sub>Grav</sub> (« GNet+AE+2C-SVM »). Configurations pour la méthode de ce chapitre : GoogLeNet+DAM+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub> (« GNet+DAM »), GNet-Modif+DAM+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub>(« GNet-Mod+DAM ») et une combinaison des deux réseaux (nommé « Combinaison+DAM »). Les classes sont : S=Saine, EL=Endommagée Légère, EG=Endommagée Grave, VL=Verte Légère, VG=Verte Grave, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

Critère	Classes	GNet+ AE+ OC-SVM	GNet+ AE+ 2C-SVM	GNet+ DAM	GNet- Mod+DAM	Combinaison+ DAM
Précision	S	<b>0,98</b>	<b>0,98</b>	<b>0,98</b>	0,95	<b>0,98</b>
	EL	0,83	<b>0,90</b>	0,83	0,88	0,85
	EG	0,83	0,86	0,79	0,84	<b>0,91</b>
	VL	0,91	0,88	0,89	0,83	<b>0,93</b>
	VG	0,89	<b>0,98</b>	0,90	0,92	0,94
	D	<b>0,92</b>	<b>0,92</b>	<b>0,92</b>	0,91	0,90
	GC	0,95	0,95	0,95	<b>0,96</b>	0,94
	R	<b>0,88</b>	<b>0,88</b>	0,84	0,78	0,81
	Moyenne	0,90	<b>0,92</b>	0,89	0,88	0,91
Rappel	S	0,98	0,98	0,98	<b>0,99</b>	0,98
	EL	0,92	0,94	0,91	0,87	<b>0,95</b>
	EG	0,73	<b>0,88</b>	0,77	<b>0,88</b>	0,83
	VL	0,67	<b>0,91</b>	0,70	0,85	0,83
	VG	<b>0,97</b>	0,95	0,96	<b>0,97</b>	<b>0,97</b>
	D	<b>0,82</b>	<b>0,82</b>	<b>0,82</b>	0,75	<b>0,82</b>
	CS	0,87	0,87	<b>0,88</b>	<b>0,88</b>	<b>0,88</b>
	R	0,95	0,95	0,95	0,95	0,95
	Moyenne	0,87	<b>0,91</b>	0,87	0,89	0,90
$F_1$ -mesure	S	<b>0,98</b>	<b>0,98</b>	<b>0,98</b>	0,97	<b>0,98</b>
	EL	0,88	<b>0,92</b>	0,86	0,88	0,90
	EG	0,78	<b>0,87</b>	0,78	0,86	<b>0,87</b>
	VL	0,78	<b>0,89</b>	0,78	0,84	0,87
	VG	0,93	<b>0,96</b>	0,93	0,95	<b>0,96</b>
	D	<b>0,87</b>	<b>0,87</b>	<b>0,87</b>	0,82	0,86
	GC	0,91	0,91	0,91	0,91	0,91
	R	<b>0,91</b>	<b>0,91</b>	0,89	0,86	0,88
	Moyenne	0,88	<b>0,91</b>	0,88	0,89	0,90

À partir des résultats obtenus, présentés dans le tableau 4.5, nous pouvons observer que l'approche faiblement supervisée proposée permet de classer les pommes de terre avec précision. En regardant les résultats obtenus avec cette approche, les meilleures performances ont été obtenues en combinant les deux réseaux (« Combinaison+DAM »), avec une valeur moyenne de  $F_1$ -mesure de 0,90, contre 0,88 et 0,89 pour GoogLeNet (« GNet+DAM ») et GNet-Modif (« GNet-Mod+DAM ») respectivement. Les meilleures performances ont été obtenues sur la classe saine avec une  $F_1$ -mesure maximal de 0,98. La dartoise a la  $F_1$ -mesure la plus faible (0,86 avec la « Combinaison+DAM ») en raison de la confusion entre cette classe et la classe saine. Nous pouvons également observer que la méthode proposée, avec les différentes configurations, est capable de classer avec précision les pommes de terre endommagées et vertes par gravité, en tirant parti des résultats de localisation et de segmentation des défauts.

En comparant les résultats des approches faiblement supervisées avec ceux obtenus avec et sans supervision (chapitre 3), nous constatons que même si la nouvelle méthode n'utilise aucune étiquette par patch ni par pixel pour réaliser la segmentation des défauts, les résultats sont très similaires à ceux obtenus avec la méthode supervisée « GNet+AE+2C-SVM », laquelle utilise une base de données annotée par patch, et surpasse la méthode non supervisée « GNet+AE+OC-SVM », laquelle réalise une localisation non supervisée. Notre approche faiblement supervisée nous permet d'obtenir des résultats de classification par gravité qui répondent aux attentes opérationnelles sans avoir recours à une base de données étiquetée par pixels ou par patches.

À partir du tableau 4.5 nous pouvons également observer que la précision, le rappel et la  $F_1$ -mesure des classes endommagée et verte (qui sont divisées par gravité) sont parmi les plus bas. Cela peut s'expliquer par le fait qu'il existe une forte ressemblance entre les classes légères et graves, et la confusion entre elles est très fréquente. Cette confusion se produit fréquemment lors du processus de classification par les experts humains. Pour confirmer nos suppositions sur l'origine de cette confusion, nous avons calculé les résultats finaux en regroupant les classes légère et grave. Il faut noter que dans cette situation, les étapes de localisation et de segmentation n'affectent pas les résultats puisqu'elles ne servent qu'à séparer les défauts selon leur gravité. Le tableau 4.6 montre les résultats obtenus, lesquels confirment notre hypothèse. En effet, la  $F_1$ -mesure obtenue pour la classe endommagée et verte augmente considérablement lorsqu'on ne prend pas en compte la division par gravité de ces défauts.

## 4.4 Réseau entièrement convolutif pour la segmentation sémantique d'images

Les réseaux de neurones convolutifs classiques abordés en détail dans le chapitre 3 (AlexNet, GoogLeNet, VGG-16) ont été initialement proposés pour résoudre des problèmes de classification d'images, où une unique classe de sortie est prédite pour chaque entrée. Ces réseaux sont principalement composés de couches de convolution et de regroupement, ce qui induit une réduction de la résolution des cartes de caractéristiques au fur et à mesure que l'on avance en profondeur dans le réseau. Cette réduction de résolution est d'import-

TABLEAU 4.6 – Précision, rappel et  $F_1$ -mesure des méthodes proposées sur la base de données de test sans la distinction par gravité pour les classes endommagée et verte. Configurations de la méthode du chapitre précédent : GoogLeNet+AE+OC-SVM<sub>Dét</sub>+2C-SVM<sub>Grav</sub> (« GNet+AE+OC-SVM ») et GoogLeNet+AE+2C-SVM<sub>Dét</sub>+2C-SVM<sub>Grav</sub> (« GNet+AE+2C-SVM »). Configurations pour la méthode de ce chapitre : GoogLeNet+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub> (« GNet+DAM »), GNet-Modif+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub> (« GNet-Mod+DAM ») et une combinaison des deux réseaux (nommé « Combinaison+DAM »). Les classes sont : S=Saine, EL+EG=Endommagée Légère et Grave , VL+VG=Verte Légère et Grave, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

Critère	Classes	GNet+ AE+ OC-SVM	GNet+ AE+ 2C-SVM	GNet+ DAM	GNet- Mod+DAM	Combinaison+ DAM
Précision	S	<b>0,98</b>	<b>0,98</b>	<b>0,98</b>	0,95	<b>0,98</b>
	EL+EG	0,93	0,93	0,91	<b>0,94</b>	0,91
	VL+VG	<b>1</b>	<b>1</b>	<b>1</b>	0,95	<b>1</b>
	D	<b>0,92</b>	<b>0,92</b>	<b>0,92</b>	0,91	0,90
	GC	0,95	0,95	0,95	<b>0,96</b>	0,94
	R	<b>0,88</b>	<b>0,88</b>	0,84	0,78	0,81
	Moyenne	<b>0,94</b>	<b>0,94</b>	0,93	0,92	0,92
Rappel	S	0,98	0,98	0,98	<b>0,99</b>	0,98
	EL+EG	<b>0,97</b>	<b>0,97</b>	<b>0,97</b>	0,94	<b>0,97</b>
	VL+VG	0,99	0,99	0,99	0,99	0,99
	D	0,82	0,82	0,82	0,75	0,82
	CS	0,87	0,87	<b>0,88</b>	<b>0,88</b>	<b>0,88</b>
	R	0,95	0,95	0,95	0,95	0,95
	Moyenne	<b>0,93</b>	<b>0,93</b>	<b>0,93</b>	0,92	0,93
$F_1$ -mesure	S	<b>0,98</b>	<b>0,98</b>	<b>0,98</b>	0,97	<b>0,98</b>
	EL+EG	<b>0,95</b>	<b>0,95</b>	0,94	0,94	0,94
	VL+VG	0,99	0,99	0,99	0,97	<b>1</b>
	D	<b>0,87</b>	<b>0,87</b>	<b>0,87</b>	0,82	0,86
	GC	0,91	0,91	0,91	0,91	0,91
	R	<b>0,91</b>	<b>0,91</b>	0,89	0,86	0,88
	Moyenne	<b>0,93</b>	<b>0,93</b>	<b>0,93</b>	0,91	0,93

tance capitale pour diminuer le nombre de paramètres du réseau, mais elle ne convient pas dans le contexte de la segmentation sémantique des images, où chaque pixel constitutif de l'image doit être classé individuellement.

Dans notre approche faiblement supervisée, nous avons employé la technique de DAM et un détecteur OC-SVM pour segmenter les images. Toutefois, cette approche implique l'apprentissage de plusieurs modèles et sa mise en œuvre est donc lourde. Pour traiter ce problème, les réseaux de neurones entièrement convolutifs (*Fully-Convolutional Networks* - FCN), proposés par Long et al. [6], sont une piste à explorer. L'entraînement de ce

type de réseau est effectué de bout en bout en utilisant l'image brute à segmenter comme entrée, et en obtenant une carte de classification dense (*classification map*) comme sortie. Cette carte dense a la même taille que l'image d'entrée et chaque pixel qui la compose est classé selon un nombre de classes prédéfini.

Afin de transformer un réseau de neurones convolutifs conçu pour la classification d'images (composée de plusieurs blocs de convolution, de couches de regroupement et finalement, de couches entièrement connectées) en un réseau entièrement convolutif FCN destiné à la segmentation d'images, deux modifications importantes doivent être effectuées. Premièrement, les couches entièrement connectées du bloc de classification du CNN sont remplacées par des couches de convolution avec des filtres qui ont la même taille que la région d'entrée (voir figure 4.10). Le remplacement des couches entièrement connectées

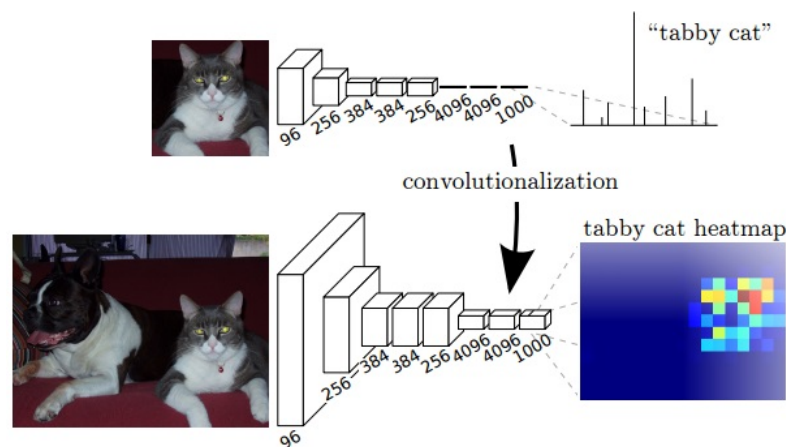


FIGURE 4.10 – Remplacement des couches entièrement connectées par des couches de convolution ce qui permet notamment d'introduire dans le réseau des images d'entrée de taille différentes [6].

par des couches de convolution réduit la complexité calculatoire du réseau et permet de se défaire de la contrainte relative à la taille des données d'entrée du réseau. En effet, l'entrée d'une couche entièrement connectée est un vecteur forcément de taille fixe. Cet aspect s'explique par le fait que chaque neurone d'une couche entièrement connectée est lié à toutes les sorties de la couche précédente, donc une fois la couche entièrement connectée définie et entraînée avec un nombre de connexions donné, ce nombre ne peut être modifié à moins de redéfinir de nouveau la couche. Pour une couche de convolution, cette contrainte n'existe pas. En effet, un filtre de convolution peut être convolué avec les cartes de caractéristiques indépendamment de leur taille (largeur et longueur).

La deuxième modification majeure est réalisée afin d'obtenir des dimensions de sortie (largeur et longueur) égales aux dimensions d'entrée, et ainsi pouvoir calculer l'erreur de prédiction au niveau du pixel. En raison des couches de regroupement couramment utilisées dans les CNNs, les cartes de caractéristiques à la sortie du réseau sont de taille réduite par rapport à l'entrée. Par conséquent, à la fin du réseau entièrement convolutif, un bloc dit de sur-échantillonnage est ajouté afin d'augmenter la taille des cartes de caractéristiques. Le bloc de sur-échantillonnage est composé de couches de convolution

transposées, à partir desquelles il est possible d'effectuer une augmentation de la résolution des cartes de caractéristiques. En effet, dans une couche de convolution, pour chaque champ récepteur une valeur de sortie est obtenue, alors que dans une couche de convolution transposée, une seule valeur est prise comme entrée afin d'obtenir plusieurs valeurs de sortie (voir figure 4.11). L'utilisation des couches de convolution transposées peut être assimilée à l'utilisation d'une méthode d'interpolation prédéfinie, telle que l'interpolation bilinéaire. Cependant, l'avantage d'utiliser des couches de convolution transposées est que le réseau apprend à réaliser l'opération de sur-échantillonnage de manière adaptée aux images traitées grâce aux poids des filtres qui sont appris lors de l'entraînement du réseau.

En plus de l'utilisation de couches de convolution transposées, des sauts (*skips*) peuvent

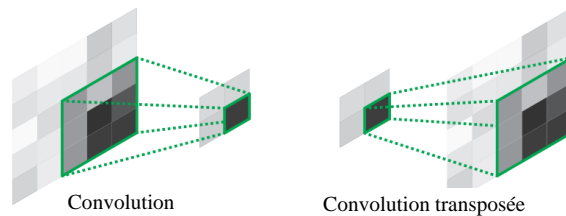


FIGURE 4.11 – Opération de convolution et de convolution transposée. Image extraite de l'article de Noh et al. [7].

être rajoutés afin de combiner les cartes de caractéristiques d'une couche donnée avec celles de couches moins profondes (voir exemple de la figure 4.12). Ces sauts sont généralement utilisés afin d'aider le réseau à réaliser la segmentation de l'image de la manière la plus fine et précise possible.

De nombreuses méthodes reposant sur un apprentissage profond inspirées du FCN ont été proposées avec succès pour réaliser différentes tâches de segmentation sémantique sur des images [7, 186, 187, 188]. Cependant, le fait que ces méthodes nécessitent une base de données large et variée étiquetée au niveau du pixel constitue un inconvénient majeur et un frein à leur utilisation.

Dans le contexte de nos travaux, nous n'avions pas à notre disposition une base de données annotée au niveau du pixel et sa création aurait été une tâche complexe et chronophage. C'est pourquoi, dès le début de nos travaux, nous nous sommes orientés vers des méthodes permettant de localiser ou de segmenter les défauts des pommes de terre sans utiliser des annotations au niveau du pixels. Malgré l'obtention d'excellents résultats grâce à ces méthodes, elles restent plus complexes que celles basées sur des FCNs, ce qui a tendance à affecter leur performances en terme de complexité calculatoire. Compte tenu du fait que nos méthodes sont destinées à être exploitées dans un environnement industriel, l'obtention d'un temps de traitement réduit est un aspect primordial. Pour cette raison, nous proposons d'utiliser des images segmentées à partir de notre méthode faiblement super-

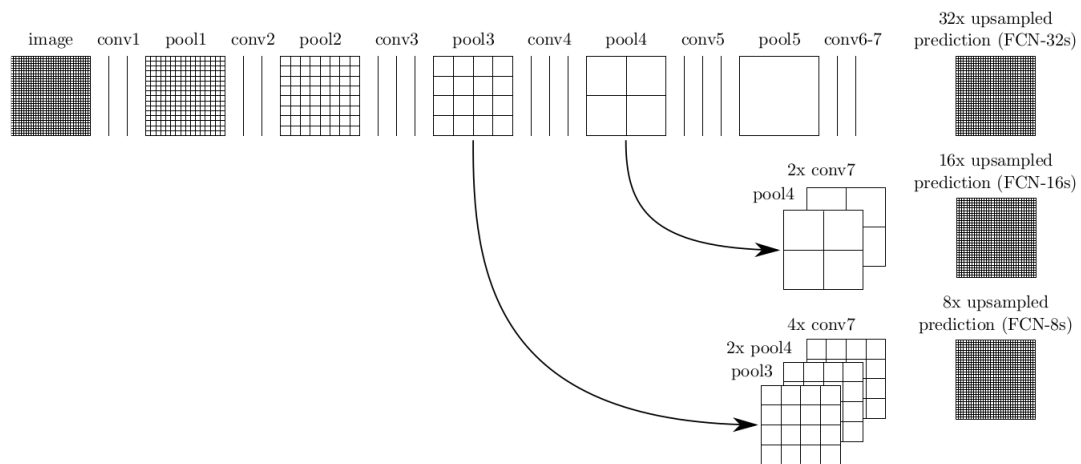


FIGURE 4.12 – Architecture du réseau entièrement convolutif proposé par Long et al. [6]. L'ajout de sauts se fait dans le but de combiner les informations des couches plus profondes avec celles présentes dans des couches moins profondes.

visée présentée dans la section 4.2 afin d'entraîner de manière supervisée un réseau dont l'architecture est inspirée du FCN.

## 4.5 Système proposé pour la segmentation sémantique d'images

La figure 4.13 illustre la méthode proposée pour la segmentation sémantique d'images de pommes de terre. Elle se compose de trois phases principales :

1. Tout d'abord, la méthode décrite dans la section 4.2 est utilisée afin de créer de manière automatique une base de données étiquetée au niveau du pixel (figure 4.13 étape A).
2. Ensuite, une nouvelle architecture inspiré de GoogLeNet, nommé FC-GoogLeNet, effectue simultanément une classification au niveau de l'image et au niveau du pixel (figure 4.13 étape B).
3. Finalement, les résultats de segmentation sémantique des pommes de terre endommagées et vertes sont utilisés comme entrée de deux  $2C-SVM_{\text{Grav}}$  afin de classer ces défauts par gravité (figure 4.13 étape C).

### 4.5.1 Etiquetage automatique au niveau du pixel d'une base de données

La méthode faiblement supervisée détaillée dans la section 4.2 (« Combinaison+DAM ») est utilisée pour créer une base de données avec des étiquettes au niveau du pixel. Dans la figure 4.14 nous pouvons observer un exemple de ces images. Chaque pixel est identifié par



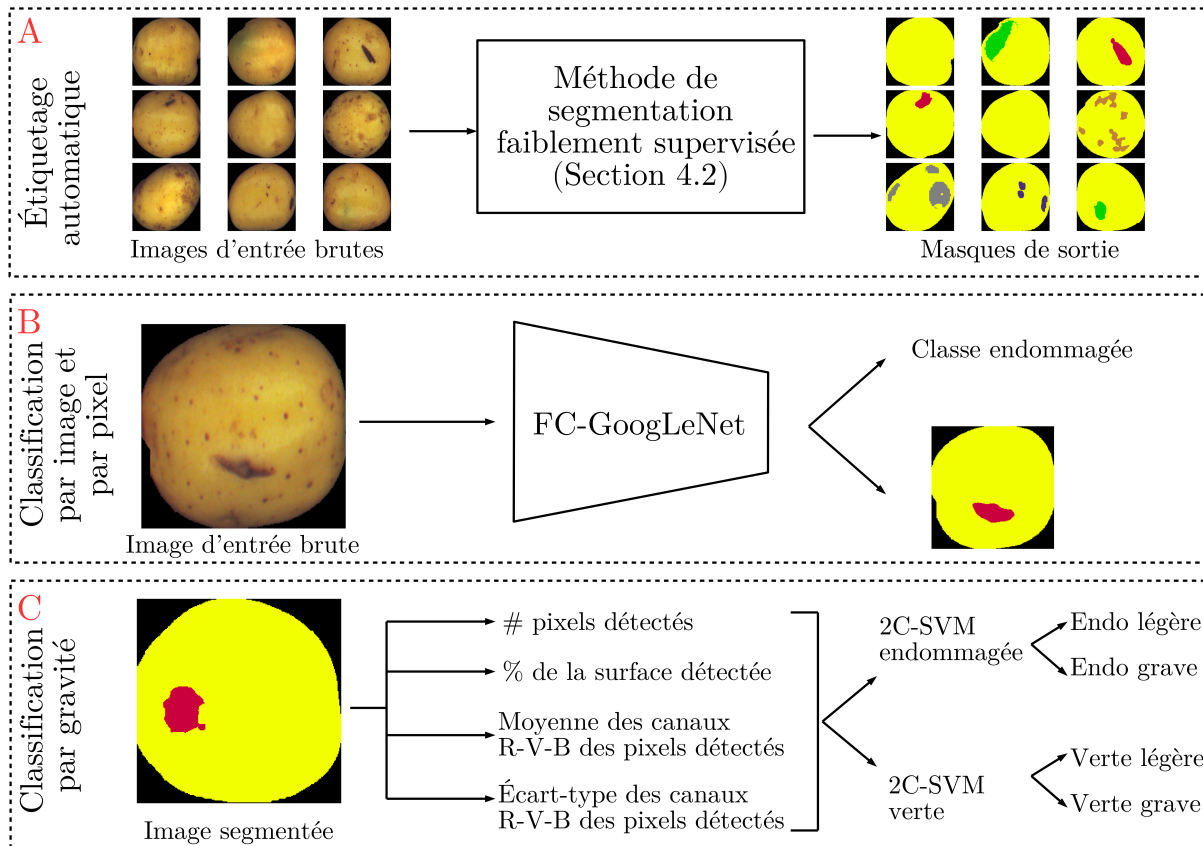


FIGURE 4.13 – Méthode proposée pour la segmentation sémantique d'images.

une couleur différente afin de représenter une classe spécifique : noir pour l'arrière plan, jaune la peau saine, rouge les endommagements, vert pour les taches vertes, gris pour la dartrose, marron indique la gale commune et, finalement, violet pour le rhizoctone. La base de données ainsi créée contient certaines erreurs d'annotations, mais dans notre domaine d'application il est fréquent que les bases de données étiquetées par des humains contiennent également des erreurs [21] notamment à cause de la subjectivité humaine et de la difficulté d'identifier certains défauts ou maladies telle que la dartrose. Nous faisons l'hypothèse que la perte de qualité induite par ces erreurs est limitée aux cas de défauts faiblement présents.

#### 4.5.2 Segmentation sémantique d'images de pommes de terre

Une nouvelle architecture de réseau de neurones convolutifs est introduite afin de réaliser la segmentation sémantique d'images. L'objectif est de classer chaque pixel d'une image dans l'une des 7 classes suivantes : arrière-plan, peau saine, endommagée, verte, dartrose, gale commune et rhizoctone.

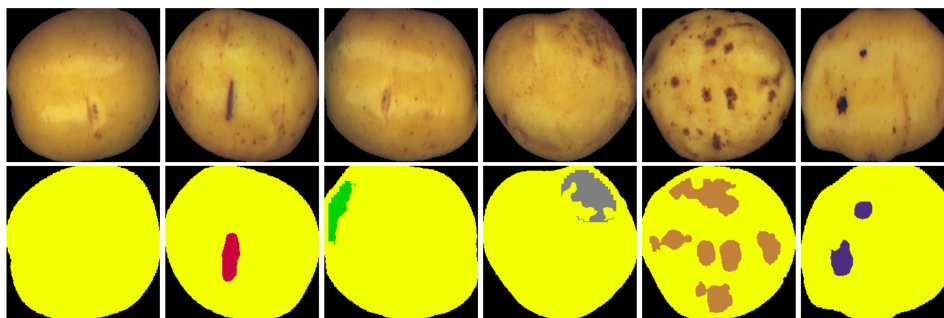


FIGURE 4.14 – Exemple des images étiquetées par pixels appartenant à la base de donnée d'entraînement. Les étiquettes sont créées en utilisant notre méthode de segmentation faiblement supervisée (« Combinaison+DAM »). De gauche à droite les classes : saine, endommagée, verte, dartrose, gale commune et rhizoctone. Par ligne : image d'entrée, image de sortie segmentée utilisée comme *ground-truth* pour entraîner le réseau FC-GoogLeNet.

L'architecture du réseau créé, nommé FC-GoogLeNet, est obtenue en apportant les modifications suivantes au réseau GoogLeNet déjà modifié et entraîné avec nos images de pommes de terre :

- *Bloc-1* : la sortie du 9ème et dernier module d'inception de GoogLeNet (taille  $7 \times 7 \times 1024$ ), est suivie de deux nouvelles couches. Une première couche de convolution avec  $n_{class}$  filtres de taille  $1 \times 1$ , où  $n_{class}$  est le nombre de classes de notre problème, y compris l'arrière-plan, et un pas de 1. Une deuxième couche de convolution transposée avec  $n_{class}$  filtres de taille  $4 \times 4$ , et un pas de 4.
- *Bloc-2* : la sortie du 7ème module d'inception (taille  $14 \times 14 \times 832$ ) est suivie d'une couche de convolution de  $n_{class}$  filtres de taille  $1 \times 1$  et un pas de 1. Puis, une couche de convolution transposée est rajoutée afin d'augmenter la résolution des cartes de caractéristiques, avec  $n_{class}$  filtres de taille  $2 \times 2$  et un pas de 2.
- *Bloc-3* : la sortie du 2ème module d'inception (taille  $28 \times 28 \times 480$ ) est suivie d'une couche de convolution de  $n_{class}$  filtres de taille  $1 \times 1$ , un pas de 1.
- *Bloc-sortie* : les sorties des *Bloc-1*, *Bloc-2* et *Bloc-3* sont concaténées afin d'obtenir une carte de caractéristiques de 21 canaux de taille  $28 \times 28$ . La couche de concaténation est suivie d'une couche de convolution avec  $n_{class}$  filtres de taille  $1 \times 1$  et un pas de 1. Puis, une dernière couche de convolution transposée de  $n_{class}$  filtres de taille  $8 \times 8$  et un pas de 8 est rajoutée pour revenir à la résolution initiale. Finalement, le volume de sortie de ce bloc a une taille de  $n_{class} \times 224 \times 224$ . Puis, une dernière fonction d'activation *softmax* est appliquée afin d'obtenir la prédiction finale au niveau du pixel.

Toutes les nouvelles couches de convolution (sauf la dernière) du réseau FC-GoogLeNet

sont suivies d'une fonction d'activation ReLU. La figure 4.15 illustre les modifications proposées. En effet, le FC-GoogLeNet proposé a deux sorties : dans la première on obtient une classification globale de l'image et dans la deuxième, chaque pixel qui compose l'image est classé individuellement. La classification au niveau de l'image est conservée afin de maintenir la performance de GoogLeNet déjà entraîné avec notre base de données originale étiquetée par image.

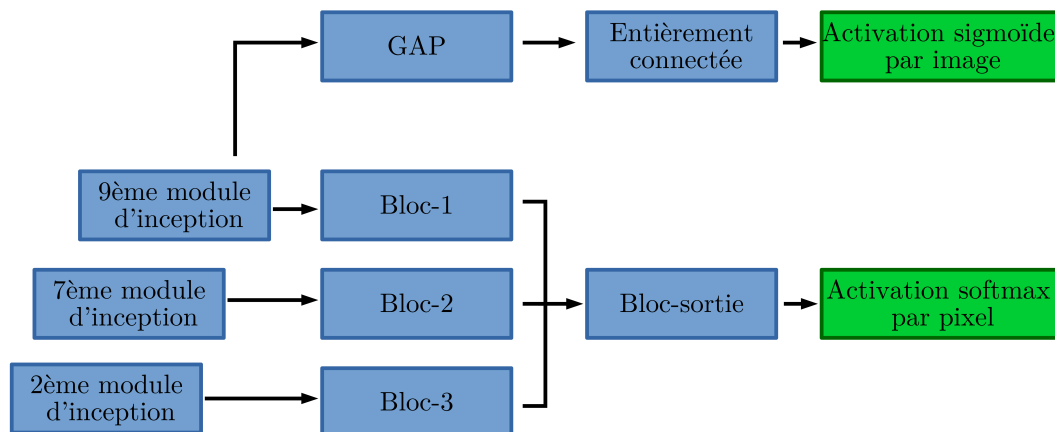


FIGURE 4.15 – Modification de l'architecture du GoogLeNet pour obtenir FCN-GoogLeNet.

Lors de l'étape de prédiction, les deux sorties du réseau sont combinées selon le diagramme de la figure 4.16. Un défaut ou maladie n'est classé que si les deux branches le détectent, c'est-à-dire si la branche de classification par image identifie la classe  $m$  et que conjointement, des pixels de cette même classe sont détectés par la branche de classification par pixels.

### 4.5.3 Classification de défauts par gravité

En suivant la méthode faiblement supervisée décrite au début de ce chapitre, les résultats de segmentation des défauts endommagés et verts sont ensuite utilisés comme entrée de deux classifieurs  $2C-SVM_{\text{Grav}}$  afin de diviser les pommes de terre endommagées et vertes par gravité. Comme nous l'avons expliqué précédemment, seuls les attributs extraits de la face de la pomme de terre dont le défaut occupe la plus grande surface relative sont pris en compte. De cette façon, chaque pomme de terre endommagée ou verte est représentée par 8 attributs :

1. Nombre de pixels endommagés/verts détectés.
2. Pourcentage de la surface de la pomme de terre détectée comme endommagée/verte.
3. Moyenne du canal R des pixels endommagés/verts détectés.
4. Moyenne du canal V des pixels endommagés/verts détectés.
5. Moyenne du canal B des pixels endommagés/verts détectés.
6. Écart-type du canal R des pixels endommagés/verts détectés.

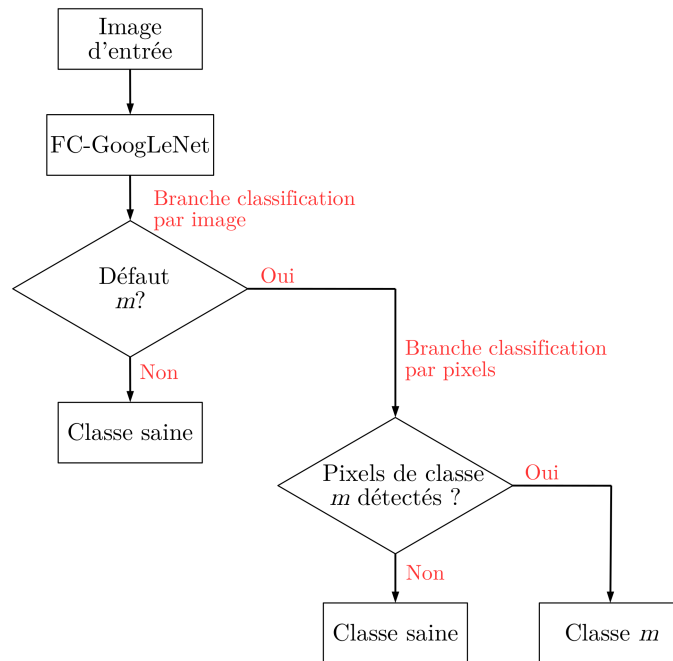


FIGURE 4.16 – Règle de décision pour la classification d’image.

7. Écart-type du canal V des pixels endommagés/verts détectés.
8. Écart-type du canal B des pixels endommagés/verts détectés.

## 4.6 Résultats expérimentaux

Nous avons initialisé le nouveau réseau FC-GoogLeNet à partir des paramètres appris par le réseau GoogLeNet entraîné pour classer nos images de pommes de terre. Nous avons ensuite appliqué du *finetuning* pour affiner uniquement les paramètres correspondant aux nouvelles couches du FC-GoogLeNet, c’est-à-dire les couches appartenant aux *Bloc-1*, *Bloc-2*, *Bloc-3* et *Bloc-sortie*. Les paramètres des couches partagées par les deux réseaux n’ont pas été modifiés afin de préserver les performances du réseau pour la tâche de classification par images. L’algorithme d’optimisation Adam [174] avec mini-lots de taille 10 a été utilisé pour entraîner le réseau. La fonction de perte d’entropie croisée a été employée pour l’étape d’entraînement. Le taux d’apprentissage des nouvelles couches a été fixé à  $1 \times 10^{-4}$ , en ce qui concerne le reste des couches, le taux d’apprentissage a été défini à 0 (*frozen layers*). L’apprentissage a été programmé pour s’arrêter automatiquement si l’erreur de validation ne diminue pas pendant 15 *epochs* afin d’éviter le sur-apprentissage.

Dans le tableau 4.7 nous comparons les résultats obtenus avec le FC-GoogLeNet et le reste des méthodes proposées tout au long de cette thèse. À partir de ce tableau nous pouvons observer que deux méthodes se partagent la valeur moyenne de  $F_1$ -mesure la plus élevée : GoogLeNet+AE+2C-SVM<sub>Dét</sub>+2C-SVM<sub>Grav</sub> (« GNet+AE+2C-SVM ») et FC-GoogLeNet+2C-SVM<sub>Grav</sub> (« FC-GNet »). Ces résultats nous indiquent que la nouvelle méthode de segmentation d’images (« FC-GNet ») atteint les performances de l’approche

supervisée malgré l'utilisation d'une base de données avec des étiquettes obtenues de manière automatique grâce à notre méthode faiblement supervisée (« Combinaison+DAM »).

TABLEAU 4.7 – Précision, rappel et  $F_1$ -mesure obtenus par les méthodes proposées sur la base de données de test. Différentes configurations de la première méthode proposée du chapitre 3 : GoogLeNet+AE+OC-SVM+2C-SVM<sub>Grav</sub> (« GNet+AE+OC-SVM ») et GoogLeNet+AE+2C-SVM<sub>Dét</sub>+2C-SVM<sub>Grav</sub> (« GNet+AE+2C-SVM »). Différentes configurations de la méthode de ce chapitre : GoogLeNet+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub> (« GNet+DAM »), GoogLeNet-Modif+OC-SVM<sub>Seg</sub>+2C-SVM<sub>Grav</sub> (« GNet-Mod+DAM ») et une combinaison des deux réseaux (nommé « Combinaison+DAM »). Dernière méthode proposée pour la segmentation d'images FC-GoogLeNet+2C-SVM<sub>Grav</sub> (« FC-GNet »). Les classes sont : S=Saine, EL=Endommagée Légère, EG=Endommagée Grave, VL=Verte Légère, VG=Verte Grave, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

Critère	Classes	GNet+ AE+ OC-SVM	GNet+ AE+ 2C-SVM	GNet+ DAM	GNet- Modif+ DAM	Combi- nation+ DAM	FC- GNet
Précision	S	<b>0,98</b>	<b>0,98</b>	<b>0,98</b>	0,95	<b>0,98</b>	0,95
	EL	0,83	0,90	0,83	0,88	0,85	<b>0,91</b>
	EG	0,83	0,86	0,79	0,84	0,91	<b>0,96</b>
	VL	0,91	0,88	0,89	0,83	<b>0,93</b>	0,76
	VG	0,89	<b>0,98</b>	0,90	0,92	0,94	0,97
	D	<b>0,92</b>	<b>0,92</b>	<b>0,92</b>	0,91	0,90	0,82
	GC	0,95	0,95	0,95	<b>0,96</b>	0,94	0,90
	R	<b>0,88</b>	<b>0,88</b>	0,84	0,78	0,81	<b>0,88</b>
	Moyenne	0,90	<b>0,92</b>	0,89	0,88	0,91	0,89
Rappel	S	0,98	0,98	0,98	<b>0,99</b>	0,98	0,98
	EL	0,92	0,94	0,91	0,87	<b>0,95</b>	0,93
	EG	0,73	0,88	0,77	0,88	0,83	<b>0,90</b>
	VL	0,67	<b>0,91</b>	0,70	0,85	0,83	0,89
	VG	<b>0,97</b>	0,95	0,96	<b>0,97</b>	<b>0,97</b>	0,89
	D	0,82	0,82	0,82	0,75	0,82	<b>0,86</b>
	CS	0,87	0,87	0,88	0,88	0,88	<b>0,93</b>
	R	0,95	0,95	0,95	0,95	0,95	<b>1,00</b>
	Moyenne	0,87	0,91	0,87	0,89	0,90	<b>0,92</b>
$F_1$ -mesure	S	<b>0,98</b>	<b>0,98</b>	<b>0,98</b>	0,97	<b>0,98</b>	0,97
	EL	0,88	<b>0,92</b>	0,86	0,88	0,90	<b>0,92</b>
	EG	0,78	0,87	0,78	0,86	0,87	<b>0,92</b>
	VL	0,78	<b>0,89</b>	0,78	0,84	0,87	0,82
	VG	0,93	<b>0,96</b>	0,93	0,95	<b>0,96</b>	0,93
	D	<b>0,87</b>	<b>0,87</b>	<b>0,87</b>	0,82	0,86	0,84
	GC	0,91	0,91	0,91	0,91	0,91	0,91
	R	0,91	0,91	0,89	0,86	0,88	<b>0,93</b>
	Moyenne	0,88	<b>0,91</b>	0,88	0,89	0,90	<b>0,91</b>

Le fait d'utiliser une telle base de données est un grand avantage pour la méthode car il n'y a pas eu d'effort humain pour étiqueter les images au niveau du pixel. Un autre avantage du « FC-GNet » est le temps de calcul réduit lors de la prédiction. En effet, dans le tableau 4.8, nous pouvons observer les temps de prédiction par image obtenus avec chacune des méthodes proposées.

« FC-GNet » est la méthode qui opère avec le temps de prédiction le plus court. La complexité calculatoire est primordiale dans un contexte industriel où les résultats doivent être obtenus le plus rapidement possible. Les deux autres méthodes avec un faible temps de prédiction sont « GNet+AE+OC-SVM » et « GNet+AE+2C-SVM ». Cependant, ces deux méthodes localisent uniquement les défauts sur les pommes de terre endommagées et vertes. Cela signifie que la localisation de défauts, qui est l'étape la plus chronophage, n'est pas effectuée sur les images classées par le CNN comme saines ou malades (dartrose, gale commune et rhizoctone). Contrairement à ces deux méthodes, l'approche faiblement supervisée fondée sur les DAMs est capable de localiser et de segmenter finement tous les défauts et maladies contenus dans la base de données. Ceci est important étant donné qu'à l'avenir, les maladies (dartrose, gale commune et rhizoctone) devraient être classées en fonction de leur gravité aussi.

Une analyse qualitative a également été effectuée afin de montrer la capacité de chaque

TABLEAU 4.8 – Temps de calcul par image pour chacune des méthodes proposées dans le contexte de cette thèse.

Méthode	Temps de calcul (sec/image)
GNet+AE+OC-SVM	0,11
GNet+AE+2C-SVM	0,10
GNet+DAM	0,46
GNet-Mod+DAM	0,41
Combinaison+DAM	0,42
FC-GNet	<b>0,08</b>

méthode à localiser ou à segmenter les défauts classés. Pour ce faire, nous présentons dans la figure 4.17 les images de sortie obtenues en utilisant chaque méthode proposée. Comme nous pouvons le voir, les méthodes « GNet+AE+OC-SVM » et « GNet+AE+2C-SVM » sont capables de localiser les défauts classés, et les méthodes « GNet+DAM », « Combinaison+DAM » et « FC-GNet » réalisent, quant à elles, une segmentation plus précise de ces défauts.

Plusieurs constatations peuvent être faites à partir des résultats obtenus :

- Les résultats de localisation obtenus avec la méthode supervisée « GNet+AE+2C-SVM » sont satisfaisants. Néanmoins, la nécessité d'utiliser une base de données étiquetée par patchs afin d'obtenir une localisation correcte des défauts est une limitation majeure de cette méthode.

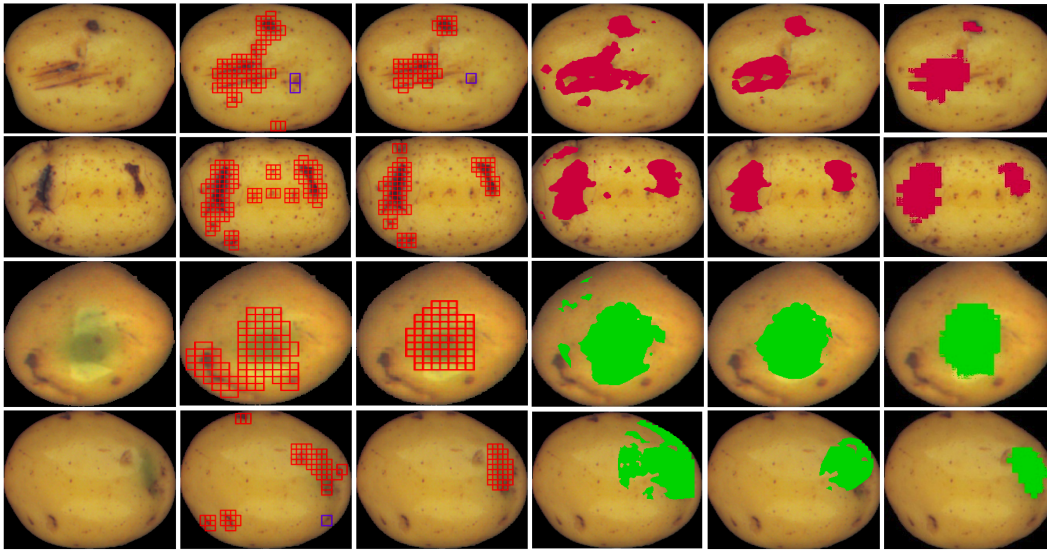


FIGURE 4.17 – Résultat de la localisation des défauts sur 4 images de test : 2 endommagements et 2 verts. De gauche à droite les résultats par méthode : image d’entrée, « GNet+AE+OC-SVM », « GNet+AE+2C-SVM », « GNet+DAM », « Combinaison+DAM » et « FC-GNet ».

- Afin de se défaire d’une base de données étiquetée par patches dans notre première méthode nous pouvons remplacer le classifieur supervisé  $2C-SVM_{Dét}$  par un classifieur non supervisé  $OC-SVM_{Dét}$  (« GNet+AE+OC-SVM »). Cependant, les résultats de localisation obtenus à partir de cette combinaison ( $OC-SVM_{Dét}$ ) sont significativement inférieurs à ceux obtenus dans le cas supervisé.
- Grâce à notre méthode faiblement supervisée (« Combinaison+DAM ») nous obtenons des résultats de segmentation de défauts satisfaisants sans recourir à une base de données étiquetée par patch ou par pixel.
- Finalement, le nouveau réseau de neurones convolutifs « FC-GNet » est capable de segmenter les défauts d’une manière précise tout en assurant une complexité calculatoire faible. Les résultats de cette méthode sont similaires à ceux obtenus avec « Combinaison+DAM », ce qui est parfaitement compréhensible : le réseau « FC-GNet » a été entraîné en utilisant les annotations par pixels générées par « Combinaison+DAM ».

## 4.7 Conclusion

Dans la première partie de ce chapitre, nous avons proposé une méthode reposant sur un apprentissage faiblement supervisé afin de classer, de localiser et de segmenter divers défauts sur la pomme de terre. En tirant partie de la capacité des CNNs à localiser les

régions des images qui sont à l'origine de la prédiction d'une classe, nous avons réussi à localiser les défauts présents dans les pommes de terre et ce, sans recourir à une base de données étiquetée au niveau du pixel. Ensuite, les défauts localisés ont été segmentés plus finement grâce à une nouvelle approche qui utilise le CNN entraîné comme extracteur de caractéristiques en conjonction avec un détecteur OC-SVM.

Grâce aux résultats expérimentaux, nous avons montré la pertinence de la méthode proposée pour résoudre notre problème de classification et de localisation de défauts sur les pommes de terre. Le fait que nous n'ayons pas besoin d'une base de données étiquetée au niveau du pixel ou du patch afin de localiser et de segmenter les défauts et maladies est l'atout important de notre méthode.

Dans la deuxième partie du chapitre, nous avons exploité les résultats de segmentation obtenus à l'aide de la méthode faiblement supervisée pour construire une base « supervisée » au niveau du pixel et entraîner une nouvelle architecture de réseau de neurones convolutifs à deux sorties (FC-GoogLeNet). Ce réseau FC-GoogLeNet a été utilisé afin d'accomplir la tâche de classification et de segmentation sémantique d'images simultanément. L'utilisation de ce nouveau réseau a deux avantages principaux : l'entraînement est réalisé de bout en bout et le temps de calcul dans la phase de prédiction est significativement plus faible que pour les autres approches. En outre, les résultats obtenus avec cette approche sont comparables à ceux obtenus avec l'approche supervisée introduite au chapitre 3, qui utilise une base de données étiquetée manuellement par patches. Enfin, nous avons comparé l'ensemble des méthodes proposées dans nos travaux de thèse, en montrant la pertinence de chacune d'entre elles, ainsi que leurs limites.

Une fois le système développé et mis en production, nous devons veiller à ce que ses performances ne se dégradent pas au fil du temps. Dans le contexte de notre application, il est possible que certaines modifications soient apportées au système d'acquisition d'images ou qu'une nouvelle variété de produit soit introduite sur le marché. Si l'une de ces deux situations se produit, les performances du modèle initialement entraîné avec la base de données originale risquent de se voir altérées. De ce fait, dans le prochain chapitre, nous étudierons en détail des méthodes couramment utilisées pour résoudre ce type de problème, connu sous le nom d'adaptation de domaine. Ensuite, nous proposerons une méthode adaptée à notre contexte.





# Chapitre 5

## Adaptation de domaine non supervisé par réseaux de neurones antagonistes

### Sommaire

---

<b>5.1</b>	<b>Introduction</b>	<b>125</b>
<b>5.2</b>	<b>Problème d'adaptation de domaine</b>	<b>126</b>
<b>5.3</b>	<b>Adaptation de domaine profonde non supervisée : état de l'art</b>	<b>128</b>
5.3.1	Les approches fondées sur la divergence	129
5.3.2	Les approches fondées sur les réseaux antagonistes	135
5.3.3	Les approches fondées sur la reconstruction	140
<b>5.4</b>	<b>Système proposé fondé sur l'apprentissage antagoniste</b>	<b>141</b>
5.4.1	Apprentissage du réseau source	143
5.4.2	Apprentissage antagoniste pour l'adaptation de domaine	143
<b>5.5</b>	<b>Résultats expérimentaux</b>	<b>145</b>
5.5.1	Bases de données utilisées	145
5.5.2	Méthodes comparatives	146
5.5.3	Détails des implémentations	148
5.5.4	Résultats sur la base de données avec un changement de luminosité	149
5.5.5	Résultats sur différentes variétés de couleur de pommes de terre	152
<b>5.6</b>	<b>Conclusion</b>	<b>155</b>

---

### 5.1 Introduction

Des méthodes de classification et de localisation reposant sur un apprentissage supervisé et faiblement supervisé ont été élaborées et testées dans les chapitres précédents. Grâce à ces travaux, nous avons montré qu'il n'est pas indispensable de disposer d'une base de données étiquetée manuellement au niveau du patch ou du pixel pour obtenir des résultats satisfaisants en terme de détection et de localisation de défauts. Une grande base de données disposant d'un étiquetage au niveau de l'image a été suffisante pour entraîner des modèles pertinents pour la classification et la localisation des défauts et des maladies

des pommes de terre. Néanmoins, il est important de souligner que ces modèles entraînés seront en mesure de garantir des performances constantes uniquement si les données analysées ont la même distribution que celles utilisées pour l'apprentissage. En d'autres termes, les images analysées doivent être prises dans les mêmes conditions (luminosité, résolution, type de la caméra) que celles utilisées pour constituer la base de données d'entraînement. Elles doivent aussi représenter la même variété de couleur de pommes de terre (par exemple variété jaune ou rouge).

Dans le monde industriel, il est fréquent de modifier certains composants de la chaîne d'acquisition (éclairage, caméra, etc.) ou la nature des produits utilisés (nouvelles variétés de pommes de terre). Ces variations peuvent engendrer un phénomène connu sous le nom de glissement de domaine (*domain shift*) [189], qui produit généralement une baisse significative des performances quand le modèle entraîné avec les images initiales (domaine source) est utilisé pour classer de nouvelles images provenant d'un domaine différent (domaine cible).

Le *finetuning* appliqué à un réseau pré-entraîné constitue une solution potentielle à ce problème en utilisant une nouvelle base de données étiquetée incluant des images provenant du domaine cible. Cependant, la création d'une telle base de données est très laborieuse voire impossible dans certains cas compte tenu de la difficulté de regrouper des exemples représentatifs de certaines classes ciblées (par exemple, certains types de défauts peu fréquents). Ainsi, différentes approches d'adaptation de domaine ont été proposées par la communauté scientifique afin d'adapter des modèles entraînés à une nouvelle distribution de données cibles sans pour autant devoir créer une nouvelle base de données étiquetées. L'idée principale de ces méthodes consiste à exploiter les images annotées provenant du domaine source afin de construire un nouveau modèle avec peu ou aucune donnée étiquetée du domaine cible.

L'adaptation de domaine est une problématique récurrente en vision par ordinateur [8, 9, 11, 190, 10, 191], mais également dans de nombreuses autres tâches d'apprentissage automatique telles que le traitement du langage naturel [192, 193, 194] ou la reconnaissance de la parole [195, 196, 197]. Pour les besoins de nos travaux, nous allons nous focaliser sur les méthodes d'adaptation de domaine appliquées à la vision par ordinateur. Le chapitre est organisé comme suit : tout d'abord nous allons présenter le cadre théorique de l'adaptation de domaine puis exposer les principales approches de la littérature. Une nouvelle méthode d'adaptation de domaine pour la classification de pommes de terre est ensuite introduite. Une analyse comparant les résultats obtenus par la méthode proposée à ceux des principales méthodes de la littérature est menée et permet de conclure le chapitre.

## 5.2 Problème d'adaptation de domaine

Le problème de l'adaptation de domaine est un cas particulier du transfert d'apprentissage qui tire partie de données étiquetées dans un ou plusieurs domaines source afin d'accomplir une nouvelle tâche dans un nouveau domaine cible.

Supposons un domaine source  $\mathcal{D}^s = \{\mathcal{X}^s, P_s\}$  composé d'un espace de caractéristiques  $\mathcal{X}^s$ , dans lequel des observations  $X^s = \{x_1^s, \dots, x_{n_s}^s\}$  sont distribuées selon la loi  $P_s$  et  $\mathcal{T}^s = \{\mathcal{Y}^s, P(Y^s|X^s)\}$  une tâche source.  $P(Y^s|X^s)$  est la distribution des étiquettes conditionnellement aux données  $X^s$  et  $\mathcal{Y}^s$  l'ensemble d'étiquettes correspondant aux données appartenant à l'espace  $\mathcal{X}^s$ .

On considère également un second domaine, nommé cible,  $\mathcal{D}^c = \{\mathcal{X}^c, P_c\}$ , ainsi qu'une tâche cible  $\mathcal{T}^c = \{\mathcal{Y}^c, P(Y^c|X^c)\}$ , où  $\mathcal{X}^c$  est un nouvel espace de caractéristiques et  $\mathcal{Y}^c$  l'espace d'étiquettes associé. Dans le cas où les deux domaines sont équivalents, c'est-à-dire  $\mathcal{D}^s = \mathcal{D}^c$  et  $\mathcal{T}^s = \mathcal{T}^c$ , des approches d'apprentissage automatique traditionnelles peuvent être appliquées. En effet, l'ensemble d'observations appartenant au domaine source  $\mathcal{D}^s$  sera utilisé comme l'ensemble d'apprentissage et l'ensemble d'observations du domaine cible  $\mathcal{D}^c$  comme ensemble de test.

Dans le cas où les tâches de source et cible sont les mêmes ( $\mathcal{T}^s = \mathcal{T}^c$ ), mais s'il existe une divergence entre domaines en raison d'un décalage dans les distributions ( $P_s \neq P_c$ ) ou d'un changement dans les espaces de caractéristiques ( $\mathcal{X}^s \neq \mathcal{X}^c$ ) nous nous retrouvons dans une situation où l'adaptation de domaine devient nécessaire [198]. Une première catégorisation de ces problèmes est donc faite par rapport au type de divergence existant. En cas de décalage uniquement entre les distributions  $P_s \neq P_c$  le problème d'adaptation est dit homogène. Dans le cas où les espaces de caractéristiques des deux domaines ne sont pas équivalents  $\mathcal{X}^s \neq \mathcal{X}^c$ , le problème d'adaptation est qualifié d'hétérogène.

Nous pouvons également différencier les méthodes d'adaptation de domaine selon les informations fournies par la base de données cible :

- Adaptation de domaine supervisée : c'est le cas le plus simple car les étiquettes de la base de données cible sont disponibles. Dans ce contexte, nous pouvons directement utiliser du *finetuning* afin d'affiner les paramètres d'un réseau entraîné initialement avec la base de données provenant du domaine source.
- Adaptation de domaine semi-supervisée : dans ce contexte, bien que la plupart des données cibles ne soient pas étiquetées, il existe quelques observations avec étiquettes. Cette situation est plus complexe que le cas précédent car une grande partie des données cibles sont sans étiquette.
- Adaptation de domaine non supervisée : c'est la situation la plus complexe due au fait qu'aucun échantillon de la base de données cible n'est étiqueté. Dans cette situation, seules les étiquettes de la base de données source sont utilisées afin de réaliser l'adaptation du modèle.

Pour ce qui concerne ce travail de thèse, deux considérations vont nous guider dans l'exploration des solutions au problème de l'adaptation de domaine :

1. Dans le cadre de notre application, l'espace de représentation des données n'est pas censé évoluer, ni subir de transformation. De ce fait nous allons focaliser notre at-

tention sur les approches d'adaptation de domaine dites homogènes ( $\mathcal{X}^s = \mathcal{X}^c$ ).

2. Dans les situations que nous avons évoquées en introduction de ce chapitre, il n'est pas raisonnable d'envisager de procéder à une nouvelle phase d'apprentissage à chaque fois que la distribution des données subira une transformation (due aux variations des conditions d'acquisition des données d'une machine à l'autre ou à l'introduction de nouvelles variétés du produit à analyser). Nous allons donc nous placer dans le contexte des approches d'adaptation non supervisée.

Dans ce contexte, notre objectif va donc être d'adapter directement le CNN initialement entraîné avec nos images sources, afin d'obtenir une nouvelle règle de décision simple et rapide ne nécessitant pas l'étiquetage d'un ensemble de nouvelles images cibles.

### 5.3 Adaptation de domaine profonde non supervisée : état de l'art

Comme cela a été mentionné dans les chapitres précédents, les méthodes reposant sur l'apprentissage profond, notamment les réseaux de neurones convolutifs, ont bouleversé le domaine de l'apprentissage automatique. À travers diverses applications, ces méthodes ont montré leur supériorité face aux méthodes fondées sur l'extraction ciblée de caractéristiques. Cependant, la majorité de ces méthodes nécessitent une grande quantité de données étiquetées afin d'être exploitées avec succès.

Le monde industriel est soumis à une évolution systématique des équipements et des produits. En effet, les évolutions du marché ainsi que la nature hostile des environnements industriels créent des transformations, aussi bien sur les produits à analyser que sur les systèmes d'acquisition des données. Ce contexte évolutif contraint les solutions de traitement de données à être adaptatives pour suivre ces évolutions. Pour répondre efficacement à ces exigences, de nombreux travaux ont été menés afin de proposer des solutions permettant aux réseaux de neurones entraînés sur des données d'un domaine source de s'adapter aux données provenant d'un domaine cible différent. Selon Wang et Deng [199], ces approches peuvent être divisées en trois catégories :

- Les approches reposant sur la divergence (*discrepancy-based methods*), qui utilisent différentes techniques afin d'aligner les distributions entre des domaines source et cible. Parmi les techniques les plus connues nous retrouvons des approches visant à modifier les distributions source et cible ( $P_s$  et  $P_c$ ) afin d'optimiser des critères associés à la différence entre les distributions  $P_s$  et  $P_c$  transformées. Nous pouvons notamment citer l'écart moyen maximal (*Maximum Mean Discrepancy* - MMD) [8, 9, 190, 10] et l'alignement par corrélation (*Deep CORAL*) [11].
- Les approches fondées sur les réseaux antagonistes (*adversarial-based methods*), qui utilisent l'apprentissage antagoniste dans l'objectif d'apprendre des caractéristiques des domaines source et cible qui soient indistinguables. Des modèles génératifs

[12, 13, 200] et discriminatoires [14, 191, 201, 202, 203] sont inclus dans ces approches.

- Les approches fondées sur la reconstruction (*reconstruction-based methods*), qui utilisent la reconstruction de la donnée source ou cible comme tâche auxiliaire afin de créer des caractéristiques invariantes au domaine [15, 204, 205, 206].

Dans les sections suivantes nous allons voir en détail chacune de ces approches.

### 5.3.1 Les approches fondées sur la divergence

Les méthodes d'adaptation de domaine fondées sur la divergence cherchent à minimiser l'écart entre les distributions des deux domaines, source et cible. L'écart moyen maximal (*Maximum Mean Discrepancy* - MMD) [207] est couramment utilisé afin de minimiser cet écart.

Étant donnée  $X^s = \{x_1^s, \dots, x_{n_s}^s\} \in \mathcal{X}^s$  des observations de domaine source avec une distribution de probabilité  $P_s$ ,  $X^c = \{x_1^c, \dots, x_{n_c}^c\} \in \mathcal{X}^c$  des observations de domaine cible avec une distribution de probabilité  $P_c$ , et  $\mathcal{F}$  un espace de fonctions  $f : \mathcal{X} \rightarrow \mathbb{R}$ , où  $\mathcal{X} = \mathcal{X}^s = \mathcal{X}^c$ , l'écart moyen maximal (MMD) est défini comme suit :

$$\text{MMD}[\mathcal{F}, P_s, P_c] = \sup_{f \in \mathcal{F}} (\mathbb{E}_{P_s}[f(x^s)] - \mathbb{E}_{P_c}[f(x^c)]) \quad (5.1)$$

L'estimateur de la mesure MMD est obtenu en remplaçant  $\mathbb{E}_{P_s}[f(x^s)]$  et  $\mathbb{E}_{P_c}[f(x^c)]$  par les moyennes empiriques calculées à partir des observations  $X^s$  et  $X^c$  :

$$\text{MMD}_b[\mathcal{F}, X^s, X^c] = \sup_{f \in \mathcal{F}} \left( \frac{1}{n_s} \sum_{i=1}^{n_s} f(x_i^s) - \frac{1}{n_c} \sum_{j=1}^{n_c} f(x_j^c) \right) \quad (5.2)$$

À partir de l'équation 5.2 nous pouvons observer que si les distributions des deux domaines sont similaires, l'écart moyen maximal MMD sera petit. Dans le cas contraire, il existera des fonctions  $f$  telles que l'écart des moyennes sera grand. La robustesse du MMD dépend de l'espace de fonctions  $\mathcal{F}$  utilisé. En effet, Gretton et al. [207] ont démontré que  $\mathcal{F}$  doit être d'une part « suffisamment riche » pour pouvoir identifier si  $P_s = P_c$  et d'une autre part, assez « restrictif » pour que l'estimateur de MMD converge rapidement vers son espérance au fur et à mesure que la taille de l'échantillon augmente. Par conséquent, pour respecter ces deux conditions, les auteurs ont proposé d'utiliser comme espace de fonctions  $\mathcal{F}$  la boule unité dans un espace de Hilbert à noyau reproduisant  $\mathcal{H}$  (*Reproducing Kernel Hilbert Space* - RKHS), c'est-à-dire  $\mathcal{F} = \{f \mid \|f\|_{\mathcal{H}} \leq 1\}$ .

L'espace de Hilbert  $\mathcal{H}$  de fonctions définies de  $\mathcal{X}$  vers  $\mathbb{R}$  est dit un espace de Hilbert à noyau reproduisant [208] si et seulement si  $\forall x \in \mathcal{X}$  la fonctionnelle d'évaluation  $\delta_x : \mathcal{H} \rightarrow \mathbb{R}$ , définie comme  $\delta_x(f) = f(x)$ , est continue. Ainsi, d'après le théorème de Représentation de Riesz [209],  $\forall x \in \mathcal{X}$  il existe un unique élément  $K_x \in \mathcal{H}$  tel que :

$$f(x) = \langle f(\cdot), K_x(\cdot) \rangle_{\mathcal{H}} \quad \forall f \in \mathcal{H} \quad (5.3)$$

Étant donné que  $K_x \in \mathcal{H}$  est lui-même une fonction définie sur  $\mathcal{X}$ , nous avons :

$$K_x(x') = \langle K_x(\cdot), K'_x(\cdot) \rangle_{\mathcal{H}} \quad (5.4)$$

Nous pouvons ainsi définir le noyau reproduisant sur  $\mathcal{H}$ ,  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  comme suit :

$$K(x, x') = \langle K(x, \cdot), K(x', \cdot) \rangle_{\mathcal{H}} = \langle K_x(\cdot), K'_x(\cdot) \rangle_{\mathcal{H}} \quad (5.5)$$

D'après les notions décrites ci-dessus, nous pouvons représenter les lois de probabilités  $P_s$  et  $P_c$  dans un RKHS  $\mathcal{H}$  en employant les opérateurs suivants [210] :

$$\begin{aligned} \mu_{P_s} &= \mathbb{E}_{P_s}[K(x^s, \cdot)] \\ \mu_{P_c} &= \mathbb{E}_{P_c}[K(x^c, \cdot)] \end{aligned} \quad (5.6)$$

où  $\mu_{P_s}$  et  $\mu_{P_c}$  sont connus sous le nom de *mean embeddings* et leurs équivalents empiriques sont donnés par :

$$\begin{aligned} \mu_{X_s} &= \frac{1}{n_s} \sum_{i=1}^{n_s} K(x_i^s, \cdot) \\ \mu_{X_c} &= \frac{1}{n_c} \sum_{i=1}^{n_c} K(x_i^c, \cdot) \end{aligned} \quad (5.7)$$

Si la condition (suffisante)  $\mathbb{E}_{P_s}[K(x, x)] < \infty$  est remplie, alors  $\mu_{P_s}$  est un élément de l'espace  $\mathcal{H}$  et, en utilisant la propriété de l'équation 5.3, nous avons [210] :

$$\langle f, \mu_{P_s} \rangle_{\mathcal{H}} = \mathbb{E}_{P_s}[f(x^s)] \quad (5.8)$$

$$\langle f, \mu_{X_s} \rangle_{\mathcal{H}} = \frac{1}{n_s} \sum_{i=1}^{n_s} f(x_i^s) \quad (5.9)$$

Autrement dit, nous pouvons calculer l'espérance et la moyenne empirique par rapport à  $P_s$  et  $X^s$  respectivement, en prenant les produits scalaires avec les moyennes dans le RKHS,  $\mu_{P_s}$  et  $\mu_{X_s}$  (démarche équivalente pour le domaine cible  $\mu_{P_c}$  et  $\mu_{X_c}$ ).

Les auteurs Gretton et al. [207] ont ainsi démontré que la mesure MMD de l'équation 5.1 peut être exprimée comme la distance dans  $\mathcal{H}$  entre les *mean embeddings* :

$$\begin{aligned} \text{MMD}^2[\mathcal{F}, P_s, P_c] &= \left[ \sup_{\|f\|_{\mathcal{H}} \leq 1} (\mathbb{E}_{P_s}[f(x^s)] - \mathbb{E}_{P_c}[f(x^c)]) \right]^2 \\ &= \left[ \sup_{\|f\|_{\mathcal{H}} \leq 1} \langle f, \mu_{P_s} - \mu_{P_c} \rangle_{\mathcal{H}} \right]^2 = \|\mu_{P_s} - \mu_{P_c}\|_{\mathcal{H}}^2 \end{aligned} \quad (5.10)$$

Ce qui peut être réécrit comme suit :

$$\begin{aligned}
 \text{MMD}^2[\mathcal{F}, P_s, P_c] &= \langle \mu_{P_s} - \mu_{P_c}, \mu_{P_s} - \mu_{P_c} \rangle_{\mathcal{H}} \\
 &= \langle \mu_{P_s}, \mu_{P_s} \rangle_{\mathcal{H}} + \langle \mu_{P_c}, \mu_{P_c} \rangle_{\mathcal{H}} - 2 \langle \mu_{P_s}, \mu_{P_c} \rangle_{\mathcal{H}} \\
 &= \mathbb{E}_{P_s} [K(x^s, x'^s)] + \mathbb{E}_{P_c} [K(x^c, x'^c)] - 2\mathbb{E}_{P_s, P_c} [K(x^s, x^c)]
 \end{aligned} \tag{5.11}$$

Puis, un estimateur non biaisé de  $\text{MMD}^2[\mathcal{F}, P_s, P_c]$  est donné par :

$$\begin{aligned}
 \text{MMD}_u^2[\mathcal{F}, X^s, X^c] &= \frac{1}{n_s(n_s - 1)} \sum_{i=1}^{n_s} \sum_{j \neq i}^{n_s} K(x_i^s, x_j^s) + \frac{1}{n_c(n_c - 1)} \sum_{i=1}^{n_c} \sum_{j \neq i}^{n_c} K(x_i^c, x_j^c) \\
 &\quad - \frac{2}{n_s n_c} \sum_{i=1}^{n_s} \sum_{j=1}^{n_c} K(x_i^s, x_j^c)
 \end{aligned} \tag{5.12}$$

Gretton et al. [207] ont également introduit une variante biaisée ( $\text{MMD}_b^2$ ) de cet estimateur :

$$\begin{aligned}
 \text{MMD}_b^2[\mathcal{F}, X^s, X^c] &= \frac{1}{n_s^2} \sum_{i,j=1}^{n_s} K(x_i^s, x_j^s) + \frac{1}{n_c^2} \sum_{i,j=1}^{n_c} K(x_i^c, x_j^c) \\
 &\quad - \frac{2}{n_s n_c} \sum_{i=1}^{n_s} \sum_{j=1}^{n_c} K(x_i^s, x_j^c)
 \end{aligned} \tag{5.13}$$

Parmi les auteurs qui ont proposé d'utiliser la mesure MMD dans le contexte d'adaptation de domaine, nous pouvons citer Tzeng et al. [8]. Dans leur travaux, deux réseaux sont entraînés en parallèle, un pour chaque domaine source et cible, avec partage des poids (voir figure 5.1). L'objectif est d'apprendre une représentation de données qui est à la fois discriminante et invariante au domaine. L'entraînement des réseaux a été fait en combinant deux fonctions de perte : la première, définie par l'entropie croisée, a été utilisée afin de classer correctement les données source étiquetées. La deuxième, estimant l'écart moyen maximal (MMD), a été introduite afin de minimiser la distance entre les distributions des deux domaines, source et cible.

En se basant sur la figure 5.1, la sortie de l'avant dernière couche entièrement connectée du CNN (nommée par les auteurs *fc\_adapt*) est utilisée afin d'obtenir les représentations des données source et cible. Ces représentations sont ensuite utilisées pour calculer la distance entre les distributions source et cible à partir de la mesure MMD (*Maximum Mean Discrepancy*). Simultanément, les observations étiquetées provenant du domaine source sont utilisées pour calculer la perte relative à la classification.

Les auteurs ont proposé de minimiser conjointement l'erreur de classification des données sources et la distance entre les distributions sources et cibles comme suit :

$$\mathcal{J}_{total} = \mathcal{J}_{class}(X^s, Y^s) + \lambda_{MMD}(\text{MMD}_b^2(X^s, X^c)) \tag{5.14}$$

où  $\text{MMD}_b^2(X^s, X^c)$  est calculé à partir de l'équation 5.2 au carré,  $\mathcal{J}_{class}(X^s, Y^s)$  représente la perte d'entropie croisée relative à la classification des données source étiquetées, et  $\lambda_{MMD}$  l'hyper-paramètre à définir qui contrôle la confusion entre domaines.



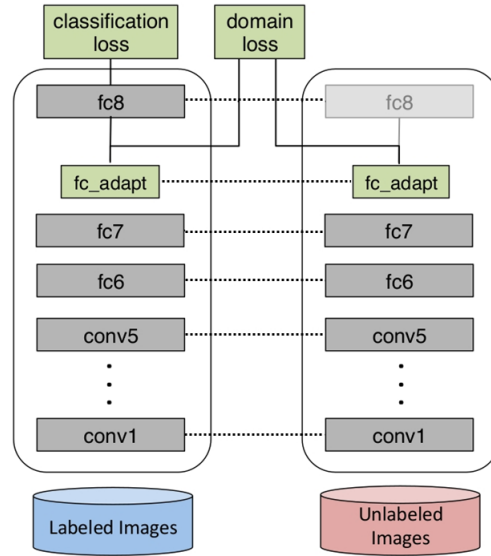


FIGURE 5.1 – Architecture des réseaux proposées par Tzeng et al. [8] afin de résoudre le problème d’adaptation de domaine. Les deux réseaux sont entraînés avec les poids partagés.

Au lieu de mesurer la divergence des domaines à la sortie d’une seule couche, Long et al. [9] ont introduit une nouvelle architecture nommée DAN (*Domain Adaptation Network*). Considérant le fait qu’une perte de transférabilité se produit généralement dans les couches plus profondes du CNN, les auteurs ont proposé de minimiser la divergence des domaines en utilisant les représentations extraites des trois dernières couches entièrement connectées (voir figure 5.2). Une variante de la mesure MMD, connue sous le nom de MK-MMD (*Multiple Kernel Maximum Mean Discrepancy*), a été introduite utilisant une combinaison convexe de  $m$  noyaux semi-définis positifs pour calculer le MMD.

Enfin, en combinant une perte de classification des données source et la perte relative à la divergence de domaines MK-MMD, les auteurs ont entraîné le CNN pour minimiser la fonction de perte totale suivante :

$$\mathcal{J}_{total} = \mathcal{J}_{class}(X^s, Y^s) + \lambda_{MK-MMD} \sum_{l=l_1}^{l_2} (\text{MK-MMD}_b^2(X^s, X^c)) \quad (5.15)$$

où  $\mathcal{J}_{class}(X^s, Y^s)$  représente la perte d’entropie croisée relative à la classification des données source étiquetées,  $l_1$  et  $l_2$  sont les indices de couches entre lesquelles l’adaptation est effectuée,  $\lambda_{MK-MMD}$  l’hyper-paramètre de pondération à définir et  $\text{MK-MMD}_b^2(X^s, X^c)$  l’estimateur de complexité linéaire<sup>1</sup> de  $\text{MMD}^2(P_s, P_c)$  proposé par Gretton et al.[211] et donné par :

1. Complexité en temps  $\mathcal{O}(n^s)$ . L’estimateur de l’équation 5.13 a une complexité en temps quadratique  $\mathcal{O}((n^s + n^c)^2)$

$$\text{MK-MMD}_b^2(X_s, X_c) = \frac{2}{n_s} \sum_{i=1}^{n_s/2} K(x_{2i-1}^s, x_{2i}^s) + K(x_{2i-1}^c, x_{2i}^c) - K(x_{2i-1}^s, x_{2i}^c) - K(x_{2i}^s, x_{2i-1}^c) \quad (5.16)$$

où nous pouvons assurer  $n_s = n_c$  en utilisant par exemple la descente de gradient par mini-lots pour entraîner le réseau (chaque mini-lot doit être constitué du même nombre d'observations sources et cibles).

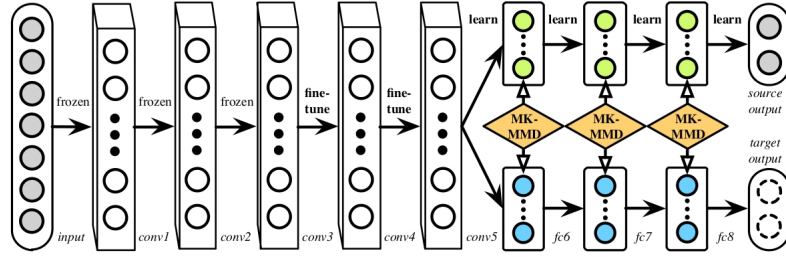


FIGURE 5.2 – Architecture de DAN proposées par Long et al. [9] afin d'adapter les représentations générées par plusieurs couches. Les réseaux sont entraînés avec partage de poids.

Afin de prendre en considération les décalages possibles dans la distribution jointe des caractéristiques d'entrée et des étiquettes de sortie, c'est-à-dire quand  $P(X^s, Y^s) \neq P(X^c, Y^c)$ , les auteurs Long et al. [10] ont introduit une nouvelle architecture de CNN, nommé *Joint Adaptation Networks* (JAN). L'idée clé de cette méthode est d'aligner la distribution jointe des caractéristiques/étiquettes au lieu de considérer uniquement les distributions  $P_s$  et  $P_c$ . L'adaptation de domaine dans cette nouvelle architecture a été réalisée

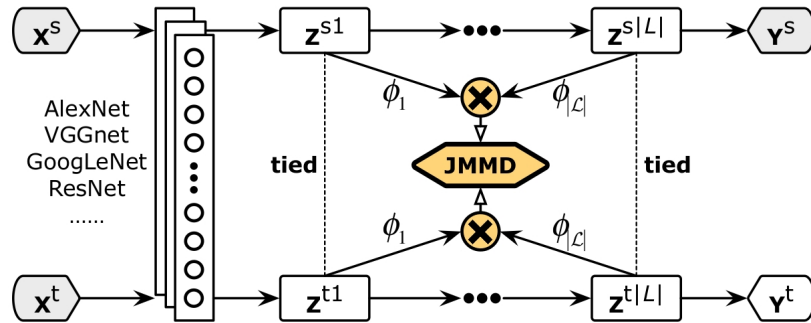


FIGURE 5.3 – Architecture de JAN proposée par Long et al. [10]. Le CNN source et le CNN cible partagent leurs poids.

en rajoutant une fonction de perte reposant sur l'écart moyen maximal joint (*Joint Maximum Mean Discrepancy* - JMMMD). En s'appuyant sur la figure 5.3, les sorties générées par deux réseaux de  $L$  couches avec partage de poids sont définies par  $\{(z_i^{s1}, \dots, z_i^{sL})\}_{i=1}^{n_s}$  et  $\{(z_i^{c1}, \dots, z_i^{cL})\}_{i=1}^{n_c}$  respectivement pour les données de source et cible. À partir de l'estimateur biaisé de MMD (voir équation 5.13), les auteurs ont introduit l'estimateur du

JMMD de la manière suivante :

$$\begin{aligned} \text{JMMD}_b^2(X^s, X^c) &= \frac{1}{n_s^2} \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} \prod_{l \in L} K^l(z_i^{sl}, z_j^{sl}) \\ &+ \frac{1}{n_c^2} \sum_{i=1}^{n_c} \sum_{j=1}^{n_c} \prod_{l \in L} K^l(z_i^{cl}, z_j^{cl}) \\ &- \frac{2}{n_s n_c} \sum_{i=1}^{n_s} \sum_{j=1}^{n_c} \prod_{l \in L} K^l(z_i^{sl}, z_j^{cl}) \end{aligned} \quad (5.17)$$

où  $K^l$  est la fonction noyau correspondant à la couche  $l$ . Inspirés de l'estimateur de complexité en temps linéaire proposé par Gretton et al. [211] (voir équation 5.16), un estimateur de complexité linéaire du JMMD a été utilisé dans l'entraînement du réseau :

$$\begin{aligned} \text{JMMD}_b^2(X^s, X^c) &= \frac{2}{n_s} \sum_{i=1}^{n_s/2} \left( \prod_{l \in L} K^l(z_{2i-1}^{sl}, z_{2i}^{sl}) + \prod_{l \in L} K^l(z_{2i-1}^{cl}, z_{2i}^{cl}) \right) \\ &- \frac{2}{n_s} \sum_{i=1}^{n_s/2} \left( \prod_{l \in L} K^l(z_{2i-1}^{sl}, z_{2i}^{cl}) + \prod_{l \in L} K^l(z_{2i-1}^{cl}, z_{2i}^{sl}) \right) \end{aligned} \quad (5.18)$$

Finalement, la fonction de perte totale utilisée pour entraîner les réseaux est donnée par :

$$\mathcal{J}_{total} = \mathcal{J}_{class}(X^s, Y^s) + \lambda_{JMMD}(\text{JMMD}_b^2(X^s, X^c)) \quad (5.19)$$

où  $\mathcal{J}_{class}(X^s, Y^s)$  représente la perte d'entropie croisée relative à la classification des données source étiquetées et  $\lambda_{JMMD} > 0$  est un hyper-paramètre qui régule l'importance de l'adaptation effectuée.

Les méthodes décrites précédemment sont principalement fondées sur la mesure MMD pour réduire l'écart de distribution entre domaines. Cette mesure nécessite le choix de noyau, ce qui reste toujours problématique. Toutefois, d'autres mesures peuvent également être utilisées pour mesurer la divergence entre les distributions. Les auteurs Sun et al. [11] ont proposé de minimiser la distance entre les matrices de covariance calculées à partir des caractéristiques extraites des données source et cible (*CORAL Loss*). En effet, comme nous pouvons le voir sur la figure 5.4, un CNN a été entraîné de bout en bout à partir de deux fonctions perte.

La première, donnée par l'entropie croisée, a été utilisée afin de mesurer la perte de classification des données source étiquetées (*classification loss*). La deuxième fonction perte, nommée «CORAL» par les auteurs, a été utilisée afin de réaliser l'adaptation de domaine (*CORAL loss*). Cette dernière fonction est donnée par l'équation suivante :

$$\mathcal{J}_{CORAL} = \frac{1}{4e^2} \|\text{Cov}^s - \text{Cov}^c\|_F^2 \quad (5.20)$$

où  $\|\cdot\|_F^2$  indique la norme de Frobenius carrée,  $\text{Cov}^s$  et  $\text{Cov}^c$  sont les matrices de covariance des vecteurs de caractéristiques de dimension  $e$  correspondant aux données source et cible

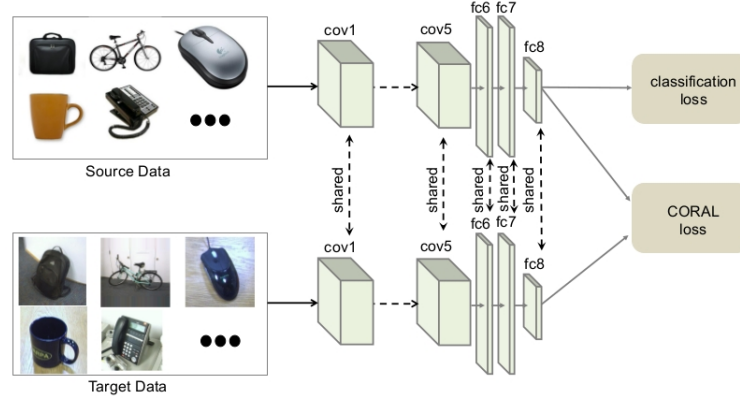


FIGURE 5.4 – Architecture de CORAL proposées par Sun et al. [11].

respectivement.

Finalement, le réseau a été entraîné en combinant les deux fonctions perte, l'entropie croisée  $\mathcal{J}_{class}(X^s, Y^s)$  et  $\mathcal{J}_{CORAL}$  :

$$\mathcal{J}_{totalCORAL} = \mathcal{J}_{class}(X^s, Y^s) + \lambda_{CORAL} \mathcal{J}_{CORAL} \quad (5.21)$$

où  $\lambda_{CORAL}$  permet de compenser la perte de classification ( $\mathcal{J}_{class}$ ) et la perte relative à l'adaptation ( $\mathcal{J}_{CORAL}$ ).

Bien que d'excellents résultats aient été obtenus avec les méthodes détaillées ci-dessus, la majorité repose sur le fait que le classifieur initialement entraîné avec les données source peut être appliqué directement aux images cibles. Cependant, lorsque les distributions jointes des caractéristiques et de l'étiquette ne sont pas identiques d'un domaine à l'autre ( $P(X^s, Y^s) \neq P(X^c, Y^c)$ ), l'adaptation des représentations de chaque domaine peut ne pas suffire [212, 203].

### 5.3.2 Les approches fondées sur les réseaux antagonistes

De nouvelles approches reposant sur les réseaux antagonistes génératifs (*Generative Adversarial Networks* - GAN) ont été récemment proposées afin de résoudre le problème d'adaptation de domaine non supervisé. Ces approches sont généralement divisés en deux étapes. Tout d'abord, l'apprentissage antagoniste est utilisé à la fois pour obtenir des représentations non dépendantes du domaine d'origine (source ou cible), mais également pour faire en sorte que ces représentations soient suffisamment discriminatives pour entraîner avec succès un classifieur avec des échantillons du domaine source étiquetés. Dans un second temps et après l'adaptation réalisée, ce classifieur entraîné est utilisé pour prédire les classes des images cibles.

Nous allons présenter par la suite brièvement les concepts théoriques des réseaux antagonistes génératifs. Nous nous focaliserons ensuite sur les différents travaux de la littérature qui utilisent ces réseaux dans le contexte de l'adaptation de domaine.

## Les réseaux antagonistes génératifs

Le GAN, initialement proposée par Goodfellow et al. [213], consiste en une combinaison de deux modèles placés en compétition : un génératif et un discriminatif. Le premier a pour objectif de générer des images ressemblant aux images d'apprentissage, et le deuxième, vise à distinguer les images d'apprentissage de celles générées par le premier modèle. Considérons par exemple  $G$  et  $D$  deux perceptrons multicouches qui représentent respectivement les modèles génératif et discriminatif,  $x$  une observation issue d'une distribution  $P_X$  et  $z$  un vecteur aléatoire issu d'une distribution  $P_z$ . Le vecteur aléatoire  $z$  est utilisé comme entrée du réseau  $G$  (générateur), lequel produit une image de sortie (de la même dimension que  $x$ ) donnée par  $G(z; \theta_G)$ , avec  $\theta_G$  l'ensemble de paramètres du réseau  $G$ . La sortie du discriminateur  $D$  estime la probabilité qu'une observation soit issue d'une distribution  $P_X$ , c'est-à-dire qu'elle soit réelle et pas générée. La sortie du discriminateur sera idéalement égale à 1 si  $x \sim P_X$  et égal à 0 si  $x \sim P_G$ , avec  $P_G$  la distribution de  $G(z; \theta_G)$ . La figure 5.5 donne une illustration de ce réseau.

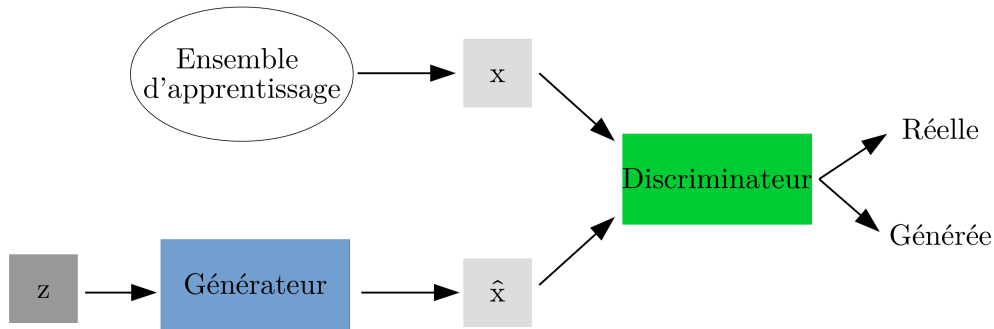


FIGURE 5.5 – Illustration du GAN.

Dans l'étape d'apprentissage, le générateur est entraîné pour produire des images convaincantes afin de leurrer le discriminateur et, en même temps, ce dernier vise à différencier les images réelles de celles générées par  $G$ . En effet, l'entraînement du GAN peut être vu comme un jeu mini-max pour deux joueurs (*minimax two-player game*) :

$$\max_G \min_D V(D, G) = \mathbb{E}_{x \sim P_X} [-\log(D(x))] + \mathbb{E}_{z \sim P_z} [-\log(1 - D(G(z)))] \quad (5.22)$$

Afin de résoudre le problème d'optimisation de l'équation 5.22, les paramètres du générateur et du discriminateur sont mis à jour de manière itérative en deux étapes :

1. Première étape :  $\theta_D^{t+1} = \theta_D^t - \alpha_D \nabla_{\theta_D^t} V(D^t, G^t)$
2. Deuxième étape :  $\theta_G^{t+1} = \theta_G^t + \alpha_G \nabla_{\theta_G^t} V(D^{t+1}, G^t)$

avec  $t$  l'indice d'itération, et  $\alpha_D$ ,  $\alpha_G$  les taux d'apprentissage pour le discriminateur et le générateur respectivement. L'objectif final de cet apprentissage est que la distribution  $P_G$  converge vers  $P_X$ .

Dans le contexte de l'adaptation de domaine, les approches reposant sur le principe des GANs peuvent être divisées en deux catégories : les modèles génératifs et les modèles discriminatifs, dont les détails suivent.

### Les modèles génératifs

Les approches fondées sur les modèles génératifs utilisent la génération d'images comme une tâche auxiliaire pour adapter les réseaux. Une nouvelle architecture nommée *Coupled Generative Adversarial Networks* (CoGAN) a été proposée par Liu et al. [12]. Comme on peut le voir dans la figure 5.6, le CoGAN est composé par deux GANs. Le premier réseau ( $\text{GAN}_1$ ) est entraîné avec des images du domaine source et le deuxième ( $\text{GAN}_2$ ) est quant à lui entraîné avec des images du domaine cible. Durant la phase d'apprentissage, un vecteur aléatoire  $z$  est pris comme entrée par les générateurs  $g_1$  source et  $g_2$  cible afin d'obtenir comme sortie deux images de même taille que les images source et cible. Puis, deux discriminateurs  $f_1$  et  $f_2$  (un pour chaque domaine) sont utilisés afin d'identifier les images réelles de celles générées. À titre d'exemple,  $f_1(x)$  sera idéalement égal à 1 si  $x \sim P_s$  et égal à 0 si  $x \sim P_{g_1}$ , avec  $P_{g_1}$  la distribution de  $g_1(z)$ . L'objectif de cet apprentissage est que la distribution de  $P_{g_1}$  s'approche de celle des données sources  $P_s$  et que la distribution  $P_{g_2}$  s'approche à celle des données cibles  $P_c$ . Les poids des premières couches des générateurs et dernières couches des discriminateurs sont liés dans le but d'apprendre un espace dans lequel les données sources et cibles partagent la même distribution. Ainsi, la partition opérée par un classifieur source (ajouté à la sortie de la dernière couche cachée du discriminateur source) peut être la même pour les observations issues de la distribution source et la distribution cible. En effet, cette partition est apprise pour minimiser la perte de classification sur les données sources étiquetées et peut ensuite s'appliquer aux données cibles.

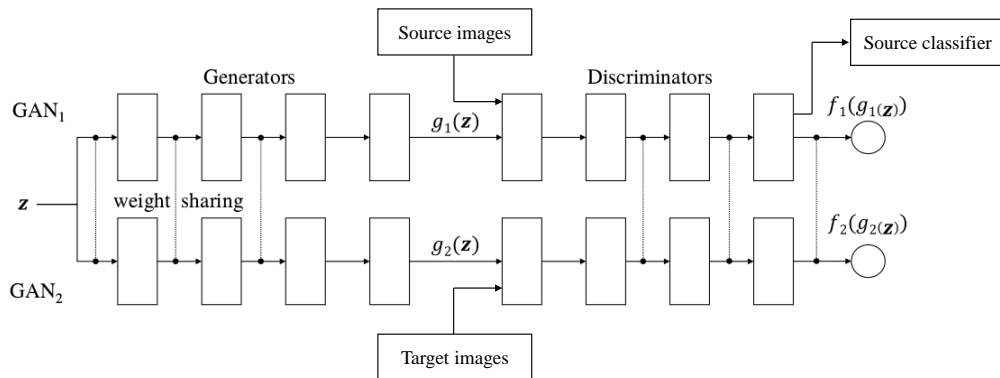


FIGURE 5.6 – Architecture CoGAN pour l'adaptation de domaine non supervisée [12].

Contrairement à la méthode précédente, où le générateur est alimenté par un vecteur aléatoire ( $z$ ) afin de générer des images, Bousmalis et al. [13] ont proposé une méthode (PixelDA) reposant sur les GANs qui utilise un vecteur de bruit en conjonction avec des images du domaine source comme entrée du générateur. L'objectif principal de la méthode est d'entraîner un classifieur à partir des images annotées du domaine source afin qu'il puisse ensuite classer les images non annotées du domaine cible. Pour ce faire, les auteurs ont proposé un modèle composé de trois réseaux (voir figure 5.7) : un générateur ( $G$ ), un discriminateur ( $D$ ) et un classifieur ( $T$ ). Pendant l'entraînement, le générateur génère des images à partir des images sources ( $x^s$ ) altérées avec un vecteur de bruit ( $z$ ). Puis, le discriminateur, alimenté par des images générées et par des images du domaine cible, indique la probabilité qu'une image donnée soit réelle (appartenant au domaine cible). En même temps, le classifieur  $T$  est entraîné de manière supervisée avec les images obtenues par le générateur (images générées). Dû au fait que ces images sont générées pour être similaires à celles du domaine cible, le classifieur  $T$  peut alors être utilisé dans la phase de classification pour classer les images provenant du domaine cible avec succès.

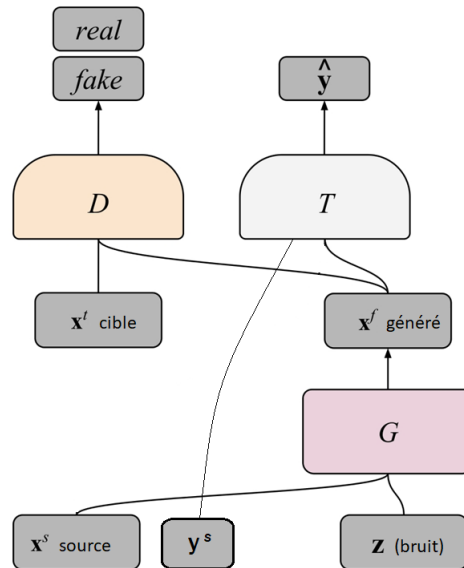


FIGURE 5.7 – Architecture du PixelDA proposée par [13].  $D$ =discriminateur,  $G$ =générateur et  $T$ =classifieur. On cherche à ce que  $x^s$  cible et  $x^f$  généré aient la même distribution.

### Les modèles discriminatifs

Au lieu d'utiliser l'entraînement antagoniste pour produire des images, les modèles discriminatifs utilisent un générateur afin d'obtenir un espace de représentation latent dans lequel les distributions des images sources et cibles sont aussi proches que possible (idéalement identiques). Par conséquent, un classifieur ne pourra pas distinguer l'origine des images.

Les auteurs Ganin et al. [14] ont introduit une nouvelle architecture, nommé DANN

(*Domain-Adversarial Neural Network*). Dans cette architecture, voir figure 5.8, nous pouvons distinguer trois blocs différents : un générateur (ou extracteur) de caractéristiques des images sources et cibles, un discriminateur de domaine et un classifieur. L'idée clé de la méthode est d'extraire des représentations des images sources et cibles qui ne contiennent pas d'informations sur le domaine de provenance. De cette manière, un classifieur entraîné de manière supervisée à partir de ces représentations pourrait être utilisé aussi bien pour les images sources que pour les images cibles.

Pour obtenir non seulement des représentations communes pour les deux domaines (source et cible), mais également assez discriminatives pour accomplir la tâche de classification, le générateur (alimenté par des images sources et cibles) est entraîné conjointement pour berner le discriminateur en extrayant des représentations dépourvues d'informations relatives au domaine et pour minimiser l'erreur de classification des images sources annotées. D'autre part, le discriminateur est entraîné pour bien classer le domaine d'origine des représentations générés par le générateur. Si le générateur est capable de berner le discriminateur, nous pouvons dire que les représentations extraites des images sources et cibles ne contiennent pas d'informations sur le domaine d'origine. De ce fait, le classifieur entraîné de manière supervisée avec les images sources peut être utilisé afin de bien classer les images du domaine cible.

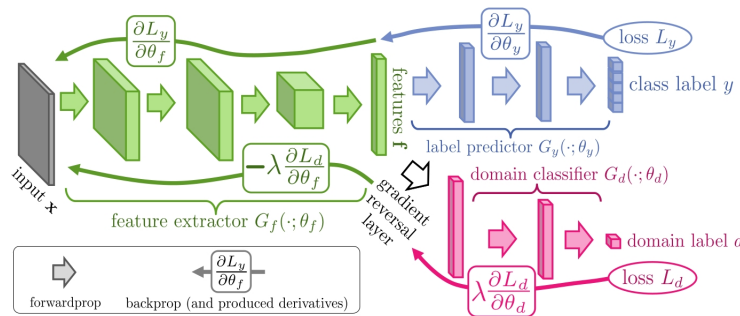


FIGURE 5.8 – Illustration de la méthode DANN proposée par Ganin et al. [14].

Au lieu d'utiliser le même extracteur de caractéristiques pour les images provenant du domaine source et du domaine cible, Tzeng et al. [191] ont proposé une méthode désignée *Adversarial Discriminative Domain Adaptation* (ADDA) qui compte deux extracteurs de caractéristiques différents, un pour chaque domaine. En effet, le modèle proposé est composé d'un extracteur de caractéristiques pour les données source (CNN source), un extracteur de caractéristiques pour les données cible (CNN cible) et un discriminateur. L'entraînement du réseau est réalisé en deux phases séparées. Dans un premier temps, le CNN source, en association avec un classifieur source, est entraîné en utilisant les images sources étiquetées. Dans un second temps, le CNN cible est initialisé avec les paramètres du CNN source pré-entraîné. Tout en conservant les paramètres du CNN source fixes, l'entraînement antagoniste est appliqué : le discriminateur est entraîné pour classer le domaine d'appartenance des caractéristiques extraits de chaque image, tandis que le CNN cible cherche à leurrer le discriminateur. L'utilisation d'extracteurs de caractéristiques adaptés à chaque domaine permet l'obtention de représentations plus appropriées. Néanmoins, les auteurs ont supposé que le classifieur entraîné initialement avec des images



sources pouvait être appliqué directement sur les nouvelles images cibles, ce qui n'est pas toujours le cas [212, 203].

### 5.3.3 Les approches fondées sur la reconstruction

Dans ces méthodes, la reconstruction des images est couramment utilisée comme une tâche auxiliaire afin d'obtenir une représentation partagée entre les deux domaines source et cible. Les auteurs Bousmalis et al. [15] ont proposé une méthode nommée *Domain Separation Networks* (DSN) afin de modéliser conjointement les représentations spécifiques de chaque domaine source et cible, et les représentations partagées par les domaines. Comme nous pouvons l'observer dans la figure 5.9, un encodeur « partagé » est utilisé afin de capturer les représentations partagées par les deux domaines, tandis que deux autres encodeurs « privés » sont utilisés afin d'apprendre individuellement les représentations pertinentes pour les domaines source et cible respectivement. Les représentations partagées et privées de chaque domaine sont ensuite combinées afin d'entraîner deux décodeurs avec poids liés qui ont pour objectif de reconstruire respectivement les images sources et cibles. Un classifieur d'images sources est également entraîné à partir des représentations extraites par l'encodeur partagé. Les auteurs ont ainsi montré qu'en partitionnant l'espace de cette manière (privé et partagé), le classifieur entraîné sur les représentations partagées était en mesure de généraliser aux différents domaines. Bien que les performances obtenues soient satisfaisantes, la méthode proposée est complexe compte tenu du grand nombre de réseaux qui doivent être entraînés (trois encodeurs, un décodeur et un classifieur).

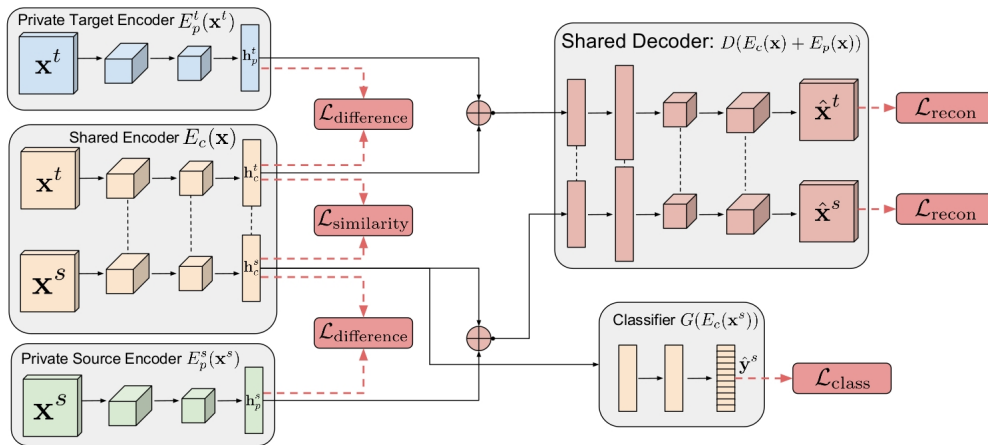


FIGURE 5.9 – Diagramme de la méthode DSN proposée par Bousmalis et al. [15]. Les lignes en tirets impliquent le partage de poids.

Une méthode moins complexe fondée sur la reconstruction d'images, *Deep Reconstruction-Classification Networks* (DRCN), a été introduite par Ghifary et al. [204] (voir la figure 5.10). Dans cette méthode, un auto-encodeur convolutif (CAE) est entraîné conjointement pour classer les images provenant du domaine source et pour reconstruire les images du domaine cible. La partie encodeur du réseau a deux sorties : la première sortie est suivie d'une couche entièrement connectée afin de classer les images annotées du domaine

source et la deuxième sortie est suivie d'un décodeur afin de reconstruire les images du domaine cible. En effet, les caractéristiques à la sortie de l'encodeur doivent être pertinentes et adaptées pour les deux domaines vu qu'elles sont ensuite utilisées à la fois pour classer les images sources et pour reconstruire les images cibles. Une fois l'étape d'apprentissage accomplie, l'encodeur en association avec le classifieur entraîné avec les images annotées du domaine source, est utilisé afin de classer les images du domaine cible.

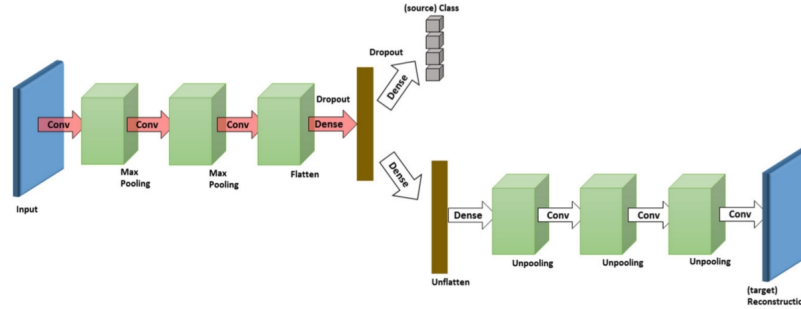


FIGURE 5.10 – Diagramme de la méthode DRCN proposée par Ghifary et al. [15].

Après avoir fait une analyse des méthodes existantes dans la littérature, nous avons pu constater que les méthodes d'adaptation de domaine fondées sur les réseaux antagonistes sont parmi les plus performantes. Nous avons également remarqué que la plupart des méthodes proposées ne prennent pas en considération les possibles décalages dans les distributions jointes de représentations et d'étiquettes ( $P(X^s, Y^s) \neq P(X^c, Y^c)$ ). En effet, une fois que les représentations obtenues des deux domaines sont alignées, le classifieur entraîné à partir des images du domaine source est utilisé sans modification sur les images cibles. À partir de ces constatations, nous avons décidé de proposer une nouvelle méthode d'adaptation de domaine fondée sur les réseaux antagonistes qui prendre en considération les possibles décalages dans les distributions jointes  $P(X^s, Y^s)$  et  $P(X^c, Y^c)$ . En effet, le but de la méthode proposée est d'aligner la distribution du domaine cible avec celle du domaine source et également d'adapter le classifieur source aux nouvelles images cibles.

## 5.4 Système proposé fondé sur l'apprentissage antagoniste

Le schéma global de la méthode proposée pour répondre efficacement au problème d'adaptation de domaine non supervisé (UDA) est décrit par la figure 5.11. Inspirés des travaux de Tzeng et al. [191], nous présentons la méthode qui se divise en trois étapes principales :

1. Tout d'abord, une combinaison d'un réseau de neurones entièrement convolutif ( $FCN_s$ ) et d'un classifieur ( $C_s$ ) est entraînée sur les images sources étiquetées. Le réseau GoogLeNet détaillé dans la section 3.4.1 est utilisé comme architecture  $FCN_s + C_s$  (figure 5.11 A).

2. Une fois le réseau de l'étape précédente entraîné, nous procédons à la deuxième étape dont l'objectif est double : obtenir des représentations des images cibles qui soient impossibles à distinguer de celles des images sources et adapter le classifieur des images sources aux nouvelles images cibles.

Pour accomplir ces objectifs, un deuxième réseau cible  $FCN_c$  initialisé avec les poids du  $FCN_s$  est défini, (figure 5.11 B). Le réseau  $FCN_c$  est entraîné conjointement pour leurrer un discriminateur  $D$  et, en association avec un classifieur cible  $C_c$  initialisé avec les poids du classifieur  $C_s$ , pour minimiser l'erreur de classification des images cibles. Le discriminateur  $D$  est entraîné à dissocier les représentations du domaine source de celles du domaine cible, donc pour l'induire en erreur le  $FCN_c$  doit être en mesure d'extraire des représentations des images cibles dépourvues d'informations sur le domaine.

Pour ce qui est du deuxième objectif qui consiste à minimiser l'erreur de classification des images cibles, compte tenu du fait que les étiquettes ne sont pas disponibles pour le domaine cible, l'association  $FCN_c$  et  $C_c$  est ajusté finement en utilisant une nouvelle fonction de perte, nommée pseudo-étiquette ( $\mathcal{J}_{pl}$ ) qui sera détaillée dans la section 5.4.2.

3. Finalement, dans l'étape de classification, le réseau entièrement convolutif cible ( $FCN_c$ ), ainsi que le classifieur cible ( $C_c$ ), sont utilisés pour les images provenant du domaine cible (figure 5.11 C).

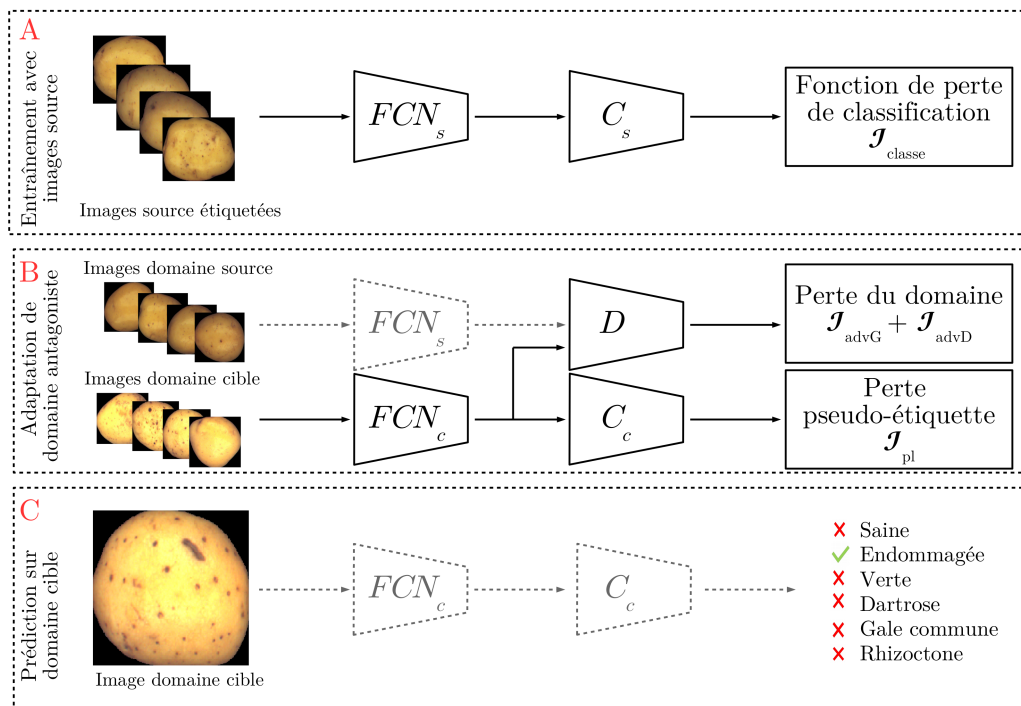


FIGURE 5.11 – Schéma de la méthode proposée pour résoudre le problème d'adaptation de domaine non supervisée. Les lignes en tirets signifient que les paramètres du réseau restent fixes.

Une explication détaillée de chaque étape est donnée dans les sections suivantes.

### 5.4.1 Apprentissage du réseau source

Dans la première étape, des images source étiquetées  $\{(x_i^s, y_i^s)\}_{i=1}^{n_s}$  sont utilisées afin d'entraîner le réseau de neurones entièrement convolutif  $FCN_s$  et le classifieur  $C_s$  d'une manière supervisée. D'un point de vue opérationnel, il est important de conserver l'architecture du modèle qui est déjà en production. L'adaptation du réseaux aux nouvelles images cibles  $\{(x_i^c)\}_{i=1}^{n_c}$  est ainsi simplifiée et sa mise en production accélérée. Selon ce principe, le réseau GoogLeNet, privé de la couche entièrement connectée, est utilisé comme  $FCN_s$ . Cette décision a été prise car c'est l'architecture qui a donnée les meilleurs résultats de classification (se référer aux chapitres 3 et 4). Pour ce qui concerne le  $C_s$ , il consiste en une couche entièrement connectée avec  $M$  neurones, où  $M$  est le nombre de classes de notre base de données, suivie d'une fonction d'activation *softmax*. L'ensemble du réseau ( $FCN_s + C_s$ ) est entraîné de façon à minimiser l'entropie croisée :

$$\mathcal{L}_{class} = -\frac{1}{n_s} \sum_{i=1}^{n_s} \sum_{m=1}^M \mathbb{1}_{(y_i^s=m)} \log(C_s(FCN_s(x_i^s; \theta_{F_s}); \theta_{C_s})) \quad (5.23)$$

où  $\theta_{F_s}$  et  $\theta_{C_s}$  sont les paramètres du  $FCN_s$  et du  $C_s$  respectivement et  $y_i^s \in \{1, 2, \dots, M\}$  est la classe réelle de l'image  $x_i^s$  appartenant à l'ensemble d'apprentissage.

En appliquant ce réseau directement aux images du domaine cible, on peut observer une réduction des performances (résultats détaillés dans les sections 5.5.4 et 5.5.5). Cela s'explique par l'écart existant entre les distributions source et cible. Il est donc nécessaire d'adapter le modèle pour réduire cet écart. En effet, le modèle adapté doit classer correctement les nouvelles images cibles, sans utiliser leurs vraies étiquettes pendant l'apprentissage.

### 5.4.2 Apprentissage antagoniste pour l'adaptation de domaine

L'objectif de cette étape est double : on cherche à aligner la distribution du domaine cible sur celle du domaine source et à adapter le classifieur source sur les nouvelles images cibles. Une fois l'apprentissage du  $FCN_s + C_s$  réalisé, les paramètres appris sont fixés, ce qui nous permet de maintenir intacte la performance du réseau sur les images source. Ensuite, les paramètres du réseau  $FCN_c$  et classifieur  $C_c$  sont initialisées à partir du réseau source pré-entraîné. Comparativement aux GANs introduits dans la section 5.3.2, la sortie du  $FCN_s$  peut être assimilée aux images réelles et le réseau  $FCN_c$  au générateur d'images fausses. Nous effectuons alors l'entraînement du  $FCN_c$  d'une manière antagoniste à l'aide d'un discriminateur de domaine ( $D$ ). En effet, les images provenant de domaine source et cible sont utilisées comme entrée des réseaux  $FCN_s$  et  $FCN_c$  respectivement. Les caractéristiques extraites de ces deux réseaux sont ensuite utilisées comme entrée du discriminateur  $D$ , qui est entraîné afin d'indiquer si les caractéristiques proviennent d'images sources ou cibles. Idéalement, la valeur de sortie du discriminateur doit être égal à 1 pour  $D(FCN_s(x_i^s))$  et égale à 0 pour  $D(FCN_c(x_i^c))$ . En contrepartie, le réseau  $FCN_c$  est entraîné pour leurrer le discriminateur  $D$ . Le problème d'optimisation est donc donné par

l'équation suivante :

$$\begin{aligned} \max_{\theta_{F_c}} \min_{\theta_D} \mathcal{J}_{advD} = & -\frac{1}{n_s} \sum_{i=1}^{n_s} \log(D(FCN_s(x_i^s); \theta_D)) \\ & - \frac{1}{n_c} \sum_{i=1}^{n_c} \log(1 - D(FCN_c(x_i^c; \theta_{F_c}); \theta_D)) \end{aligned} \quad (5.24)$$

où  $\theta_D$  et  $\theta_{F_c}$  sont les paramètres du discriminateur et du  $FCN_c$  respectivement.

Au début de l'apprentissage, le discriminateur peut facilement distinguer les deux domaines puisque le « générateur »  $FCN_c$  n'a pas encore été entraîné correctement. Pour cette raison le terme  $-\log(1 - D(FCN_c(x_i^c; \theta_{F_c}); \theta_D))$  sature (valeur proche de zéro) au début de l'entraînement et les paramètres du générateur ne sont pas mis à jour. En suivant le travail des auteurs Goodfellow et al. [213], nous évitons ce problème de saturation en entraînant  $FCN_c$  pour minimiser  $-\log(D(FCN_c(x_i^c; \theta_{F_c}); \theta_D))$ , au lieu de maximiser  $-\log(1 - D(FCN_c(x_i^c; \theta_{F_c}); \theta_D))$ . Finalement, l'apprentissage antagoniste peut être divisée en deux opérations avec 2 objectifs distincts. Le discriminateur  $D$  est entraîné pour minimiser  $\mathcal{J}_{advD}$  (équation 5.24), et le  $FCN_c$  pour minimiser :

$$\mathcal{J}_{advG} = -\frac{1}{n_c} \sum_{i=1}^{n_c} \log(D(FCN_c(x_i^c; \theta_{F_c}))) \quad (5.25)$$

En plus d'entraîner le  $FCN_c$  pour berner le discriminateur, il est également optimisé, en association avec le  $C_c$ , pour la classification des images du nouveau domaine cible. Ben-David et al. [214] ont montré que même si les deux domaines source et cible sont alignés, il est possible que le classifieur source ( $C_s$ ) ne soit pas tout à fait adapté pour bien classer les échantillons cibles à cause d'un décalage résiduel des distributions jointes ( $P(X^s, Y^s) \neq P(X^c, Y^c)$ ). Il est également possible que la distribution jointe  $P(X^c, Y^c)$  contienne un part d'information spécifique qui contribue à la discrimination et qu'un pur alignement perdrait. Face à l'absence d'étiquettes des images cibles, il n'est pas aisé d'entraîner l'association  $FCN_c + C_c$  pour la tâche de classification. Cependant, étant donné que le  $FCN_c$  et le  $C_c$  ont été initialisés avec les paramètres du  $FCN_s$  et du  $C_s$  respectivement, il est possible d'obtenir des étiquettes pour les images cibles. La classification obtenue avec l'association  $FCN_c + C_c$  est donc désignée par pseudo-étiquette et utilisée pour l'entraînement si la plus grande valeur de sortie dépasse un seuil de confiance (fixé à 0,9 dans les différentes expérimentations). De cette façon, nous n'attribuons un label que si le réseau donne un résultat clairement favorable à une classe. Il faut noter que cette approche de pseudo-étiquette a été employé avec succès dans certaines méthodes d'apprentissage semi-supervisé [215, 216]. Au cours de l'apprentissage, le nombre d'images cibles auxquelles un label est attribue tend à augmenter. Cette augmentation est due au fait que le  $FCN_c$  apprend une représentation plus appropriée et plus discriminative des données cibles.

L'entraînement du  $FCN_c$  et du  $C_c$  se fait donc en minimisant :

$$\mathcal{J}_{pl} = -\frac{1}{n_c} \sum_{i=1}^{n_c} \sum_{m=1}^M \mathbb{1}_{(\hat{y}_i^c = m)} \log(C_c(FCN_c(x_i^c; \theta_{F_c}); \theta_{C_c})) \quad (5.26)$$

où  $\theta_{F_c}$  et  $\theta_{C_c}$  sont respectivement les paramètres du  $FCN_c$  et du classifieur cible  $C_c$ ,  $M$  est le nombre de classes et  $\widehat{y}_i^c \in \{1, 2, \dots, M\}$  est la pseudo-étiquette, correspondant à l'image cible  $x_i^c$ , obtenue à partir du réseau cible ( $FCN_c + C_c$ ).

En résumé, cette étape d'adaptation de domaine non supervisée est basée sur la minimisation itérative de la fonction de perte suivante :

$$\mathcal{J}_{total} = \mathcal{J}_{adv_D} + \mathcal{J}_{adv_G} + \lambda_{pl}\mathcal{J}_{pl} \quad (5.27)$$

où  $\lambda_{pl}$  est l'hyper-paramètre qui permet d'ajuster l'importance de la fonction de perte de pseudo-étiquette.

Une fois l'entraînement terminé, le  $FCN_c$  et le  $C_c$  sont utilisés conjointement pour classer les images provenant du domaine cible.

## 5.5 Résultats expérimentaux

Dans cette section, nous présentons les résultats obtenus par la méthode proposée selon deux scénarios différents. Dans le premier cas, un changement de luminosité sur les images cibles est considéré. Dans le second cas, une nouvelle variété de pommes de terre est analysée. Nous allons également comparer les résultats de notre méthode avec d'autres approches de l'état de l'art. Nous allons montrer qu'en appliquant cette méthode nous sommes en mesure de classer avec succès les défauts des pommes de terre provenant d'un domaine cible en utilisant uniquement les étiquettes des images d'un domaine source différent. Les implémentations des méthodes ont été réalisées en python. La bibliothèque pytorch [217] a été utilisée pour la partie relative à l'apprentissage profond. Toutes les expériences ont été réalisées avec un GPU NVIDIA GeForce GTX 1070 Ti (8 Go de mémoire).

### 5.5.1 Bases de données utilisées

Afin d'entraîner et valider la méthode proposée, deux bases de données ont été créées et utilisées. La première base a pour objectif de tester la pertinence de l'approche d'adaptation de domaine non supervisée dans le contexte d'un changement de luminosité des images. La deuxième base sert à montrer la pertinence de la méthode proposée quand une nouvelle variété de pomme de terre doit être analysée.

#### Images avec un changement de luminosité

La base de données présentée au chapitre 3 a été utilisée pour ces expériences. L'ensemble de 9688 images incluant 6 classes a été divisé en deux : 70% des images ont été utilisées pour former l'ensemble d'apprentissage, et le reste a été utilisé pour tester les modèles. L'ensemble d'apprentissage a été ensuite divisé en images sources et images cibles. Pour créer ces dernières, la luminosité des images a été augmentée pour simuler un changement des conditions d'éclairage lors de l'acquisition. En ce qui concerne l'ensemble de test, toutes les images ont été modifiées pour appartenir au domaine cible. Le tableau 5.1 montre le nombre d'images par classe et par domaine. Un exemple d'images de l'ensemble

de données est présenté dans la figure 5.12, où la même image est montrée avant et après la transformation afin d'observer la différence entre les deux domaines. Lors de la génération de la base de données d'entraînement, la même image ne pouvait appartenir qu'à un seul domaine.

TABLEAU 5.1 – Nombre d'images dans la base de données de pommes de terre jaunes.

Classe	Nb d'images	Nb d'images sources d'entraînement	Nb d'images cibles d'entraînement	Nb d'images cibles de test
Saine	5325	2989	747	1586
Endommagée	984	535	134	315
Verte	1263	700	175	391
Dartrose	597	344	86	167
Gale Commune	1276	733	183	360
Rhizoctone	243	139	35	69
Total	9688	5440	1360	2888

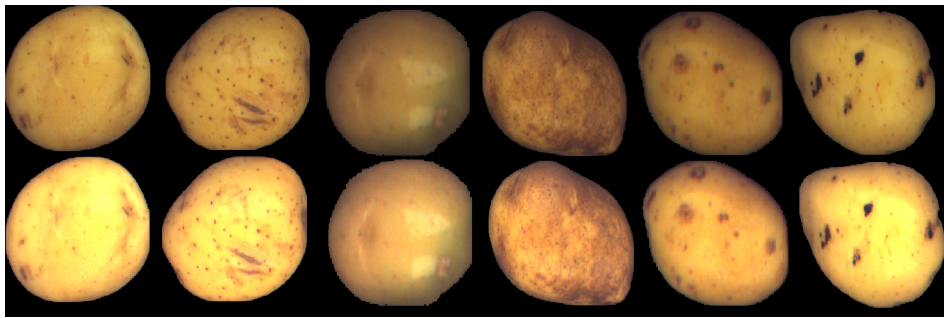


FIGURE 5.12 – Images des différentes classes appartenant au domaine source en haut et cible en bas. Par colonne les classes : saine, endommagée, verte, dartrose, gale commune et rhizoctone.

### Images d'une nouvelle variété de pommes de terre

Dans le contexte de l'adaptation de domaine pour une nouvelle variété de pommes de terre, les images de la base de données initiale (tubercules jaunes) ont été utilisées pour former l'ensemble source. Le deuxième ensemble contenant des images du domaine cible, a été constitué de pommes de terre rouges étiquetées. Nous montrons un exemple des images appartenant à chaque domaine dans la figure 5.13. Comme nous pouvons l'observer, les deux variétés sont très différentes. Le nombre d'images appartenant à chaque domaine est indiqué dans le tableau 5.2.

### 5.5.2 Méthodes comparatives

Comme nous avons vu précédemment dans la section 5.3, plusieurs approches d'adaptation de domaine ont été proposées dans la littérature. Afin de réaliser une étude comparative,



FIGURE 5.13 – Images des différentes classes appartenant au domaine source en haut (variété jaune) et cible en bas (variété rouge). Par colonne les classes : saine, endommagée, verte, dartrose, gale commune et rhizoctone.

TABLEAU 5.2 – Nombre d’images dans la base de données de pommes de terre jaunes (source) et rouges (cible).

Classe	Nb d’images	Nb d’images sources d’entraînement	Nb d’images cibles d’entraînement	Nb d’images cibles de test
Saine	7850	3736	3274	840
Endommagée	1420	669	612	139
Verte	2030	875	931	224
Dartrose	958	430	427	101
Gale Commune	2182	918	1024	240
Rhizoctone	376	172	156	48
Total	14816	6800	6424	1592

nous avons comparé notre méthode avec deux approches fondées sur la divergence et avec une autre méthode fondée sur l’apprentissage antagoniste. Pour ce qui est des approches fondées sur la divergence, Deep Correlation Alignment (Deep CORAL) [11] et Joint Adaptation Networks (JAN) [10] ont été évaluées (les détails de chacune de ces méthodes se trouvent dans la section 5.3.1). La méthode Adversarial Discriminative Domain Adaptation (ADDA) [191], a été choisie comme méthode comparative fondée sur l’apprentissage antagoniste. Cette méthode est très similaire à notre méthode mis à part le fait qu’elle utilise le même classifieur entraîné avec les images sources pour classer ensuite les images cibles, voir section 5.3.2. Dans notre méthode, le classifieur initialement entraîné avec les images sources est adapté pour le nouveau domaine cible, il est donc intéressant de pouvoir évaluer la pertinence de cette adaptation complémentaire.

Il est important de noter que toutes les méthodes comparatives respectent l’exigence selon laquelle l’architecture du réseau pré-entraîné avec les images sources étiquetées est conservée lors de la phase d’adaptation. L’architecture du réseau cible est identique à celle du réseau initial. Cette exigence nous permet de mettre en œuvre rapidement le nouveau modèle, ce qui est essentiel dans notre contexte industrielle.



### 5.5.3 Détails des implémentations

Pour toutes les méthodes, le réseau GoogLeNet sans la dernière couche entièrement connectée a été utilisé comme réseau entièrement convolutif ( $FCN_s$  et  $FCN_c$ ). Les classifieurs source et cible ( $C_s$  et  $C_c$ ) étaient quant à eux constitués d'une couche entièrement connectée de 6 neurones, où 6 est le nombre de classes de notre ensemble de données (saine, endommagée, verte, dartrose, gale commune et rhizoctone), suivi d'une fonction *softmax*. Les images d'entrée ont été redimensionnées à  $224 \times 224$  pixels afin de respecter l'architecture du réseau GoogLeNet. Le discriminateur ( $D$ ) est composé de 3 couches entièrement connectées : la première couche ayant 1024 unités, la deuxième 500 unités et la troisième une unité pour classer le domaine (source ou cible). Les deux premières couches entièrement connectées utilisent une fonction d'activation ReLU afin d'introduire la non-linéarité dans le réseau. La dernière couche, emploie une fonction sigmoïde. Les techniques d'augmentation des données, telles que la rotation et le retournement vertical et horizontal, ont été appliquées de manière aléatoire lors de l'apprentissage du réseau.

Dans la première étape de la méthode proposée (section 5.4.1), les réseaux  $FCN_s$  et  $C_s$  ont été entraînés en utilisant la descente de gradient stochastique (SGD) avec une taille de mini-lot de 32, un momentum de 0,9 et un *weight decay* de  $5 \times 10^{-4}$ . Dans la deuxième étape (section 5.4.2), le réseau entièrement convolutif  $FCN_c$  et le classifieur  $C_c$  ont été entraînés en utilisant également l'algorithme de SGD avec une taille de mini-lot de 128 et un taux d'apprentissage de  $1 \times 10^{-5}$ . Quant au discriminateur  $D$ , il a été entraîné en utilisant l'optimiseur Adam avec  $\beta_1 = 0.5$ ,  $\beta_2 = 0.999$  et un taux d'apprentissage de  $1 \times 10^{-4}$ . Le nombre d'*epochs* a été fixé à 200. L'hyper-paramètre  $\lambda_{pl}$  qui pondère la fonction de perte de pseudo-étiquette a été progressivement modifié de 0 à 1 par  $\lambda_{pl} = \frac{n_e - 1}{t_e - 1}$ , où  $n_e$  est le nombre d'*epochs* courant et  $t_e$  le nombre maximal d'*epochs*.

Pour ce qui est de la méthode Deep CORAL (introduite dans la section 5.3.1), les caractéristiques obtenues par la dernière couche de  $FCN_s$  et  $FCN_c$  ont été utilisées pour calculer la fonction de perte CORAL ( $\mathcal{J}_{CORAL}$ , équation 5.20). La descente de gradient avec une taille de mini-lot de 32, un momentum de 0,9 et un *weight decay* de  $5 \times 10^{-4}$  a été utilisée pour entraîner le réseau. Le taux d'apprentissage a été fixé à  $1 \times 10^{-4}$ , sauf pour la couche entièrement connectée de  $C_s$ , qui a été fixée à 10 fois le taux d'apprentissage initial. L'hyper-paramètre  $\lambda_{CORAL}$  a été fixé à 5 pour compenser la perte de classification des images source et l'adaptation de domaine.

En ce qui concerne la méthode JAN (introduite dans la section 5.3.1), nous avons utilisé un noyau gaussien avec un paramètre de lissage défini par la distance médiane au carré entre les observations d'entraînement [207]. Deux couches ont été utilisées pour calculer la pénalité JMMD (équation 5.18) : la sortie du réseau  $FCN_s$  et la sortie du classifieur  $C_s$ . Similaire à la méthode de Deep CORAL, les réseaux source et cible utilisaient des poids liés, donc  $FCN_s = FCN_c$  et  $C_s = C_c$ . L'entraînement a été effectué grâce à l'algorithme de SGD avec les mêmes hyper-paramètres que ceux utilisés pour l'entraînement de Deep CORAL. Nous avons également modifié de manière graduelle l'hyper-paramètre  $\lambda_{JAN}$  de

0 à 1 avec  $\lambda_{JAN} = \frac{n_e - 1}{t_e - 1}$ .

Quant à la dernière méthode de comparaison ADDA (introduite dans la section 5.3.2), nous avons utilisé l’optimiseur et les hyper-paramètres détaillées précédemment pour notre méthode. Cependant, à la différence de notre méthode, ADDA n’utilise pas un classifieur cible car le classifieur entraîné sur les images du domaine source est directement appliqué sur les nouvelles images cibles dans l’étape de classification.

Finalement, nous avons suivi les travaux de French et al. [218] afin de définir l’*epoch* auquel les performances sont calculées. En effet, dans le contexte de l’adaptation de domaine non supervisée, un ensemble de validation étiqueté d’images cibles n’est pas disponible. De ce fait, nous avons calculé la performance de chaque méthode sur l’ensemble de test à l’*epoch* où le plus grand nombre d’images d’entraînement du domaine cible a obtenu un taux de classification maximal avec une valeur de sortie du réseau supérieure au seuil de confiance, fixé à 0,9.

#### 5.5.4 Résultats sur la base de données avec un changement de luminosité

La moyenne et l’écart-type de la précision, le rappel et la F-mesure obtenus à partir de cinq expériences réalisées en définissant aléatoirement les différents groupes de données sont indiqués dans le tableau 5.3. Une limite inférieure et une limite supérieure de performance ont été calculées pour comparer les méthodes. « Seulement Source » correspond à la limite inférieure de la performance, où aucune adaptation n’est effectuée. « Seulement Cible », quant à lui, représente les résultats de la limite supérieure, où le réseau est entraîné sur l’ensemble de données cible avec étiquettes. Idéalement, cette limite supérieure est ce qu’on peut espérer d’atteindre.

Nous pouvons observer à partir des résultats obtenus que l’utilisation d’une méthode d’adaptation de domaine est indispensable pour éviter une diminution significative des performances dans le domaine cible. La classe avec la performance la plus faible est la dartoise, qui est la plus difficile à détecter même dans le cas où nous réalisons l’entraînement du réseau avec les étiquettes des images cible (« Seulement Cible »). Nous pouvons également observer que les méthodes reposant sur l’apprentissage antagoniste (ADDA et la nôtre) donnent de meilleurs résultats que les méthodes fondées sur la divergence (Deep CORAL et JAN). En comparant ADDA avec notre méthode, nous pouvons remarquer que l’ajout de la fonction de perte de pseudo-étiquette a aidé le classifieur à augmenter légèrement la performance sur l’ensemble de données cibles. Enfin, notre méthode est comparable à la limite supérieure « Seulement Cible », qui est entraînée d’une manière entièrement supervisée (avec les étiquetées des images cibles).

Dans l’objectif d’analyser les confusions entre les différentes classes nous avons également calculé les matrices de confusion pour chaque méthode. Ces matrices, représentées à la figure 5.14, nous permettent de conclure que :

TABLEAU 5.3 – Moyenne et écart-type de la précision, le rappel et la F-mesure sur l’ensemble de test des images cibles. Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

Critère	Classes	Seulement Source	Deep CORAL	JAN	ADDA	Notre méthode	Seulement Cible
Précision	S	0,77 ± 0,06	0,85 ± 0,01	0,89 ± 0,00	0,89 ± 0,00	<b>0,90 ± 0,01</b>	0,89 ± 0,00
	E	0,50 ± 0,14	0,84 ± 0,02	0,83 ± 0,03	0,84 ± 0,03	0,85 ± 0,01	<b>0,87 ± 0,01</b>
	V	<b>1,00 ± 0,00</b>	0,91 ± 0,03	0,92 ± 0,02	0,99 ± 0,01	0,98 ± 0,01	0,95 ± 0,01
	D	0,00 ± 0,00	0,56 ± 0,03	0,71 ± 0,01	0,74 ± 0,05	0,77 ± 0,02	<b>0,78 ± 0,02</b>
	GC	0,54 ± 0,09	0,77 ± 0,02	0,83 ± 0,03	0,85 ± 0,03	0,83 ± 0,02	<b>0,86 ± 0,02</b>
	R	0,66 ± 0,18	<b>0,91 ± 0,01</b>	0,84 ± 0,04	0,85 ± 0,05	0,88 ± 0,03	0,87 ± 0,01
	Moy.	0,58 ± 0,08	0,81 ± 0,02	0,84 ± 0,02	0,86 ± 0,03	<b>0,87 ± 0,02</b>	<b>0,87 ± 0,01</b>
Rappel	S	0,82 ± 0,08	0,92 ± 0,01	0,93 ± 0,01	<b>0,95 ± 0,01</b>	<b>0,95 ± 0,01</b>	<b>0,95 ± 0,00</b>
	E	0,66 ± 0,09	0,75 ± 0,01	0,83 ± 0,02	0,85 ± 0,01	<b>0,86 ± 0,01</b>	0,83 ± 0,01
	V	0,20 ± 0,08	0,71 ± 0,02	<b>0,82 ± 0,03</b>	0,78 ± 0,01	0,79 ± 0,01	0,81 ± 0,02
	D	0,00 ± 0,00	0,57 ± 0,05	0,70 ± 0,02	<b>0,73 ± 0,03</b>	<b>0,73 ± 0,04</b>	0,72 ± 0,02
	GC	0,74 ± 0,14	0,78 ± 0,02	0,79 ± 0,05	0,78 ± 0,03	<b>0,82 ± 0,03</b>	0,79 ± 0,02
	R	0,50 ± 0,17	0,55 ± 0,05	0,82 ± 0,04	0,79 ± 0,03	0,79 ± 0,03	<b>0,85 ± 0,02</b>
	Moy.	0,49 ± 0,09	0,72 ± 0,02	0,81 ± 0,03	0,81 ± 0,02	<b>0,82 ± 0,02</b>	<b>0,82 ± 0,01</b>
F-mesure	S	0,79 ± 0,02	0,88 ± 0,00	0,91 ± 0,00	<b>0,92 ± 0,00</b>	<b>0,92 ± 0,00</b>	<b>0,92 ± 0,00</b>
	E	0,54 ± 0,07	0,79 ± 0,01	0,83 ± 0,01	<b>0,85 ± 0,01</b>	<b>0,85 ± 0,00</b>	<b>0,85 ± 0,00</b>
	V	0,32 ± 0,11	0,80 ± 0,01	<b>0,87 ± 0,01</b>	<b>0,87 ± 0,01</b>	<b>0,87 ± 0,01</b>	<b>0,87 ± 0,01</b>
	D	0,00 ± 0,00	0,57 ± 0,02	0,70 ± 0,01	0,73 ± 0,02	<b>0,75 ± 0,02</b>	0,74 ± 0,01
	GC	0,59 ± 0,06	0,77 ± 0,01	0,81 ± 0,02	0,81 ± 0,01	<b>0,83 ± 0,01</b>	0,82 ± 0,00
	R	0,51 ± 0,09	0,69 ± 0,03	0,83 ± 0,02	0,81 ± 0,03	0,83 ± 0,03	<b>0,86 ± 0,01</b>
	Moy.	0,46 ± 0,06	0,75 ± 0,01	0,82 ± 0,01	0,83 ± 0,01	<b>0,84 ± 0,01</b>	<b>0,84 ± 0,01</b>

1. La classe la plus affectée par le changement de luminosité des images a été la dartrose (D). Sans méthode d’adaptation de domaine, cette classe n’est plus détectée (voir figure 5.14a), et elle est dans la plupart des cas confondue avec la classe saine.
2. La classe verte (V) a été également affectée par le changement de domaine, passant de 78,6% de détection avec notre méthode à 19,6% sans l’adaptation (« Seulement source »). Ceci peut s’expliquer par l’importance des couleurs de l’image pour détecter correctement cette classe.
3. La confusion entre les classes similaires, par exemple la gale commune (GC) et le rhizoctone (R), a augmenté significativement en l’absence d’adaptation du domaine.
4. La plus grande différence entre la méthode ADDA et notre méthode se situe dans la détection de la gale commune, passant de 77,6% à 82,4%, respectivement.

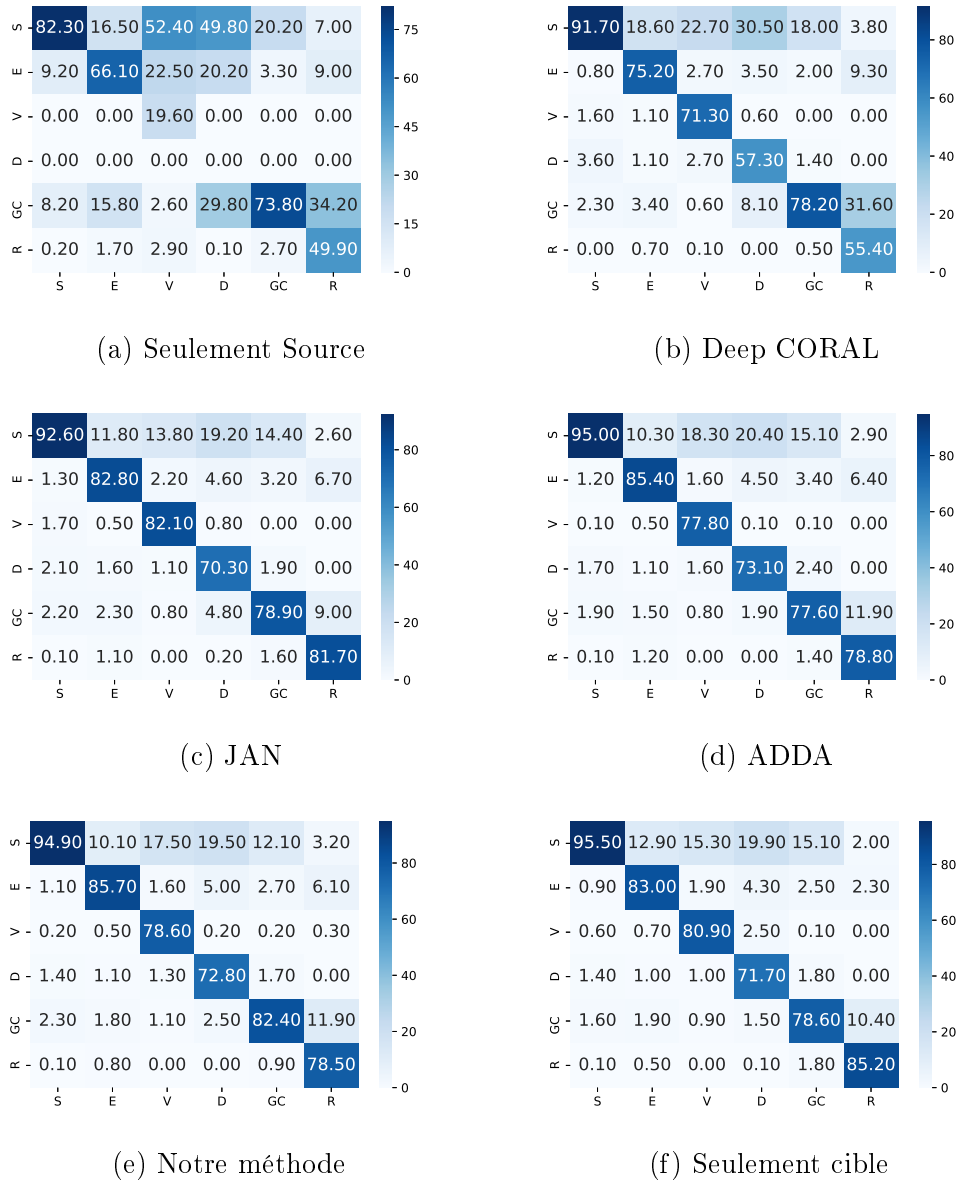


FIGURE 5.14 – Matrices de confusion obtenus à partir des méthodes (a) seulement source, (b) deep CORAL, (c) JAN, (d) ADDA, (e) notre méthode, (f) seulement cible. Par colonne : classe réelle. Par ligne : classe prédite. Classes : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone. Valeurs en %.

### Visualisation des représentations

Nous utilisons la technique t-SNE [219] afin de visualiser les représentations apprises par le réseau entièrement convolutionnel  $FCN_c$  entraîné selon les méthodes : « Seulement Source », Deep CORAL, JAN, ADDA et notre méthode. La figure 5.15 montre les représentations obtenues. Nous pouvons observer que si nous utilisons uniquement des images du domaine source pour l'entraînement du réseau (figure 5.15a), les représenta-

tions extraites des images cibles de différentes classes sont très mélangées. D'autre part, les caractéristiques sont plus discriminantes lorsque des méthodes d'adaptation de domaine sont appliquées (figures 5.15b-5.15e). Enfin, nous pouvons observer que même si les visualisations des représentations obtenues à partir des différentes méthodes d'adaptation de domaine sont similaires, certaines classes sont plus compactes avec notre méthode.

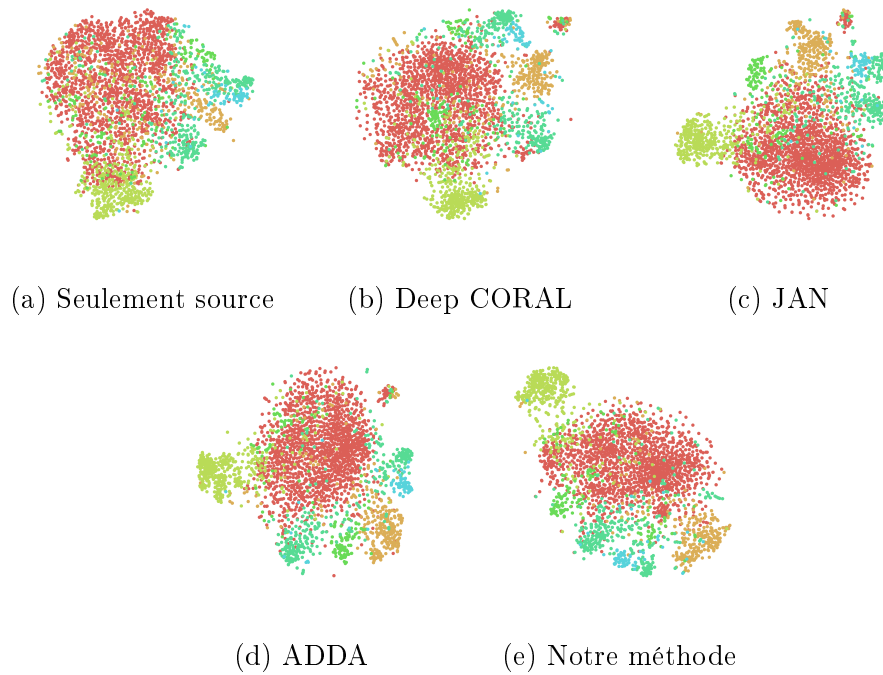


FIGURE 5.15 – Visualisation des caractéristiques extraites des images provenant de domaine cible (variété jaune avec un changement de luminosité) en utilisant la méthode t-SNE à partir des différents méthodes (a) Seulement source, (b) Deep CORAL, (c) JAN, (d) ADDA and (e) Notre méthode. Chaque couleur représente une classe différente : saine, endommagée, verte, dartrose, gale commune et rhizoctone.

### 5.5.5 Résultats sur différentes variétés de couleur de pommes de terre

Pour étudier un cas de grande variation entre les domaines source et cible, nous présentons les résultats obtenus en utilisant des images de pommes de terre jaunes comme base de données du domaine source et des pommes de terre avec une peau de couleur rouge comme base de données du domaine cible. La moyenne et l'écart-type de la précision, le rappel et la F-mesure obtenus en effectuant cinq expériences aléatoires sont présentés dans le tableau 5.4. À partir de ces résultats nous pouvons constater que le réseau entraîné avec les images sources ne peut pas être directement utilisé pour classer les nouvelles images cibles (F-mesure moyenne : 0,12). De ce fait, l'utilisation d'une méthode d'adaptation du domaine est indispensable afin d'améliorer les performances du réseau avec les images

cibles. Comme précédemment (voir section 5.5.4), on peut observer que les méthodes basées sur l'apprentissage antagoniste (ADDA et notre méthode) sont plus performantes que les méthodes basées sur la divergence (Deep CORAL et JAN).

Nous pouvons également constater que dans le cas où la différence entre les domaines est très importante, nous ne pouvons pas garantir l'adaptation de certains défauts ou maladies. Cela se produit notamment pour les classes dartrose et gale commune, dont les symptômes visuels sont très évolutifs en fonction de la variété de la pomme de terre affectée (voir figure 5.13). À titre d'exemple, la  $F_1$ -mesure moyenne obtenue à l'aide de notre méthode pour la dartrose et la gale commune est de 0,22 et 0,50 respectivement.

TABLEAU 5.4 – Moyenne et écart-type de la précision, le rappel et la F-mesure sur l'ensemble de test des images cibles (variété rouge). Les classes sont : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone.

Critère	Classes	Seulement Source	Deep CORAL	JAN	ADDA	Notre méthode	Seulement Cible
Précision	S	$0,59 \pm 0,23$	$0,68 \pm 0,01$	$0,76 \pm 0,01$	$0,76 \pm 0,00$	$0,75 \pm 0,00$	<b><math>0,89 \pm 0,01</math></b>
	E	$0,21 \pm 0,03$	$0,54 \pm 0,01$	$0,66 \pm 0,04$	$0,71 \pm 0,02$	$0,72 \pm 0,01$	<b><math>0,88 \pm 0,01</math></b>
	V	$0,57 \pm 0,22$	$0,81 \pm 0,02$	$0,85 \pm 0,06$	$0,90 \pm 0,01$	$0,90 \pm 0,01$	<b><math>0,93 \pm 0,01</math></b>
	D	$0,07 \pm 0,00$	$0,18 \pm 0,03$	$0,16 \pm 0,07$	$0,36 \pm 0,07$	$0,39 \pm 0,10$	<b><math>0,81 \pm 0,05</math></b>
	GC	$0,16 \pm 0,09$	$0,5 \pm 0,01$	$0,48 \pm 0,05$	$0,66 \pm 0,02$	$0,68 \pm 0,04$	<b><math>0,81 \pm 0,01</math></b>
	R	$0,00 \pm 0,00$	$0,86 \pm 0,06$	$0,88 \pm 0,03$	$0,82 \pm 0,04$	$0,85 \pm 0,05$	<b><math>0,93 \pm 0,02</math></b>
	Moy.	$0,27 \pm 0,08$	$0,59 \pm 0,02$	$0,63 \pm 0,04$	$0,70 \pm 0,03$	$0,71 \pm 0,04$	<b><math>0,87 \pm 0,02</math></b>
Rappel	S	$0,03 \pm 0,03$	$0,90 \pm 0,01$	$0,85 \pm 0,02$	$0,92 \pm 0,01$	<b><math>0,93 \pm 0,00</math></b>	$0,92 \pm 0,01$
	E	$0,38 \pm 0,13$	$0,48 \pm 0,03$	$0,63 \pm 0,06$	$0,63 \pm 0,04$	$0,61 \pm 0,00$	<b><math>0,85 \pm 0,03</math></b>
	V	$0,05 \pm 0,03$	$0,43 \pm 0,07$	$0,80 \pm 0,03$	$0,90 \pm 0,02$	$0,90 \pm 0,01$	<b><math>0,92 \pm 0,00</math></b>
	D	$0,09 \pm 0,05$	$0,17 \pm 0,02$	$0,12 \pm 0,05$	$0,17 \pm 0,06$	$0,15 \pm 0,08$	<b><math>0,74 \pm 0,02</math></b>
	GC	$0,75 \pm 0,12$	$0,23 \pm 0,01$	$0,40 \pm 0,05$	$0,43 \pm 0,02$	$0,40 \pm 0,01$	<b><math>0,76 \pm 0,02</math></b>
	R	$0,00 \pm 0,00$	$0,45 \pm 0,04$	$0,52 \pm 0,05$	$0,59 \pm 0,08$	$0,66 \pm 0,06$	<b><math>0,87 \pm 0,02</math></b>
	Moy.	$0,22 \pm 0,06$	$0,44 \pm 0,03$	$0,55 \pm 0,04$	$0,61 \pm 0,04$	$0,61 \pm 0,03$	<b><math>0,84 \pm 0,02</math></b>
F-mesure	S	$0,05 \pm 0,06$	$0,78 \pm 0,00$	$0,80 \pm 0,00$	$0,83 \pm 0,00$	$0,83 \pm 0,00$	<b><math>0,90 \pm 0,00</math></b>
	E	$0,25 \pm 0,07$	$0,51 \pm 0,01$	$0,64 \pm 0,02$	$0,66 \pm 0,02$	$0,66 \pm 0,01$	<b><math>0,87 \pm 0,01</math></b>
	V	$0,09 \pm 0,06$	$0,56 \pm 0,06$	$0,82 \pm 0,03$	$0,90 \pm 0,01$	$0,90 \pm 0,00$	<b><math>0,93 \pm 0,00</math></b>
	D	$0,07 \pm 0,02$	$0,17 \pm 0,03$	$0,14 \pm 0,05$	$0,23 \pm 0,06$	$0,22 \pm 0,10$	<b><math>0,77 \pm 0,01</math></b>
	GC	$0,26 \pm 0,01$	$0,32 \pm 0,01$	$0,43 \pm 0,04$	$0,52 \pm 0,02$	$0,50 \pm 0,01$	<b><math>0,78 \pm 0,01</math></b>
	R	$0,00 \pm 0,00$	$0,59 \pm 0,04$	$0,65 \pm 0,04$	$0,69 \pm 0,05$	$0,74 \pm 0,04$	<b><math>0,90 \pm 0,01</math></b>
	Moy.	$0,12 \pm 0,04$	$0,49 \pm 0,02$	$0,58 \pm 0,03$	$0,64 \pm 0,03$	$0,64 \pm 0,02$	<b><math>0,86 \pm 0,01</math></b>

Dans le but d'analyser les confusions entre les différentes classes, nous présentons les matrices de confusion obtenus pour chaque méthode à la figure 5.16. Comme nous pouvons l'observer, les classes dartrose (D) et gale commune (GC) sont couramment confondues avec la classe saine (S). Cette confusion est fortement réduite dans le cas où les étiquettes des images cibles sont utilisées dans l'entraînement du réseau (« Seulement Cible »). En comparant les méthodes basées sur l'apprentissage antagoniste (ADDA et la nôtre), on

constate que l'adaptation du classifieur améliore légèrement les résultats. La plus grande différence réside dans la bonne détection de la classe rhizoctone (R). En effet, la bonne détection la plus élevée de la classe rhizoctone (R) a été obtenue avec notre méthode (66.25%). Dans le cas de la méthode ADDA, cette classe a obtenu une détection plus faible (59.17%) en raison d'une plus grande confusion avec la gale commune (GC).

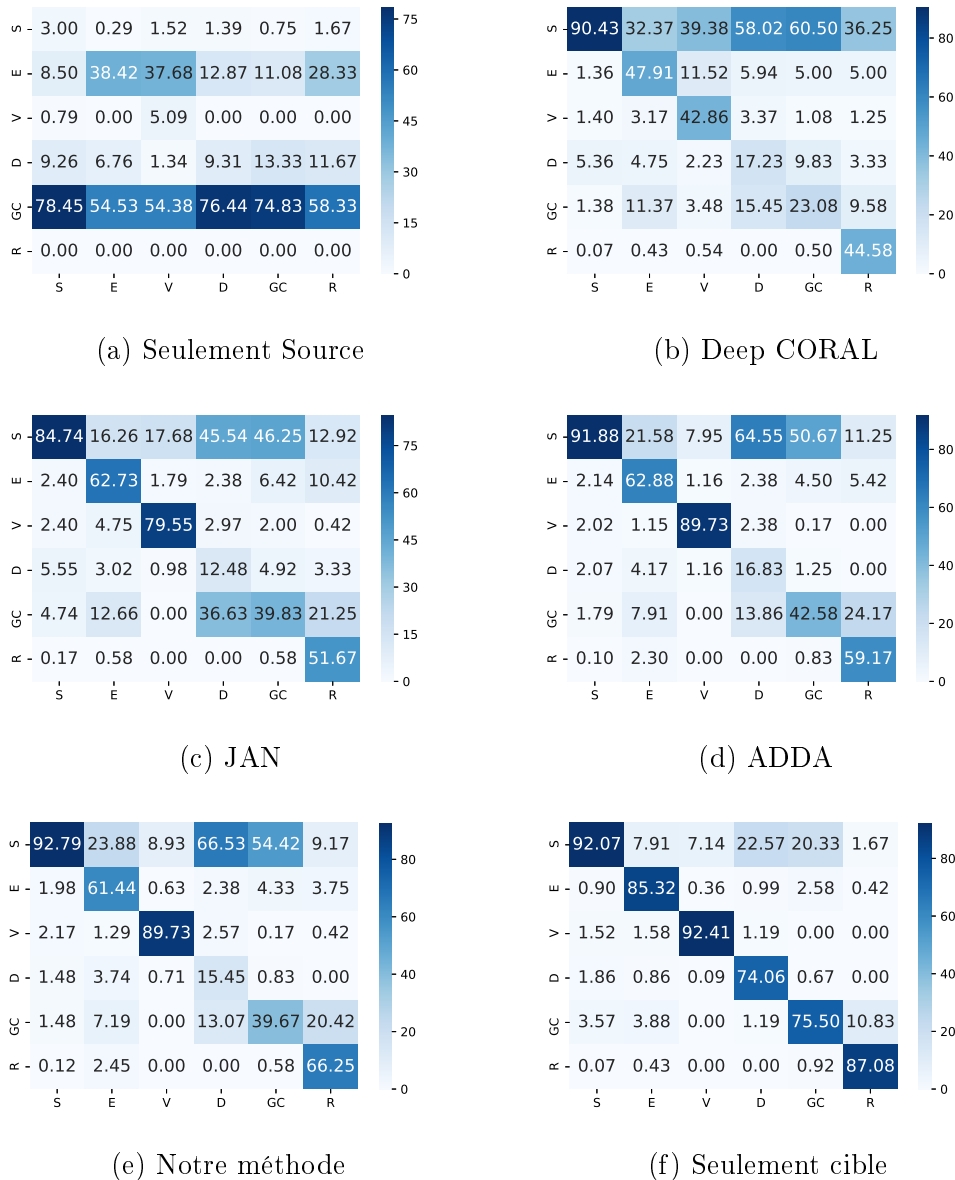


FIGURE 5.16 – Matrices de confusion obtenus à partir des méthodes (a) seulement source, (b) deep CORAL, (c) JAN, (d) JAN, (e) notre méthode, (f) seulement cible. Par colonne : classe réelle. Par ligne : classe prédite. Classes : S=Saine, E=Endommagée, V=Verte, D=Dartrose, GC=Gale Commune et R=Rhizoctone. Valeurs en %.

## Visualisation des représentations

Afin de visualiser les représentations apprises par le réseau de chaque méthode (« Seule-ment Source », Deep CORAL, JAN, ADDA et notre méthode) nous avons utilisé la technique t-SNE. La figure 5.17 illustre les résultats obtenus. Comme nous pouvons le noter, sans l’adaptation du domaine les représentations de chaque classe sont, dans la plupart des cas, indissociables (figure 5.17a). L’obtention de ces représentations nous aide à comprendre la raison pour laquelle l’utilisation d’une méthode d’adaptation du domaine est indispensable. Les représentations obtenues pour chaque classe sont plus faciles à distinguer et à séparer en appliquant notre méthode (figure 5.17e). Cette séparation est très importante pour faciliter la tâche de classification.

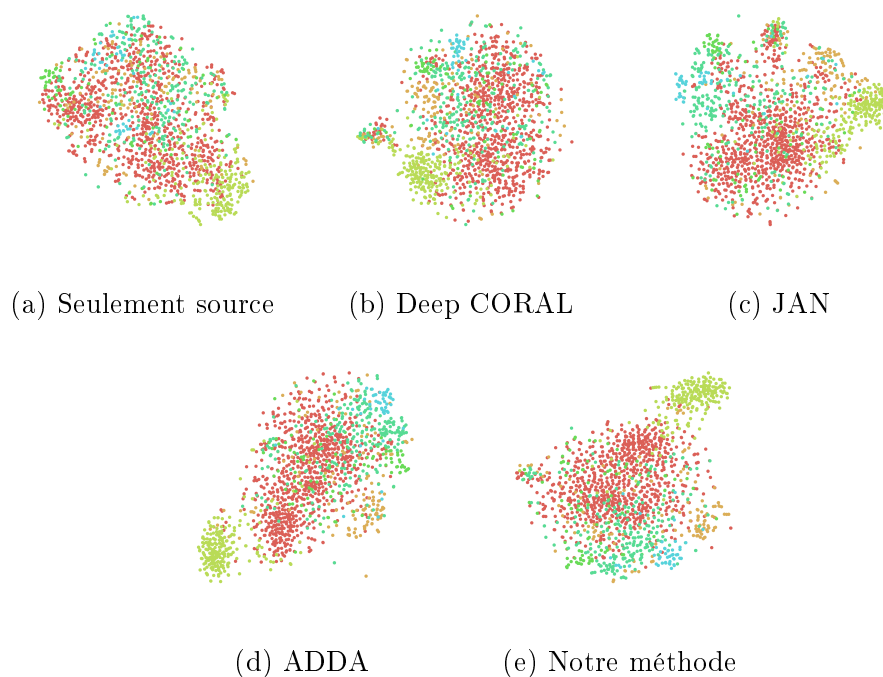


FIGURE 5.17 – Visualisation des caractéristiques extraites des images provenant de domaine cible (variété rouge) en utilisant la méthode t-SNE à partir des différents méthodes (a) Seule-ment source, (b) Deep CORAL, (c) JAN, (d) ADDA and (e) Notre méthode. Chaque couleur représente une classe différente : saine, endommagée, verte, dartrose, gale commune et rhizoctone.

## 5.6 Conclusion

Dans ce chapitre, nous avons présenté une méthode efficace d’adaptation de domaine non supervisée à base d’apprentissage antagoniste. Deux bases de données ont été créées et évaluées. Dans le premier cas, une augmentation de la luminosité a été appliquée aux images originales (sources) afin de créer des images provenant d’un nouveau domaine cible assez similaires aux données source. Dans le deuxième cas, une nouvelle variété de pomme de



terre (peau rouge) a été introduite pour représenter les images cibles significativement différentes des images sources (pommes de terre jaunes).

Afin d'adapter un modèle entraîné avec des images sources au nouvel ensemble d'images cibles, une méthode d'adaptation de domaine a été introduite. Tout d'abord, un réseau entièrement convolutif ( $FCN_s$ ) et un classifieur ( $C_s$ ) ont été entraînés de manière supervisée avec des images sources étiquetées. Ensuite, l'apprentissage antagoniste a été utilisé afin d'obtenir des représentations des images cibles qui ne soient pas distinguables de celles des images sources et d'adapter le classifieur des images sources aux nouvelles images cibles. Compte tenu du fait que les étiquettes des images cibles ne sont pas disponibles, l'adaptation du classifieur est accomplie à l'aide de pseudo-étiquettes. L'avantage de cette approche est de ne pas nécessiter d'étiquettes réelles.

À partir des résultats expérimentaux obtenus, nous avons montré qu'une méthode d'adaptation du domaine est indispensable afin d'obtenir des résultats satisfaisants sur le nouvel ensemble de données cibles. Nous avons également montré que la méthode proposée fondée sur l'apprentissage antagoniste surpasse les approches reposant sur la divergence, telles que Deep CORAL et JAN. De plus, nous avons constaté que les résultats de notre méthode sont comparables à ceux obtenus à partir d'un réseau entraîné d'une manière supervisée dans le scénario d'un changement de luminosité sur les images. Finalement, dans le cas d'une faible différence entre domaines, l'objectif principal d'éviter l'étiquetage manuel des nouvelles images cibles en exploitant les images sources disponibles avec étiquettes a été atteint avec succès. Cependant, dans le cas où la différence entre domaines est plus importante, nous avons pu constater une perte significative de performance. Pour cette raison, nous prévoyons d'étudier par la suite la possibilité de modifier notre méthode pour en faire une approche d'adaptation de domaine semi-supervisée. En effet, la modification à réaliser est simple : il suffit de remplacer les pseudo-étiquettes des images cibles par des étiquettes réelles afin d'améliorer l'adaptation du classifieur dans le domaine cible.

# Chapitre 6

## Conclusions et perspectives

### Sommaire

---

<b>6.1</b>	<b>Rappel de la problématique traitée . . . . .</b>	<b>157</b>
<b>6.2</b>	<b>Réalisations au cours de la thèse . . . . .</b>	<b>158</b>
<b>6.3</b>	<b>Perspectives et approfondissements . . . . .</b>	<b>160</b>

---

### 6.1 Rappel de la problématique traitée

Le contrôle qualité de produits agricoles joue un rôle fondamental dans l'industrie agro-alimentaire. Dans le cas particulier de la pomme de terre, ce contrôle facilite la négociation du prix et le choix du marché auquel la production sera destinée. Actuellement, le contrôle qualité est couramment effectué de manière manuelle par des opérateurs humains experts dans le domaine. En effet, ces opérateurs ont pour objectif de classer avec précision les divers défauts et maladies qui peuvent affecter la qualité du produit analysé. Néanmoins, comme nous avons pu démontrer dans le chapitre 1, les résultats ainsi obtenus manquent de précision et de répétabilité. En effet, trois principaux inconvénients ressortent de cette analyse manuelle. Premièrement, la stratégie décisionnelle d'un même opérateur peut varier dans le temps en raison de la lassitude et de la monotonie de la tâche, ce qui conduit à un manque de répétabilité de résultats. Deuxièmement, la subjectivité des opérateurs lors de la réalisation de l'analyse engendre un manque de crédibilité vis-à-vis des résultats obtenus. Finalement, la tâche manuelle est chronophage, et plus les opérateurs cherchent à être précis, plus ils ont besoin de temps pour analyser chaque produit.

Afin d'améliorer les performances, de faciliter et d'automatiser cette tâche, un vaste éventail de techniques qui utilisent principalement la vision par ordinateur et l'apprentissage automatique ont été développées dans le domaine de l'agriculture. Dans le cadre de ces techniques, nous pouvons trouver des approches traditionnelles reposant sur l'extraction ciblée de caractéristiques, ainsi que des approches plus récentes fondées principalement sur l'apprentissage profond (*Deep Learning*). Malgré la pertinence des travaux exposés dans la littérature, la plupart d'entre eux sont limités en termes de nombre de défauts et/ou maladies détectées. C'est pourquoi nous nous sommes attachés, dans cette

thèse, à proposer un nouveau système de contrôle qualité qui soit complet, flexible et performant. Pour cela, nous avons étudié en détail diverses méthodes de classification, de localisation et de segmentation des défauts de la pomme de terre. Le développement de tels systèmes a impliqué plusieurs réalisations dont nous allons rappeler les principaux éléments dans la section suivante.

## 6.2 Réalisations au cours de la thèse

Dans ce travail de thèse nous avons analysé et proposé une série de méthodes de classification et de localisation de défauts et de maladies sur les pommes de terre fondées sur l'apprentissage profond, mais également sur des techniques de classification traditionnelles. Pour ce faire, nous avons effectué les réalisations suivantes :

- Dans une première partie, nous avons réalisé une étude approfondie de l'état de l'art relatif aux méthodes de classification et de localisation de divers défauts présents sur les produits agricoles. Nous nous sommes concentrés sur les approches les plus adéquats pour notre problème, tant sur les approches traditionnelles reposant principalement sur une extraction ciblée de caractéristiques, que sur les approches plus contemporaines, reposant sur l'apprentissage de caractéristiques. Grâce à cette première partie de la thèse, nous avons analysé les atouts et les limites des différentes approches et, partant de là, nous avons pu définir les pistes clés à suivre pour le développement de nos méthodes.
- Dans une deuxième partie, nous avons proposé une première méthode de classification et de localisation de certains défauts et maladies sur les pommes de terre reposant sur un apprentissage supervisé. Tout d'abord, nous avons créé notre propre base de données étiquetée composée de 9688 images provenant de 2422 tubercules différents. Puis, une méthode divisée en trois étapes a été proposée.

Dans la première phase de la méthode, un réseau de neurones convolutifs pré-entraîné est réglé plus finement grâce à du *finetuning* afin de classer chaque image de pomme de terre en 6 catégories.

Ensuite, une étape de localisation des défauts (endommagements et verts) est effectuée par un auto-encodeur en conjonction avec un classifieur 2C-SVM. En effet, l'auto-encodeur permet d'extraire des attributs compacts et pertinents de patches provenant des images défectueuses et le classifieur 2C-SVM est, quant à lui, utilisé afin de classer chaque patch et ainsi localiser les défauts.

Dans la troisième et dernière étape, les pommes de terre endommagées et vertes sont classées selon la gravité du défaut détecté (léger ou grave). Pour ce faire, les informations relatives aux patches détectés dans l'étape de localisation sont utilisées pour entraîner un classifieur 2C-SVM dont l'objectif est de classer les défauts par gravité.

À travers cette première méthode, nous avons mis en évidence la supériorité des méthodes fondées sur l'apprentissage de caractéristiques, notamment en utilisant des

architectures profondes (CNN), par rapport à celles reposant sur l'extraction ciblée de caractéristiques. De plus, nous avons constaté l'importance de disposer d'une base de données étiquetée par patches afin d'atteindre des résultats de localisation de défauts précis, et par conséquent, classer correctement les défauts par gravité.

- Malgré les résultats encourageants atteints avec la première méthode proposée, le fait qu'elle soit dépendante d'un ensemble de données étiquetées par patches nous a poussé à analyser des approches alternatives reposant sur l'apprentissage faiblement supervisé. L'idée clé de ces approches est d'utiliser une base de données étiquetées uniquement au niveau de l'image afin d'obtenir des résultats plus complets au niveau du pixel. Dans cette troisième partie, nous avons donc développé une méthode fondée sur l'apprentissage faiblement supervisé afin de segmenter les défauts et les maladies sur les pommes de terre. Bien que la phase de classification des images soit restée identique par rapport à la première méthode, nous avons ajouté deux nouvelles étapes. La première étape vise à localiser les défauts et la seconde à segmenter finement les défauts précédemment localisés.

Dans l'étape de localisation de défauts, les différents motifs de l'image favorisant la décision proposée par le CNN sont identifiés. Ensuite, dans l'étape de segmentation, un classifieur OC-SVM est utilisé afin de détecter au sein de ces régions tous les pixels anormaux.

Comme cette approche faiblement supervisée a donné des résultats de segmentation satisfaisants, nous avons décidé d'en tirer parti en l'utilisant pour créer de manière automatique une base de données étiquetée au niveau du pixel. En utilisant cette nouvelle base de données, nous avons entraîné un nouveau réseau de neurones convolutifs afin de classer et de segmenter chaque image de pomme de terre. En effet, ce réseau a été développé pour deux raisons principales : à partir d'un unique réseau entraîné de bout en bout nous obtenons une segmentation de défauts précise et le temps de calcul dans la phase de décision est réduit, ce qui est fondamental dans un environnement industriel.

Grâce aux résultats expérimentaux obtenus comparables aux résultats en mode supervisée, nous avons montré la pertinence des approches faiblement supervisées pour la segmentation de divers défauts et maladies de la pomme de terre. En effet, s'affranchir de créer une base de données étiquetées manuellement à l'échelle du patch ou du pixel pour obtenir une segmentation précise des défauts rend notre méthode flexible et bien plus simple à mettre en œuvre.

- Dans le contexte industriel dans lequel cette thèse a été réalisée, la flexibilité, l'adaptabilité et la robustesse des systèmes proposés a toujours été d'une grande importance. Par conséquent, dans une quatrième partie, nous avons poursuivi nos travaux en proposant une méthode d'adaptation de domaine non supervisée.

Une fois la machine équipée de l'algorithme, elle doit être opérationnelle dans le temps en terme de classification et de localisation de défauts, conformément aux niveaux des performances initiales de l'algorithme. Toutefois, il est très possible qu'au cours du temps, certaines conditions viennent à changer dans le contexte d'utilisation

de la machine. Il est possible, en effet, qu'il faille changer totalement ou partiellement le système d'acquisition d'images, auquel cas des problèmes de calibrage ou d'étalonnage peuvent perturber plus ou moins fortement les données transmises à l'algorithme. Il se peut aussi que l'on ait à un moment à analyser des variétés de pommes de terre différentes de celles qui ont contribué à la mise au point de l'algorithme.

Dans le cadre de ces hypothèses, nous avons donc proposé une méthode d'adaptation de domaine non supervisée, fondée sur l'apprentissage antagoniste afin de réduire au maximum la possible baisse de performances causée par des évolutions. L'objectif principal de cette nouvelle approche est d'adapter un réseau entraîné avec des images initiales provenant d'un domaine source, à de nouvelles images d'un domaine différent dit cible, sans utiliser les étiquettes de ces nouvelles images. Grâce aux résultats expérimentaux obtenus, nous avons montré la pertinence de la méthode proposée dans deux situations où il existait une différence limitée et significative respectivement entre le domaine source et le domaine cible.

### 6.3 Perspectives et approfondissements

Dans le cadre de cette thèse, nous avons étudié, proposé et évalué un grand nombre de solutions au problème initial. Néanmoins, il demeure quelques questions d'un grand intérêt pour des travaux futurs :

- Une amélioration du système d'acquisition d'images pourrait être envisagée afin d'assurer une couverture à 100% de la surface des pommes de terre. En travaillant avec nos images actuelles, les défauts trouvés aux extrémités des tubercules ne sont pas visibles. Une solution possible à ce problème est l'utilisation de plus d'une caméra à partir desquelles on peut obtenir des images provenant de plusieurs angles de chaque tubercule. L'utilisation d'images 3D peut être également une piste intéressante à explorer afin de faciliter l'analyse de toute la surface de la pomme de terre en une seule fois, y compris les extrémités.
- Dans la classification de défauts et de maladies, le réseau de neurones convolutifs (CNN) a été entraîné à partir d'une base de données déséquilibrée. Cela pourrait avoir un effet négatif sur la précision des classes minoritaires. Les techniques d'ajustement de poids (*weight balancing*), de sur-échantillonnage (*oversampling*) et de sous-échantillonnage (*undersampling*) pourraient être étudiées et évaluées en profondeur afin d'analyser leur utilité dans notre problématique.
- Dans le chapitre 4 de cette thèse, nous avons proposé d'entraîner un réseau de neurones convolutifs à deux sorties (FC-GoogLeNet) avec des images étiquetées automatiquement en utilisant notre méthode de segmentation faiblement supervisée. Cependant, cette base de données est susceptible de contenir des erreurs d'étiquettes qui peuvent nuire aux performances du FC-GoogLeNet. Afin d'améliorer la perfor-

mance du réseau, nous pourrions adopter une approche semi-supervisée et construire une base de données avec peu d'images annotées correctement par des opérateurs experts. En effet, en combinant les étiquettes plus précises créées par les experts et celles un peu moins précises créées par notre méthode, nous pourrions sans doute obtenir des résultats de segmentation plus performants sans avoir besoin d'annoter manuellement au niveau du pixel une grande quantité d'images.

- Dans le chapitre 5, nous avons constaté que l'efficacité des approches d'adaptation de domaine dépend fortement du degré de décalage existant entre les domaines source et cible. De ce fait, nous pourrions envisager d'exploiter notre approche d'adaptation de domaine d'une manière semi-supervisée afin de résoudre les problèmes d'adaptation plus complexes quand la différence entre domaines est importante. De plus, l'approche d'adaptation de domaine proposée est focalisée sur la classification d'images. Il reste à étudier l'adaptation de domaine dans le contexte de la segmentation d'images et de la classification de défauts par gravité.
- Dans cette thèse, le réseau GoogLeNet a été utilisé pour la classification d'images. Toutefois, des réseaux plus récents et plus performants tels que Inception-v4 [220], Xception [221] et ResNeXt-50[222] devraient être évalués pour déterminer si une amélioration des performances est possible.
- Bien que les méthodes proposées se soient avérées efficaces pour classer et localiser certains défauts et maladies des pommes de terre, il est important d'évaluer leur adaptabilité à de nouveaux défauts externes qui doivent être détectés. À terme, nous pourrions également confirmer la pertinence des méthodes proposées dans le contrôle qualité d'autres produits agricoles tels que les carottes, les oignons, les raisins, etc.



# Bibliographie

- [1] Y. LeCun, “L’apprentissage profond, une révolution en intelligence artificielle,” *La lettre du Collège de France*, no. 41, p. 13, 2016.
- [2] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, pp. 1097–1105, 2012.
- [4] S. Shehata, *Using mid- and high-level visual features for surgical workflow detection in cholecystectomy procedures*. PhD thesis, 07 2016.
- [5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.
- [6] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.
- [7] H. Noh, S. Hong, and B. Han, “Learning deconvolution network for semantic segmentation,” in *Proceedings of the IEEE international conference on computer vision*, pp. 1520–1528, 2015.
- [8] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, “Deep domain confusion : Maximizing for domain invariance,” *arXiv preprint arXiv :1412.3474*, 2014.
- [9] M. Long, Y. Cao, J. Wang, and M. I. Jordan, “Learning transferable features with deep adaptation networks,” in *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, pp. 97–105, JMLR. org, 2015.
- [10] M. Long, H. Zhu, J. Wang, and M. I. Jordan, “Deep transfer learning with joint adaptation networks,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2208–2217, JMLR. org, 2017.
- [11] B. Sun and K. Saenko, “Deep coral : Correlation alignment for deep domain adaptation,” in *European Conference on Computer Vision*, pp. 443–450, Springer, 2016.
- [12] M.-Y. Liu and O. Tuzel, “Coupled generative adversarial networks,” in *Advances in neural information processing systems*, pp. 469–477, 2016.



- [13] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, “Unsupervised pixel-level domain adaptation with generative adversarial networks,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [14] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, “Domain-adversarial training of neural networks,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.
- [15] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, “Domain separation networks,” in *Advances in Neural Information Processing Systems*, pp. 343–351, 2016.
- [16] FAOSTAT, “Food and agriculture organization of the united nations.” <http://www.fao.org>, 2017. Accessed : 2019-05-23.
- [17] B. K. Miller and M. J. Delwiche, “A color vision system for peach grading,” *Transactions of the ASAE*, vol. 32, no. 4, pp. 1484–1490, 1989.
- [18] Y. Tao, P. Heinemann, Z. Varghese, C. Morrow, and H. Sommer Iii, “Machine vision for color inspection of potatoes and apples,” *Transactions of the ASAE*, vol. 38, no. 5, pp. 1555–1561, 1995.
- [19] R. M. Bolle, J. H. Connell, N. Haas, R. Mohan, and G. Taubin, “Veggievision : A produce recognition system,” in *Applications of Computer Vision, 1996. WACV’96., Proceedings 3rd IEEE Workshop on*, pp. 244–251, IEEE, 1996.
- [20] J. Blasco, N. Aleixos, J. Gómez-Sanchis, and E. Moltó, “Recognition and classification of external skin damage in citrus fruits using multispectral data and morphological features,” *Biosystems engineering*, vol. 103, no. 2, pp. 137–145, 2009.
- [21] M. Barnes, T. Duckett, G. Cielniak, G. Stroud, and G. Harper, “Visual detection of blemishes in potatoes using minimalist boosted classifiers,” *Journal of Food Engineering*, vol. 98, no. 3, pp. 339–346, 2010.
- [22] N. Razmjoo, B. S. Mousavi, and F. Soleymani, “A real-time mathematical computer method for potato inspection using machine vision,” *Computers & Mathematics with Applications*, vol. 63, no. 1, pp. 268–279, 2012.
- [23] G. ElMasry, S. Cubero, E. Moltó, and J. Blasco, “In-line sorting of irregular potatoes by using automated computer-based machine vision system,” *Journal of Food Engineering*, vol. 112, no. 1-2, pp. 60–68, 2012.
- [24] M. Brahim, K. Boukhalfa, and A. Moussaoui, “Deep learning for tomato diseases : classification and symptoms visualization,” *Applied Artificial Intelligence*, vol. 31, no. 4, pp. 299–315, 2017.
- [25] J. Ma, K. Du, F. Zheng, L. Zhang, Z. Gong, and Z. Sun, “A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network,” *Computers and Electronics in Agriculture*, vol. 154, pp. 18–24, 2018.
- [26] A. Picon, A. Alvarez-Gila, M. Seitz, A. Ortiz-Barredo, J. Echazarra, and A. Johannes, “Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild,” *Computers and Electronics in Agriculture*, 2018.

- 
- [27] S. Zhang, S. Zhang, C. Zhang, X. Wang, and Y. Shi, "Cucumber leaf disease identification with global pooling dilated convolutional neural network," *Computers and Electronics in Agriculture*, vol. 162, pp. 422–430, 2019.
- [28] J. Heikkila, O. Silven, *et al.*, "A four-step camera calibration procedure with implicit image correction," in *cvpr*, vol. 97, p. 1106, Citeseer, 1997.
- [29] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani, "Person re-identification by symmetry-driven accumulation of local features," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2360–2367, IEEE, 2010.
- [30] S. Bouindour, H. Snoussi, M. M. Hittawe, N. Tazi, and T. Wang, "An on-line and adaptive method for detecting abnormal events in videos using spatio-temporal convnet," *Applied Sciences*, vol. 9, no. 4, p. 757, 2019.
- [31] J. Seo, S. Han, S. Lee, and H. Kim, "Computer vision techniques for construction safety and health monitoring," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 239–251, 2015.
- [32] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. Van Der Laak, B. Van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical image analysis*, vol. 42, pp. 60–88, 2017.
- [33] C.-H. Pham, C. Tor-Díez, H. Meunier, N. Bednarek, R. Fablet, N. Passat, and F. Rousseau, "Multiscale brain mri super-resolution using deep 3d convolutional networks," *Computerized Medical Imaging and Graphics*, vol. 77, p. 101647, 2019.
- [34] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of machine learning research*, vol. 12, no. Aug, pp. 2493–2537, 2011.
- [35] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649, IEEE, 2013.
- [36] X. Zheng, H. Chen, and T. Xu, "Deep learning for chinese word segmentation and pos tagging," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 647–657, 2013.
- [37] J. Kim and J. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," in *Proceedings of the IEEE international conference on computer vision*, pp. 2942–2950, 2017.
- [38] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest : Automated testing of deep-neural-network-driven autonomous cars," in *Proceedings of the 40th international conference on software engineering*, pp. 303–314, ACM, 2018.
- [39] M. Bramberger, J. Brunner, B. Rinner, and H. Schwabach, "Real-time video analysis on an embedded smart camera for traffic surveillance," in *Proceedings. RTAS 2004. 10th IEEE Real-Time and Embedded Technology and Applications Symposium, 2004.*, pp. 174–181, IEEE, 2004.
- [40] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data : a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2014.

- [41] A. Mathews and J. Jensen, “Visualizing and quantifying vineyard canopy lai using an unmanned aerial vehicle (uav) collected high density structure from motion point cloud,” *Remote Sensing*, vol. 5, no. 5, pp. 2164–2183, 2013.
- [42] A.-K. Mahlein, “Plant disease detection by imaging sensors—parallels and specific demands for precision agriculture and plant phenotyping,” *Plant disease*, vol. 100, no. 2, pp. 241–251, 2016.
- [43] F. Rançon, L. Bombrun, B. Keresztes, and C. Germain, “Comparison of sift encoded and deep learning features for the classification and detection of esca disease in bordeaux vineyards,” *Remote Sensing*, vol. 11, no. 1, p. 1, 2019.
- [44] M. A. Ayub, A. B. Mohamed, and A. H. Esa, “In-line inspection of roundness using machine vision,” *Procedia Technology*, vol. 15, pp. 807–816, 2014.
- [45] L. Li, K. Ota, and M. Dong, “Deep learning for smart industry : Efficient manufacture inspection system with fog computing,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4665–4673, 2018.
- [46] D. Unay and B. Gosselin, “Artificial neural network-based segmentation and apple grading by machine vision,” in *IEEE International Conference on Image Processing 2005*, vol. 2, pp. II–630, IEEE, 2005.
- [47] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv :1409.1556*, 2014.
- [48] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, “Residual attention network for image classification,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3156–3164, 2017.
- [49] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn : Towards real-time object detection with region proposal networks,” in *Advances in neural information processing systems*, pp. 91–99, 2015.
- [50] J. Dai, Y. Li, K. He, and J. Sun, “R-fcn : Object detection via region-based fully convolutional networks,” in *Advances in neural information processing systems*, pp. 379–387, 2016.
- [51] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once : Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.
- [52] G. Lin, A. Milan, C. Shen, and I. Reid, “Refinenet : Multi-path refinement networks for high-resolution semantic segmentation,” in *IEEE conference on computer vision and pattern recognition (CVPR)*, vol. 1, p. 3, 2017.
- [53] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 2881–2890, 2017.
- [54] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab : Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

- 
- [55] E. Denton, S. Chintala, A. Szlam, and R. Fergus, "Deep generative image models using a laplacian pyramid of adversarial networks," in *29th Annual Conference on Neural Information Processing Systems, NIPS 2015*, pp. 1486–1494, Neural information processing systems foundation, 2015.
- [56] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw : A recurrent neural network for image generation," *arXiv preprint arXiv :1502.04623*, 2015.
- [57] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski, "Plug & play generative networks : Conditional iterative generation of images in latent space," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4467–4477, 2017.
- [58] I. Paulus, R. De Busscher, and E. Schrevens, "Use of image analysis to investigate human quality classification of apples," *Journal of Agricultural Engineering Research*, vol. 68, no. 4, pp. 341–353, 1997.
- [59] J. Amara, B. Bouaziz, A. Algergawy, *et al.*, "A deep learning-based approach for banana leaf diseases classification.," in *BTW (Workshops)*, pp. 79–88, 2017.
- [60] Y. Tian, G. Yang, Z. Wang, E. Li, and Z. Liang, "Detection of apple lesions in orchards based on deep learning methods of cyclegan and yolov3-dense," *Journal of Sensors*, vol. 2019, 2019.
- [61] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.
- [62] V. Leemans and M.-F. Destain, "A real-time grading method of apples based on features extracted from defects," *Journal of Food Engineering*, vol. 61, no. 1, pp. 83–89, 2004.
- [63] A. Brar and K. Singh, "Potato defect detection using fuzzy c-mean clustering based segmentation," *Indian Journal of Science and Technology*, vol. 9, no. 32, pp. 1–6, 2016.
- [64] A. Fuentes, S. Yoon, S. Kim, and D. Park, "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition," *Sensors*, vol. 17, no. 9, p. 2022, 2017.
- [65] S. Bargoti and J. Underwood, "Deep fruit detection in orchards," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3626–3633, IEEE, 2017.
- [66] J. Li, X. Rao, and Y. Ying, "Detection of common defects on oranges using hyperspectral reflectance imaging," *Computers and Electronics in Agriculture*, vol. 78, no. 1, pp. 38–48, 2011.
- [67] D. Unay, B. Gosselin, O. Kleynen, V. Leemans, M.-F. Destain, and O. Debeir, "Automatic grading of bi-colored apples by multispectral machine vision," *Computers and electronics in agriculture*, vol. 75, no. 1, pp. 204–212, 2011.
- [68] R. Lu and D. P. Ariana, "Detection of fruit fly infestation in pickling cucumbers using a hyperspectral reflectance/transmittance imaging system," *Postharvest Biology and Technology*, vol. 81, pp. 44–50, 2013.

- [69] R. Hassankhani, H. Navid, H. Seyedarabi, *et al.*, “Potato surface defect detection in machine vision system.,” *African Journal of Agricultural Research*, vol. 7, no. 5, pp. 844–850, 2012.
- [70] X. Sun, S. Mu, Y. Xu, Z. Cao, and T. Su, “Image recognition of tea leaf diseases based on convolutional neural network,” *arXiv preprint arXiv :1901.02694*, 2019.
- [71] J. C. Noordam, G. W. Otten, T. J. Timmermans, and B. H. van Zwol, “High-speed potato grading and quality inspection based on a color vision system,” in *Machine Vision Applications in Industrial Inspection VIII*, vol. 3966, pp. 206–217, International Society for Optics and Photonics, 2000.
- [72] T. Pearson and N. Toyofuku, “Automated sorting of pistachio nuts with closed shells,” *Applied Engineering in Agriculture*, vol. 16, no. 1, p. 91, 2000.
- [73] R. S. Jadhav and S. Patil, “A fruit quality management system based on image processing,” *IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)*, vol. 8, no. 6, pp. 01–05, 2013.
- [74] O. K. M. Yahaya, M. Z. MatJafri, A. A. Aziz, and A. F. Omar, “Non-destructive quality evaluation of fruit by color based on rgb leds system,” in *2014 2nd International Conference on Electronic Design (ICED)*, pp. 230–233, IEEE, 2014.
- [75] E. Saldaña, R. Siche, M. Luján, and R. Quevedo, “Computer vision applied to the inspection and quality control of fruits and vegetables,” *Brazilian journal of food technology*, vol. 16, no. 4, pp. 254–272, 2013.
- [76] J. Blasco, N. Aleixos, J. Gómez, and E. Moltó, “Citrus sorting by identification of the most common defects using multispectral computer vision,” *Journal of food engineering*, vol. 83, no. 3, pp. 384–393, 2007.
- [77] D. P. Ariana and R. Lu, “Hyperspectral waveband selection for internal defect detection of pickling cucumbers and whole pickles,” *Computers and Electronics in Agriculture*, vol. 74, no. 1, pp. 137–144, 2010.
- [78] U.-O. Dorj, M. Lee, and S.-s. Yun, “An yield estimation in citrus orchards via fruit detection and counting using image processing,” *Computers and Electronics in Agriculture*, vol. 140, pp. 103–112, 2017.
- [79] W. Ming, J. Du, D. Shen, Z. Zhang, X. Li, J. R. Ma, F. Wang, and J. Ma, “Visual detection of sprouting in potatoes using ensemble-based classifier,” *Journal of food process engineering*, vol. 41, no. 3, p. e12667, 2018.
- [80] G. Dhingra, V. Kumar, and H. D. Joshi, “A novel computer vision based neutrosophic approach for leaf disease identification and classification,” *Measurement*, vol. 135, pp. 782–794, 2019.
- [81] M. R. Paulsen and W. McClure, “Illumination for computer vision systems,” *Transactions of the ASAE*, vol. 29, no. 5, pp. 1398–1404, 1986.
- [82] M.-h. Hu, Q.-l. Dong, B.-l. Liu, and P. K. Malakar, “The potential of double k-means clustering for banana image segmentation,” *Journal of Food Process Engineering*, vol. 37, no. 1, pp. 10–18, 2014.
- [83] B. G. Batchelor, *Machine Vision Handbook*. Springer, 2012.

- 
- [84] B. Zhang, W. Huang, J. Li, C. Zhao, S. Fan, J. Wu, and C. Liu, "Principles, developments and applications of computer vision for external quality inspection of fruits and vegetables : A review," *Food Research International*, vol. 62, pp. 326–343, 2014.
- [85] J. Blasco, N. Aleixos, S. Cubero, J. Gómez-Sanchís, and E. Moltó, "Automatic sorting of satsuma (citrus unshiu) segments using computer vision and morphological features," *Computers and electronics in agriculture*, vol. 66, no. 1, pp. 1–8, 2009.
- [86] T. Hakala, E. Honkavaara, H. Saari, J. Mäkynen, J. Kaivosoja, L. Pesonen, and I. Pölönen, "Spectral imaging from uavs under varying illumination conditions," in *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, International Society for Photogrammetry and Remote Sensing (ISPRS), 2013.
- [87] L. Yao, L. Lu, and R. Zheng, "Study on detection method of external defects of potato image in visible light environment," in *2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pp. 118–122, IEEE, 2017.
- [88] F. Pedreschi, J. Leon, D. Mery, and P. Moyano, "Development of a computer vision system to measure the color of potato chips," *Food Research International*, vol. 39, no. 10, pp. 1092–1098, 2006.
- [89] A. Dacal-Nieto, A. Formella, P. Carrión, E. Vazquez-Fernandez, and M. Fernández-Delgado, "Common scab detection on potatoes using an infrared hyperspectral imaging system," in *International conference on image analysis and processing*, pp. 303–312, Springer, 2011.
- [90] Q. Su, N. Kondo, M. Li, H. Sun, D. F. Al Riza, and H. Habaragamuwa, "Potato quality grading based on machine vision and 3d shape analysis," *Computers and electronics in agriculture*, vol. 152, pp. 261–268, 2018.
- [91] F. López-García, G. Andreu-García, J. Blasco, N. Aleixos, and J.-M. Valiente, "Automatic detection of skin defects in citrus fruits using a multivariate image analysis approach," *Computers and Electronics in Agriculture*, vol. 71, no. 2, pp. 189–197, 2010.
- [92] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in plant science*, vol. 7, p. 1419, 2016.
- [93] T.-T. Le, C.-Y. Lin, *et al.*, "Deep learning for noninvasive classification of clustered horticultural crops—a case for banana fruit tiers," *Postharvest Biology and Technology*, vol. 156, p. 110922, 2019.
- [94] D. S. Prabha and J. S. Kumar, "Assessment of banana fruit maturity by image processing technique," *Journal of food science and technology*, vol. 52, no. 3, pp. 1316–1327, 2015.
- [95] A. Johannes, A. Picon, A. Alvarez-Gila, J. Echazarra, S. Rodriguez-Vaamonde, A. D. Navajas, and A. Ortiz-Barredo, "Automatic plant disease diagnosis using mobile capture devices, applied on a wheat use case," *Computers and electronics in agriculture*, vol. 138, pp. 200–209, 2017.

- [96] S. Jana, S. Basak, and R. Parekh, “Automatic fruit recognition from natural images using color and texture features,” in *2017 Devices for Integrated Circuit (DevIC)*, pp. 620–624, IEEE, 2017.
- [97] T. R. Chavan and A. V. Nandedkar, “Agroavnet for crops and weeds classification : A step forward in automatic farming,” *Computers and electronics in agriculture*, vol. 154, pp. 361–372, 2018.
- [98] N. Kondo, “Automation on fruit and vegetable grading system and food traceability,” *Trends in Food Science & Technology*, vol. 21, no. 3, pp. 145–152, 2010.
- [99] S. G. Kandi, “Automatic defect detection and grading of single-color fruits using hsv (hue, saturation, value) color space,” *Journal of Life Science*, vol. 4, no. 7, pp. 39–45, 2010.
- [100] E. Chavolla, D. Zaldivar, E. Cuevas, and M. A. Perez, “Color spaces advantages and disadvantages in image color clustering segmentation,” in *Advances in Soft Computing and Machine Learning in Image Processing*, pp. 3–22, Springer, 2018.
- [101] W. Dana and W. Ivo, “Computer image analysis of seed shape and seed color for flax cultivar description,” *Computers and electronics in agriculture*, vol. 61, no. 2, pp. 126–135, 2008.
- [102] X. Liming and Z. Yanchao, “Automated strawberry grading system based on image processing,” *Computers and Electronics in Agriculture*, vol. 71, pp. S32–S39, 2010.
- [103] S. Cárdenas-Pérez, J. Chanona-Pérez, J. V. Méndez-Méndez, G. Calderón-Domínguez, R. López-Santiago, M. J. Perea-Flores, and I. Arzate-Vázquez, “Evaluation of the ripening stages of apple (golden delicious) by means of computer vision system,” *biosystems engineering*, vol. 159, pp. 46–58, 2017.
- [104] M. Riquelme, P. Barreiro, M. Ruiz-Altisent, and C. Valero, “Olive classification according to external damage using image analysis,” *Journal of Food Engineering*, vol. 87, no. 3, pp. 371–379, 2008.
- [105] D. L. Hernández-Rabadán, F. Ramos-Quintana, and J. Guerrero Juk, “Integrating soms and a bayesian classifier for segmenting diseased plants in uncontrolled environments,” *The Scientific World Journal*, vol. 2014, 2014.
- [106] P. Moallem, A. Serajoddin, and H. Pourghassem, “Computer vision-based apple grading for golden delicious apples based on surface features,” *Information Processing in Agriculture*, vol. 4, no. 1, pp. 33–40, 2017.
- [107] F. Mendoza, P. Dejmek, and J. M. Aguilera, “Calibrated color measurements of agricultural foods using image analysis,” *Postharvest Biology and Technology*, vol. 41, no. 3, pp. 285–295, 2006.
- [108] L. F. S. Pereira, S. Barbon Jr, N. A. Valous, and D. F. Barbin, “Predicting the ripening of papaya fruit with digital imaging and random forests,” *Computers and Electronics in Agriculture*, vol. 145, pp. 76–82, 2018.
- [109] M. Brahim, M. Arsenovic, S. Laraba, S. Sladojevic, K. Boukhalfa, and A. Mousaoui, “Deep learning for plant diseases : detection and saliency map visualisation,” in *Human and Machine Learning*, pp. 93–117, Springer, 2018.

- 
- [110] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern recognition*, vol. 26, no. 9, pp. 1277–1294, 1993.
- [111] T. Brosnan and D.-W. Sun, "Improving quality inspection of food products by computer vision—a review," *Journal of food engineering*, vol. 61, no. 1, pp. 3–16, 2004.
- [112] Y. Lanthier, A. Bannari, D. Haboudane, J. R. Miller, and N. Tremblay, "Hyperspectral data segmentation and classification in precision agriculture : A multi-scale analysis," in *IGARSS 2008-2008 IEEE International Geoscience and Remote Sensing Symposium*, vol. 2, pp. II–585, IEEE, 2008.
- [113] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.
- [114] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.
- [115] R. Thendral, A. Suhasini, and N. Senthil, "A comparative analysis of edge and color based segmentation for orange fruit recognition," in *2014 International Conference on Communication and Signal Processing*, pp. 463–466, IEEE, 2014.
- [116] D. Unay and B. Gosselin, "Apple defect detection and quality classification with mlp-neural networks," in *Proceedings of the ProRISC Workshop on Circuits, Systems and Signal Processing*, Citeseer, 2002.
- [117] D. Unay and B. Gosselin, "Automatic defect segmentation of 'jonagold' apples on multi-spectral images : A comparative study," *Postharvest Biology and Technology*, vol. 42, no. 3, pp. 271–279, 2006.
- [118] M. P. Arakeri *et al.*, "Computer vision based fruit grading system for quality evaluation of tomato in agriculture industry," *Procedia Computer Science*, vol. 79, pp. 426–433, 2016.
- [119] M. Islam, A. Dinh, K. Wahid, and P. Bhowmik, "Detection of potato diseases using image segmentation and multiclass support vector machine," in *Electrical and Computer Engineering (CCECE), 2017 IEEE 30th Canadian Conference on*, pp. 1–4, IEEE, 2017.
- [120] P. Ninawe and S. Pandey, "A completion on fruit recognition system using k-nearest neighbors algorithm," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 3, no. 7, pp. 2352–2356, 2014.
- [121] Z.-C. Huang, P. P. Chan, W. W. Ng, and D. S. Yeung, "Content-based image retrieval using color moment and gabor texture feature," in *2010 International Conference on Machine Learning and Cybernetics*, vol. 2, pp. 719–724, IEEE, 2010.
- [122] R. Brunelli and O. Mich, "Histograms analysis for image retrieval," *Pattern Recognition*, vol. 34, no. 8, pp. 1625–1637, 2001.
- [123] M. A. Stricker and M. Orengo, "Similarity of color images," in *Storage and retrieval for image and video databases III*, vol. 2420, pp. 381–392, International Society for Optics and Photonics, 1995.
- [124] D.-C. He, L. Wang, and J. Guibert, "Texture feature extraction," *Pattern recognition letters*, vol. 6, no. 4, pp. 269–273, 1987.



- [125] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern recognition*, vol. 29, no. 1, pp. 51–59, 1996.
- [126] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, no. 6, pp. 610–621, 1973.
- [127] D. Gabor, "Theory of communication. part 1 : The analysis of information," *Journal of the Institution of Electrical Engineers-Part III : Radio and Communication Engineering*, vol. 93, no. 26, pp. 429–441, 1946.
- [128] M. Yang, K. Kpalma, and J. Ronsin, "A survey of shape feature extraction techniques," 2008.
- [129] E. Alpaydin, *Introduction to machine learning*. MIT press, 2009.
- [130] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [131] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, 1992.
- [132] V. Vapnik, "Estimation of dependences based on empirical data [in russian] 1979," *English Translation by Kotz, Samuel in*, 1982.
- [133] M. A. Aizerman, "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and remote control*, vol. 25, pp. 821–837, 1964.
- [134] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.
- [135] P. C. Mahalanobis, "On the generalized distance in statistics," National Institute of Science of India, 1936.
- [136] D. E. Rumelhart, G. E. Hinton, R. J. Williams, *et al.*, "Learning representations by back-propagating errors," *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.
- [137] D. Opitz and R. Maclin, "Popular ensemble methods : An empirical study," *Journal of artificial intelligence research*, vol. 11, pp. 169–198, 1999.
- [138] P. Moallem, N. Razmjooy, and M. Ashourian, "Computer vision-based potato defect detection using neural networks and support vector machine," *International Journal of Robotics and Automation*, vol. 28, no. 2, pp. 137–145, 2013.
- [139] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.
- [140] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *International conference on artificial neural networks*, pp. 92–101, Springer, 2010.

- 
- [141] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet : A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.
- [142] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” in *Advances in neural information processing systems*, pp. 3320–3328, 2014.
- [143] D. Hughes, M. Salathé, *et al.*, “An open access repository of images on plant health to enable the development of mobile disease diagnostics,” *arXiv preprint arXiv :1511.08060*, 2015.
- [144] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [145] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd : Single shot multibox detector,” in *European conference on computer vision*, pp. 21–37, Springer, 2016.
- [146] P. O. Pinheiro and R. Collobert, “From image-level to pixel-level labeling with convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1713–1721, 2015.
- [147] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille, “Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation,” in *Proceedings of the IEEE international conference on computer vision*, pp. 1742–1750, 2015.
- [148] A. Kolesnikov and C. H. Lampert, “Seed, expand and constrain : Three principles for weakly-supervised image segmentation,” in *European Conference on Computer Vision*, pp. 695–711, Springer, 2016.
- [149] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2921–2929, 2016.
- [150] Y. Zhou, Y. Zhu, Q. Ye, Q. Qiu, and J. Jiao, “Weakly supervised instance segmentation using class peak response,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3791–3800, 2018.
- [151] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Is object localization for free ?-weakly-supervised learning with convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 685–694, 2015.
- [152] L. Bazzani, A. Bergamo, D. Anguelov, and L. Torresani, “Self-taught object localization with deep networks,” in *2016 IEEE winter conference on applications of computer vision (WACV)*, pp. 1–9, IEEE, 2016.
- [153] J. Lu, J. Hu, G. Zhao, F. Mei, and C. Zhang, “An in-field automatic wheat disease diagnosis system,” *Computers and electronics in agriculture*, vol. 142, pp. 369–379, 2017.
- [154] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *European conference on computer vision*, pp. 818–833, Springer, 2014.

- [155] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
- [156] H. Hotelling, “Analysis of a complex of statistical variables into principal components.,” *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.
- [157] A. Ng, “Cs294a lecture notes : Sparse autoencoder,” 2010.
- [158] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked denoising autoencoders : Learning useful representations in a deep network with a local denoising criterion,” *Journal of machine learning research*, vol. 11, no. Dec, pp. 3371–3408, 2010.
- [159] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [160] C. Hung, J. Nieto, Z. Taylor, J. Underwood, and S. Sukkarieh, “Orchard fruit segmentation using multi-spectral feature learning,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5314–5320, IEEE, 2013.
- [161] M.-M. Yang, A. Nayeem, and L.-L. Shen, “Plant classification based on stacked autoencoder,” in *2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pp. 1082–1086, IEEE, 2017.
- [162] H. F. Pardede, E. Suryawati, R. Sustika, and V. Zilvan, “Unsupervised convolutional autoencoder-based feature learning for automatic detection of plant diseases,” in *2018 Ibrahim2017deepnernational Conference on Computer, Control, Informatics and its Applications (IC3INA)*, pp. 158–162, IEEE, 2018.
- [163] M. Abbaszadeh, A. Rahimifard, M. Eftekhari, H. G. Zadeh, A. Fayazi, A. Dini, and M. Danaeian, “Deep learning-based classification of the defective pistachios via deep autoencoder neural networks,” *arXiv preprint arXiv :1906.11878*, 2019.
- [164] E. Bellocchio, T. A. Ciarfuglia, G. Costante, and P. Valigi, “Weakly supervised fruit counting for yield estimation using spatial consistency,” *IEEE Robotics and Automation Letters*, 2019.
- [165] V. Kulikov, V. Yurchenko, and V. Lempitsky, “Instance segmentation by deep coloring,” *arXiv preprint arXiv :1807.10007*, 2018.
- [166] R. Bellman, “Dynamic programming,” *Science*, vol. 153, no. 3731, pp. 34–37, 1966.
- [167] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, “Handwritten digit recognition with a back-propagation network,” in *Advances in neural information processing systems*, pp. 396–404, 1990.
- [168] S. Ruder, “An overview of gradient descent optimization algorithms,” *arXiv preprint arXiv :1609.04747*, 2016.
- [169] M. Janzamin, H. Sedghi, and A. Anandkumar, “Beating the perils of non-convexity : Guaranteed training of neural networks using tensor methods,” *arXiv preprint arXiv :1506.08473*, 2015.
- [170] R. Hecht-Nielsen, “Theory of the backpropagation neural network,” in *Neural networks for perception*, pp. 65–93, Elsevier, 1992.
- [171] T. Dozat, “Incorporating nesterov momentum into adam,” 2016.

- 
- [172] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization,” *Journal of machine learning research*, vol. 12, no. Jul, pp. 2121–2159, 2011.
- [173] G. Hinton, N. Srivastava, and K. Swersky, “Neural networks for machine learning lecture 6a overview of mini-batch gradient descent,” *Cited on*, vol. 14, no. 8, 2012.
- [174] D. P. Kingma and J. Ba, “Adam : A method for stochastic optimization,” *arXiv preprint arXiv :1412.6980*, 2014.
- [175] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103, 2008.
- [176] K. Hornik, M. Stinchcombe, H. White, *et al.*, “Multilayer feedforward networks are universal approximators.,” *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.
- [177] A. Krogh and J. A. Hertz, “A simple weight decay can improve generalization,” in *Advances in Neural Information Processing Systems 4* (J. E. Moody, S. J. Hanson, and R. P. Lippmann, eds.), pp. 950–957, Morgan-Kaufmann, 1992.
- [178] M. Aiserman, E. M. Braverman, and L. Rozonoer, “Theoretical foundations of the potential function method in pattern recognition,” *Avtomat. i Telemekh.*, vol. 25, pp. 917–936, 1964.
- [179] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, “Support vector method for novelty detection,” in *Advances in neural information processing systems*, pp. 582–588, 2000.
- [180] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, “Estimating the support of a high-dimensional distribution,” *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.
- [181] D. M. Tax and R. P. Duin, “Support vector data description,” *Machine learning*, vol. 54, no. 1, pp. 45–66, 2004.
- [182] M. Bekkar, H. K. Djemaa, and T. A. Alitouche, “Evaluation measures for models assessment over imbalanced datasets,” *Journal Of Information Engineering and Applications*, vol. 3, no. 10, 2013.
- [183] N. Chinchor, “Muc-3 evaluation metrics,” in *Proceedings of the 3rd conference on Message understanding*, pp. 17–24, Association for Computational Linguistics, 1991.
- [184] Y.-M. Huang and S.-X. Du, “Weighted support vector machine for classification with uneven training class sizes,” in *2005 International Conference on Machine Learning and Cybernetics*, vol. 7, pp. 4365–4369, IEEE, 2005.
- [185] M. F. Møller, “A scaled conjugate gradient algorithm for fast supervised learning,” *Neural networks*, vol. 6, no. 4, pp. 525–533, 1993.
- [186] O. Ronneberger, P. Fischer, and T. Brox, “U-net : Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.
- [187] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” *arXiv preprint arXiv :1511.07122*, 2015.

- [188] T. Pohlen, A. Hermans, M. Mathias, and B. Leibe, “Full-resolution residual networks for semantic segmentation in street scenes,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4151–4160, 2017.
- [189] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset shift in machine learning*. The MIT Press, 2009.
- [190] M. Long, H. Zhu, J. Wang, and M. I. Jordan, “Unsupervised domain adaptation with residual transfer networks,” in *Advances in Neural Information Processing Systems*, pp. 136–144, 2016.
- [191] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, “Adversarial discriminative domain adaptation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7167–7176, 2017.
- [192] X. Glorot, A. Bordes, and Y. Bengio, “Domain adaptation for large-scale sentiment classification : A deep learning approach,” in *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 513–520, 2011.
- [193] M. Freitag and Y. Al-Onaizan, “Fast domain adaptation for neural machine translation,” *arXiv preprint arXiv :1612.06897*, 2016.
- [194] Y.-B. Kim, K. Stratos, and R. Sarikaya, “Frustratingly easy neural domain adaptation,” in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics : Technical Papers*, pp. 387–396, 2016.
- [195] J. Deng, Z. Zhang, F. Eyben, and B. Schuller, “Autoencoder-based unsupervised domain adaptation for speech emotion recognition,” *IEEE Signal Processing Letters*, vol. 21, no. 9, pp. 1068–1072, 2014.
- [196] S. Sun, B. Zhang, L. Xie, and Y. Zhang, “An unsupervised deep domain adaptation approach for robust speech recognition,” *Neurocomputing*, vol. 257, pp. 79–87, 2017.
- [197] W.-N. Hsu, Y. Zhang, and J. Glass, “Unsupervised domain adaptation for robust speech recognition via variational autoencoder-based data augmentation,” in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 16–23, IEEE, 2017.
- [198] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.
- [199] M. Wang and W. Deng, “Deep visual domain adaptation : A survey,” *Neurocomputing*, vol. 312, pp. 135–153, 2018.
- [200] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.
- [201] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, “Cycada : Cycle-consistent adversarial domain adaptation,” *arXiv preprint arXiv :1711.03213*, 2017.
- [202] W. Zhang, W. Ouyang, W. Li, and D. Xu, “Collaborative and adversarial network for unsupervised domain adaptation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3801–3809, 2018.

- 
- [203] M. Long, Z. Cao, J. Wang, and M. I. Jordan, “Conditional adversarial domain adaptation,” in *Advances in Neural Information Processing Systems*, pp. 1645–1655, 2018.
- [204] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, “Deep reconstruction-classification networks for unsupervised domain adaptation,” in *European Conference on Computer Vision*, pp. 597–613, Springer, 2016.
- [205] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.
- [206] M. Ghifary, W. Bastiaan Kleijn, M. Zhang, and D. Balduzzi, “Domain generalization for object recognition with multi-task autoencoders,” in *Proceedings of the IEEE international conference on computer vision*, pp. 2551–2559, 2015.
- [207] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, “A kernel two-sample test,” *Journal of Machine Learning Research*, vol. 13, no. Mar, pp. 723–773, 2012.
- [208] N. Aronszajn, “Theory of reproducing kernels,” *Transactions of the American mathematical society*, vol. 68, no. 3, pp. 337–404, 1950.
- [209] F. Riesz, *Sur une espèce de géométrie analytique des systèmes de fonctions sommables*. Gauthier-Villars, 1907.
- [210] A. Smola, A. Gretton, L. Song, and B. Schölkopf, “A hilbert space embedding for distributions,” in *International Conference on Algorithmic Learning Theory*, pp. 13–31, Springer, 2007.
- [211] A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, and B. K. Sriperumbudur, “Optimal kernel choice for large-scale two-sample tests,” in *Advances in neural information processing systems*, pp. 1205–1213, 2012.
- [212] G. Csurka, “A comprehensive survey on domain adaptation for visual applications,” in *Domain Adaptation in Computer Vision Applications*, pp. 1–35, Springer, 2017.
- [213] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, pp. 2672–2680, 2014.
- [214] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, “A theory of learning from different domains,” *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.
- [215] D.-H. Lee, “Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks,” in *Workshop on challenges in representation learning, ICML*, vol. 3, p. 2, 2013.
- [216] Z. Li, B. Ko, and H.-J. Choi, “Naive semi-supervised deep learning using pseudo-label,” *Peer-to-Peer Networking and Applications*, vol. 12, no. 5, pp. 1358–1368, 2019.
- [217] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” in *NIPS-W*, 2017.

- [218] G. French, M. Mackiewicz, and M. Fisher, “Self-ensembling for visual domain adaptation,” in *International Conference on Learning Representations*, 2018.
- [219] L. v. d. Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.
- [220] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” 2017.
- [221] F. Chollet, “Xception : Deep learning with depthwise separable convolutions,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [222] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

# Sofia MARINO

## Doctorat : Optimisation et Sûreté des Systèmes

### Année 2020

#### Apprentissage profond pour l'analyse de la qualité des pommes de terre

La pomme de terre est l'un des produits agricoles les plus consommés dans le monde. L'aspect visuel de ce tubercule est d'une grande importance pour la plupart des consommateurs. En effet, la qualité des tubercules peut être couramment affectée par divers défauts qui altèrent leur peau et donc leur aspect. Depuis plusieurs années, des méthodes manuelles ont été appliquées afin d'identifier ces défauts, néanmoins cette tâche manuelle est coûteuse, chronophage et subjective. L'objectif ultime de cette thèse est de développer un système de vision artificielle qui soit capable de fournir des informations sur la qualité des échantillons de pommes de terre de manière automatique. Tout d'abord, nous utilisons un système de prise d'images afin de constituer une base de données large et variée. Ensuite, nous proposons et évaluons trois méthodes de classification et de localisation de divers défauts et maladies. Nous avons combiné des techniques traditionnelles d'apprentissage automatique ainsi que des techniques plus récentes reposant sur l'apprentissage profond afin de maximiser les performances de chacune des méthodes proposées. Enfin, pour assurer le bon fonctionnement du système en cas de changements de la distribution des données (évolution du système d'acquisition...), nous avons proposé une méthode d'adaptation de domaine non supervisée fondée sur l'apprentissage antagoniste. Les résultats expérimentaux obtenus démontrent la pertinence de chacune des méthodes présentées, ainsi que leur applicabilité dans un environnement industriel.

Mots clés : apprentissage profond – vision par ordinateur – classification automatique – qualité des produits – pommes de terre, maladies.

#### Deep learning for potato quality analysis

Potato is one of the most widely consumed agricultural produce in the world. The visual appearance of this tuber is of crucial importance to most consumers who are increasingly demanding. The quality of tubers can be commonly affected by defects or diseases that alter their skin and thus their appearance. For several years now, manual methods have been applied to detect and classify these defects. Nevertheless, this manual task is laborious, subjective and time-consuming. Therefore, the development of methods that automate quality control is of paramount importance to increase efficiency, reduce costs, and obtain objective results that enhance customer confidence. The main objective of this thesis is to develop a computer vision system that can provide information on the quality of potato samples automatically. First, an imaging system is used to build up a large and varied image database. Secondly, three methods for classifying and localizing several defects and diseases are proposed and evaluated. Traditional machine learning techniques as well as more recent techniques based on deep learning are combined to maximize the performance of each new proposed approach. Finally, to ensure that the system continues to perform well even if changes occur in the data distribution, an unsupervised domain adaptation method based on adversarial learning is proposed. The experimental results obtained demonstrate the relevance of each of the proposed methods as well as their applicability in an industrial environment.

Keywords: deep Learning – computer vision – automatic classification – quality of product – potatoes, diseases and pests.

Thèse réalisée en partenariat entre :

