École Normale Supérieure Paris-Saclay

Master 1 GC
UE MÉMOIRE DE STAGE DE MASTER $1^{e}ANNEE$

---

**Effectué à L'Ecole Nationale Supérieure d'Architecture Paris-Malaquais**
**Sous la direction de Robert Le Roy**

# Utilisation d'un réseau de neurones artificiels pour diminuer le taux de faux positifs lors de la détection de défauts sur le bois

---

par
Arthur Calvi

## Remerciements

Je remercie chaleureusement tout celles et ceux qui m'ont épaulé et orienté durant ce stage.
J'adresse tout particulièrement mes remerciements à Robert Le Roy pour m'avoir proposé ce sujet et pour m'avoir guidé tout au long du stage. Thierry Ciblac pour m'avoir conseillé dans mes pistes de réflexions.
Je remercie également Antoine Roux de Charpente-Concept pour son expertise professionnelle sur le sujet.
Enfin, je remercie vivement Alexandra Bourdot qui a consacré du temps à la lecture de mon travail.

# Table des matières

# Table des figures

# Liste des tableaux

**Mots clefs :** Apprentissage automatique, Réseau de neurones, Vision artificielle, Contrôle du bois, Défauts du bois, Qualité du bois, Réemploi du bois.

**Keywords :** Machine learning, Neural network, Artificial vision, Wood control, Wood defects, Wood quality, Wood reuse.

**Résumé :** Le contrôle de la qualité du bois est une tâche qui aujourd'hui peut être réalisée automatiquement de façon précise. Grâce à l'apprentissage automatique, les ordinateurs sont aujourd'hui capables de reconnaître les défauts du bois avec une assez bonne précision pour être exploitée par l'industrie. Dans cette étude, nous nous intéressons aux performances d'un réseau de neurone de type Perceptron pour filtrer les faux positifs d'une base de données. Dans notre cas, un faux positif fait référence à une zone de bois saine détectée comme étant un défaut par un algorithme de reconnaissance. Le but du filtre est de diminuer le taux de faux positifs et ainsi d'améliorer la fiabilité de l'algorithme de reconnaissance. Cette fiabilité est importante car elle détermine la justesse du contrôle de qualité du bois, un taux élevé de faux positifs amène à des erreurs de jugement de qualité et à des pertes matérielles.

**Abstract :** Controlling the quality of wood is a task that today can be done automatically and precisely. Thanks to machine learning, computers today are able to recognize defects in wood with enough precision to be exploited by industry. In this study, we are interested in the performance of a Perceptron-type neural network to filter false positives from a database. In our case, a false positive refers to a healthy area of wood detected as a defect by a recognition algorithm. The purpose of the filter is to decrease the rate of false positives and thus improve the reliability of the recognition algorithm. This reliability is important because it determines the correctness of wood quality control. A high rate of false positives leads to quality errors in judgment and material loss.

# Chapitre 1

# Cadre de l'étude

## 1.1 Introduction

La reconnaissance d'image, dans le contexte de la vision artificielle, est la capacité d'un programme à identifier des objets comme des personnes, des voitures ou encore des cellules anormales. Le champ d'application de cette technologie est vaste : il va de la détection de défauts *(cellules, matériaux, etc)* dans le domaine médical ou du génie civil jusqu'au monde du transport avec la conduite automatique *(reconnaissance piétons, voitures, etc)*. Bien que cette technologie soit très bien développée dans certains domaines elle reste encore discrète dans l'industrie du bois. Pourtant, les applications ne manquent pas. Le classement des parquets, des planches de bois et la caractérisation des éléments structuraux nécessitent l'identification de défauts.

Aujourd'hui, pour caractériser le bois de structure la méthode couramment utilisée est l'essai mécanique qui conduit inéluctablement à des pertes matérielles et donc économiques. Il existe bien des méthodes non intrusives à l'instar du Sylvatest. Cet instrument utilise les ondes acoustiques pour déterminer une estimation de la résistance des matériaux [1]. Cependant, il s'agit de la seule méthode non intrusive utilisée par les charpentiers pour effectuer des diagnostics *(arbres, éléments structuraux in-situ)*. Pour améliorer les diagnostics il serait intéressant de pouvoir utiliser deux méthodes non intrusives et de les confronter/coupler. L'idée qui motive ce stage est le développement d'un logiciel capable, à partir de photos, de déduire la classe de résistance équivalente du bois selon les normes de caractérisation visuelle *(NF B52-001-1 et NF B52-001-2)*.

Le développement d'un tel logiciel nécessite l'élaboration de plusieurs programmes :

1. Un programme de détection de défauts
2. Un programme de classification des défauts
3. Un programme d'application de la norme
4. Un programme de reconstitution d'image.

Dans cette étude, nous nous intéresserons uniquement au programme de classification des défauts. En effet, une fois que les défauts ont été détectés il faut les classer par catégorie *(noeud vif, noeud mort, échauffure, fente, etc)* dans le but d'utiliser la norme pour estimer la classe de résistance du bois. Ce classement se fera par l'intermédiaire d'un réseau de neurones entraîné au préalable sur une banque d'images. Ce travail correspond donc à la première étape de développement du logiciel. Il s'agit de connaître la capacité des réseaux neurones à classifier les défauts du bois. Bien sûr, ce travail pourrait servir pour d'autres applications. La classification des défauts est également utilisée pour le classement des planches de bois ainsi que pour la confection des parquets en bois.

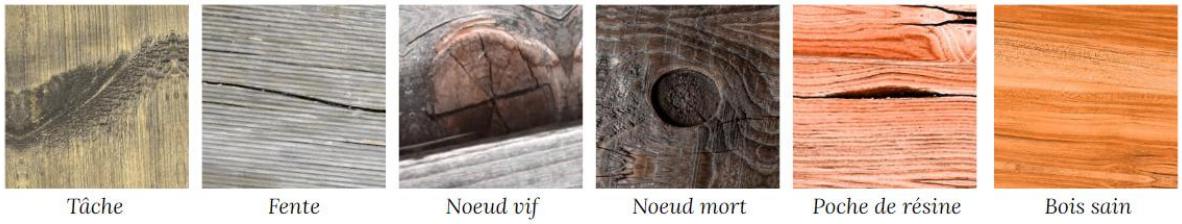



**Figure 1.1** – Catégories de défauts et bois sain

## 1.2 Le laboratoire Géométrie Structure Architecture (GSA) de l'Ecole Nationale Supérieure d'Architecture de Paris Malaquais

Le laboratoire GSA, au sein duquel j'ai fait mon stage, est un lieu de recherche à l'articulation des sciences de l'ingénieur et de l'architecture. Véritable pont entre le génie civil et l'architecture, ce laboratoire se penche sur des projets de morphologie structurale et d'intégration technologique pour le bien de l'environnement. Afin de marier au mieux possible ces domaines et de comprendre leurs interactions, le laboratoire s'intéresse à l'histoire du croisement des sciences, des techniques et de l'art de la construction.

Le laboratoire fédère des chercheurs, enseignants et étudiants dans plusieurs écoles françaises qui sont principalement des écoles d'architectures. Pour faire cohabiter recherche et enseignement, le laboratoire GSA est un adepte de l'expérimentation constructive sur prototypes. Grâce à ces partenaires industriels le laboratoire est en mesure de créer plusieurs prototypes dans le cadre de ces projets.

Pour mon stage très théorique, j'ai eu l'honneur d'être dirigé dans mes travaux par :

- **Thierry Ciblac**, professeur des ENSA – champ STA – ENSA Paris-Malaquais, docteur en génie civil de l'INSA Lyon, ingénieur des Travaux Publics de l'État. Membre du Conseil de laboratoire.
- **Robert Le Roy**, professeur des ENSA – champ STA – ENSA Paris-Malaquais, professeur de l'École des Ponts ParisTech, HDR Université de Marne la Vallée, agrégé de génie civil.

Malheureusement, en raison de la crise sanitaire de la COVID-19, je n'ai pas eu l'honneur de visiter ces laboratoires. Ma mission a été essentiellement réalisée en télé-travail. Robert le Roy et Thierry Ciblac ont suivi l'avancement de mes travaux via des visio-conférences toutes les deux semaines. J'ai été très bien guidé par mes tuteurs et eu beaucoup de liberté dans mes travaux. Cela m'a permis d'effectuer la recherche comme je le souhaitais et d'avancer à mon rythme.

# Chapitre 2

# Approche

## 2.1 État de l'art

Dans cet état de l'art nous allons parler des algorithmes permettant la classification des défauts mais également des différentes techniques de détections des défauts. D'une part car la détection peut causer des faux positifs [1], un élément important à prendre en compte pour le programme de classification. D'autre part, car la détection sera un programme à utiliser dans le cadre du développement du logiciel qui motive cette étude.

Il existe plusieurs méthodes pour détecter les défauts et réaliser la segmentation de l'image pour les isoler. Une des méthodes les plus utilisées est le seuillage. Cette méthode fut introduite par Conners et al. (1983, 1984) pour détecter les défauts du bois [2]. Ils ont utilisé un algorithme de seuillage pour repérer les surfaces de défauts sur des images en niveaux de gris de poutre de résineux. Une autre méthode est la détection de contour qui fut utilisée sur le bois par Lepage et al. (1992) sur des images en niveaux de gris également [3]. Ils ont utilisé plusieurs masques pour détecter des contours sur plusieurs échelles. Un classificateur *(basé sur un réseau de neurones)* se servait ensuite des informations des masques pour détecter les défauts. Forrer et al. (1988, 1989) ont développé un algorithme de traitement d'image morphologique [4]. Cet algorithme de segmentation utilise la méthode de dilatation et érosion sur des images en niveaux de gris décrite par Werman et Peleg (1985) [5]. La précision de détection de cet algorithme *(MISM)* est comparable avec les méthodes de seuillage mais produit plus de faux positifs. Après avoir développé des algorithmes sur des images en niveaux de gris, les chercheurs se sont intéressés aux images en couleur. En fait, ils ont découvert que les défauts subtils comme les tâches d'origine fongique peuvent être détectées avec une meilleure précision en utilisant les couleurs des images. Souvent avec des images en niveaux de gris, les tâches de croissance du bois sont confondues avec des échauffures, des bleuissements ou autres par les algorithmes ; ce qui apporte un pourcentage de faux positif élevé. Butler et al. (1989) ont montré qu'utiliser les informations des canaux RGB augmentait la précision de détection des défauts subtiles par 20% [6].

Après les années 2000 commencent à arriver d'autres types d'algorithmes reposant sur les réseaux de neurones. Gonzalo A.Ruz et al. (2005) présente l'algorithme : *NeuroFuzzy Color Image Segmentation (FMMIS - Fuzzy Min Max Image Segmentation)* [7]. Cet algorithme utilise une méthode de seuillage adaptative pour localiser les pixels initiaux d'un défaut puis utilise un réseau de neurone et la théorie de la *logique floue* [2] pour définir la zone du défaut [8, 9]. Les résultats sont convaincants avec un taux de faux positifs peu élevé : 6% en moyenne.

1. Un faux positif est la détection d'une zone comme comportant un défaut alors qu'elle n'en contient pas.
2. Fuzzy Logic en anglais - c'est une logique polyvalente où les valeurs de vérité des variables - au lieu d'être vrai ou faux - sont des réels entre 0 et 1. En ce sens, elle étend la logique booléenne classique avec des valeurs de vérités partielles1. Elle consiste à tenir compte de divers facteurs numériques pour aboutir à une décision qu'on souhaite acceptable.

En terme d'évolution, les algorithmes de détection se sont améliorés au fur et à mesures des années grâce à l'utilisation de nouvelles méthodes de résolution comme les réseaux de neurones. Le taux de faux positifs pour les algorithmes d'aujourd'hui est meilleur qu'auparavant mais reste élevé. Rappelons qu'un taux de faux positif bas est très important dans le cadre de l'industrie du bois, cela permet d'éviter des pertes inutiles et donc de maintenir la profitabilité économique dans plusieurs activités comme le classement des planches ou la scierie.

Les auteurs de l'algorithme *FMMIS* précisent que le taux de faux positifs pourrait être diminué grâce à l'utilisation d'un filtre. Ce filtre pourrait simplement être un classificateur capable de dissocier les zones présentant des défauts aux zones saines. La méthode la plus commune pour classifier est l'utilisation d'un Perceptron multicouche[3] (Haykin 1994) [10]. Pour classifier les défauts du bois, Pham et Alcock (1999) ont comparé l'utilité des différentes caractéristiques *(circularité de l'objet, statistiques de la fenêtre de segmentation, etc)* à donner au réseau de neurones [11]. Les meilleurs résultats furent obtenus avec l'utilisation de tous les éléments, atteignant 85.2 % de précision pour 9 classes de défauts. Pour sélectionner les caractéristiques à fournir au réseau de neurones, P. Estévez et al. (2003) ont utilisé un algorithme génétique. Leur réseau de neurones obtint une précision de 80% pour 11 classes [12].

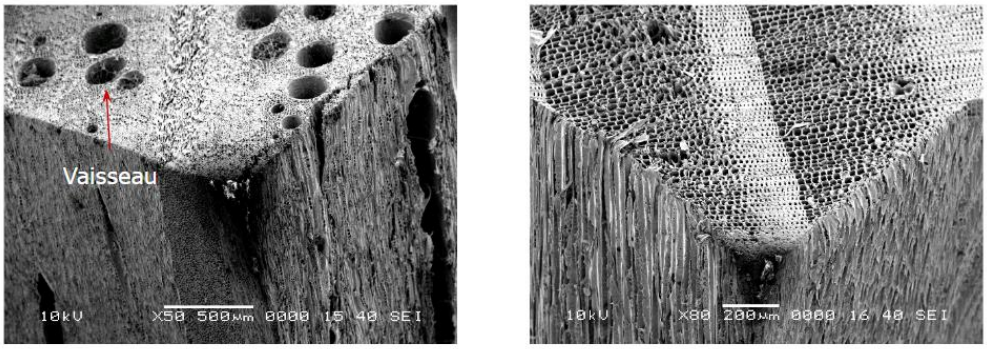
Dans d'autres domaines, les chercheurs ont utilisé l'association de plusieurs classificateurs pour obtenir de meilleurs résultats. C'est le cas de K. Suzuki et al. (2003) qui ont utilisé plusieurs réseaux de neurones, chacun entraîné pour distinguer des nodules sains de nodules anormaux d'un certain type [13]. Cette idée de synergie entre plusieurs réseaux de neurones a d'ailleurs été utilisée par Pham et Alcock [14].

Ainsi, la classification des défauts de bois est un problème sur lequel les chercheurs se penchent depuis une trentaine d'années. Bien que les résultats obtenus soient bons, les dernières avancées *(architecture, préparation des données, etc)* dans le monde des réseaux de neurones artificiels pourraient permettre des améliorations significatives des classificateurs de défauts de bois.

## 2.2 Pré-requis

Il existe deux types de bois : les résineux et les feuillus. Les principales différences sont botaniques et sont associées à la manière dont les arbres grandissent et sont abattus. Pour les résineux, les cellules sont proches et ne peuvent fonctionner comme des conduits. En revanche, pour les feuillus les cellules peuvent fonctionner comme des conduits à nutriments. Pour les résineux ce sont les vaisseaux qui se chargent de conduire les nutriments. Les feuillus sont plus enclins à recevoir des traitements chimiques grâce à la structure ouverte des cellules. Les résineux sont utilisés pour des constructions à long terme comme les bureaux, les planchers et les éléments structuraux. Les feuillus ont un domaine d'application plus large comprenant les équipements d'habitation *(fenêtres, portes)*, les meubles, le papier et les MDF *(Medium Density Fiberboard)*.

3. Multilayer Perceptron en anglais - Le Perceptron multicouche est un Classifieur linéaire de type réseau neuronal formel organisé en plusieurs couches au sein desquelles une information circule de la couche d'entrée vers la couche de sortie uniquement.




Résineux Feuillu

**Figure 2.1** – Micro-structures des résineux et feuillus

Les aspects des défauts varient des résineux aux feuillus mais également d'une espèce à une autre. C'est pourquoi, selon l'essence de bois exploitée l'algorithme devra être entraîné avec des images de cette même essence pour avoir les meilleurs résultats de prédiction possible.

## 2.3 Démarche

Le *Machine Learning* nécessite un nombre de données important pour être performant. Dans notre cas, la base de données sera limitée à une petite taille : un peu plus de 500 images pour 6 classes : *noeud vif, noeud mort, fente, échauffure/tâche, poche de résine et bois sain.* En fait, il y a trop peu d'images par classe pour espérer développer un classificateur de défauts efficace. Malheureusement, les défauts du bois sont très similaires et parfois il s'avère être dur pour un humain de les différencier. Par exemple les noeuds vifs et morts sont durs à distinguer, même un oeil d'expert pourrait s'y tromper. En *Machine Learning* plus les classes se ressemblent et sont dures à distinguer plus l'algorithme nécessite d'images différentes pour être performant. Généralement, les bases de données sont composées d'au moins 1000 images par classe. Les bases d'images récemment crées comportent elles parfois 1M d'images par classe. Face à ce problème de taille de base de données, nous avons opté pour la stratégie suivante :

- Le développement d'un classificateur binaire *(qui sera appelée **CNN-1**)*. Le but de ce réseau de neurones est de distinguer les zones de bois saines des zones de défauts. En d'autres termes, cet algorithme doit être en mesure de réduire le nombre de faux positifs créés par la détection de défaut.
- Le développement d'un classificateur à 6 classes [4] *(**CNN-2**)* selon les résultats du premier réseau de neurones.
- Le développement d'un algorithme d'expansion automatisée de la base de données selon les résultats des deux réseaux de neurones.

Les deux classificateurs seront pré-entraînés sur cette petite base de données. Selon les résultats, les classificateurs pourraient ensuite être utilisés pour agrandir la base de données. Par exemple, un des réseaux de neurones pourrait être utilisé pour classer un lot d'images en différentes catégories avec une certaine précision. Puis un expert pourrait valider et corriger cette classification. Par la suite, ce même classificateur pourrait être ré-entraîné sur cette même base de données agrandie. Cet apprentissage devrait alors aboutir à une meilleure précision de prédiction. L'étape pourrait être ainsi répétée plusieurs fois afin d'augmenter la taille de la base de données et également améliorer la précision du réseau de neurones. Ce genre d'approche est connu et a par exemple permis de créer des bases de données comme *'LSUN'*. En effet, Fisher Yu et al. ont utilisé une approche semi-automatique pour collecter une base de données d'1M d'images par classe [15]. Ils ont utilisé un système de propagation d'étiquetage des images qui amplifie automatiquement les efforts de l'étiquetage manuel humain. Leur précision d'étiquetage fut supérieure en

4. 5 catégories de défaut et le bois sain.

moyenne à 90%. C'est légèrement moins précis que pour un étiquetage humain mais cela reste assez juste pour entraîner des classificateurs de bonne précision.

# Chapitre 3

# La base de données

De précédentes études comme ImageNet [16] ont utilisé Amazon Mechanical Turk [1] pour créer leur base d'images. À travers cette plateforme, des humains peuvent étiqueter des images en ligne afin de constituer les différentes classes de la base d'images. Ce concept très pratique est néanmoins long puisqu'il a fallu environ un an pour collecter 1000 images pour des classes simples : plantes, animaux, etc. Dans notre cas, seuls des humains ayant des connaissances sur les défauts des bois seraient capables de classifier les images de notre base de données. Ainsi, Amazon Mechanical Turk n'apparait pas comme une option viable pour constituer notre base de données. C'est pourquoi, nous avons décidé de réaliser manuellement la création d'une petite base d'images pour pré-entrainer nos modèles.

## 3.1 Acquisition

Pour créer la base d'images, des images de texture de bois ont été collectées via Google image et d'autre banques. Ensuite, de ces images ont été extraites manuellement des zones de défauts puisque nous n'avions pas accès à un programme de détection [2]. Les zones extraites sont manuellement choisie de forme carrée pour créer des images utilisables par le réseau de neurones (nous reviendrons plus tard sur ce détail). Finalement les zones extraites englobent totalement le défaut. En plus de zones de défauts, des zones 'saines' ont été recueillies pour permettre au réseau de neurones de distinguer les zones saines des zones de défauts et ainsi limiter le taux de faux positif. Pendant l'acquisition des images, elles ont été classées dans 6 catégories : *noeud vif, noeud mort, poche de résine, fentes, échauffure/tâche et bois sain.*

1. C'est une plateforme web de crowdsourcing qui vise à faire effectuer par des humains, contre rémunération, des tâches plus ou moins complexes.

2. Les algorithmes de détection ont tendance à créer des zones de défaut rectangulaires notamment pour les fentes mais il est possible de modifier leur code pour extraire des zones parfaitement carrée.

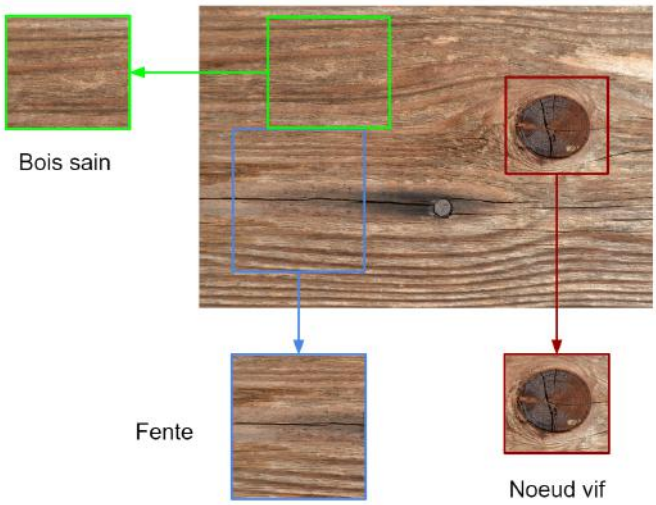



**FIGURE 3.1** – Processus d'acquisition.

Au total, environ 200 images ont été collectées. À partir de ces images, plus de 500 zones ont été classées dans les 6 catégories. En moyenne, une zone de défaut et une zone de bois sain ont été extraites d'une image. Toutes les zones ont été enregistrées avec les lignes de fibre horizontales dans le but d'imiter un processus industriel. Toutes les images ont également été enregistrées avec une résolution fixe de $256 \times 256$ pixels puisque le réseau de neurones nécessite une entrée de dimension fixe. Parce que les zones ont été recadrées de manière carrée manuellement, un algorithme de recadrage a été utilisé pour obtenir des formes parfaitement carrées. L'algorithme change d'abord la résolution de l'image tel que le plus petit côté mesure 256 pixels puis recadre l'image de façon centrée pour créer une forme parfaitement carrée.

## 3.2 Augmentation

Le *Deep learning* nécessite des bases de données de très grandes tailles. Étant limité à une petite base de données, la technique d'*augmentation* a été utilisée. L'*augmentation artificielle*[3] permet d'améliorer l'apprentissage et d'éviter le phénomène de *surapprentissage*[4]. L'apprentissage se fait en donnant plusieurs fois les images de la base de données au réseau de neurones pour qu'il détermine des motifs pour reconnaître ce qu'est un noeud mort ou du bois sain par exemple. L'idée de l'augmentation est de donner une version différente de l'image à chaque fois. L'algorithme d'augmentation à partir de l'image originelle d'un défaut de la base d'apprentissage créé une image *augmentée* différente à chaque époque[5] grâce à des transformations, tout en garantissant la préservation de la classe de l'image originelle [16]. Ce processus permet de diversifier l'apprentissage : le réseau de neurones ne verra jamais plusieurs fois la même image (mais verra le même

3. Processus consistant à modifier une image via des transformations afin de créer des nouvelles données pour le réseau de neurones.

4. Overfitting en anglais - En statistique, le surapprentissage, ou sur-ajustement, ou encore surinterprétation, est une analyse statistique qui correspond trop étroitement ou exactement à un ensemble particulier de données. Ainsi, cette analyse peut ne pas correspondre à des données supplémentaires ou ne pas prévoir de manière fiable les observations futures. Un modèle surajusté est un modèle statistique qui contient plus de paramètres que ne peuvent le justifier les données.

Le problème existe aussi en apprentissage automatique. Il est en général provoqué par un mauvais dimensionnement de la structure utilisée pour classifier ou faire une régression. De par sa trop grande capacité à capturer des informations, une structure dans une situation de surapprentissage aura de la peine à généraliser les caractéristiques des données. Elle se comporte alors comme une table contenant tous les échantillons utilisés lors de l'apprentissage et perd ses pouvoirs de prédiction sur de nouveaux échantillons.

5. Durant l'apprentissage, le réseau a vu une époque lorsque qu'il a vu toutes les images de la base d'apprentissage une fois.

défaut plusieurs fois de façons différentes) même si il est entraîné sur plusieurs époques. Ce processus peut être assimilé à notre façon d'apprendre : pour reconnaître un type d'objet l'homme va inspecter celui-ci : le regarder selon plusieurs angles, selon plusieurs lumières pour se faire une idée de la représentation de cet objet avec des caractéristiques différentes de vision *(éclairage, angle de vue, etc)* et ainsi pouvoir reconnaître ce type d'objet dans d'autres situations.

Inspiré par les travaux de Ren Wu et al. [17], les transformations suivantes ont été utilisées en prenant en compte le contexte de notre étude et en utilisant l'API Python *Keras*[6].

— Rotation : une petite variation entre $0^o$ et $10^o$ a été choisie en présumant que l'erreur de placement ou de niveau horizontal d'un humain prenant une photographie est faible.
— Luminosité : la luminosité est multipliée par un facteur aléatoire compris entre 0.6 et 1.4 permettant de simuler différents éclairages. Cela correspond à une variation allant d'une sous-exposition jusqu'à une surexposition légère.
— Retournement vertical et horizontal puisque les défauts n'ont pas d'endroit ni d'envers.
— Recadrage aléatoire : extraction de zones de $224 \times 224$ pixels depuis des images de $256 \times 256$ pixels. Processus utilisé par ImageNet , cela permet d'augmenter la fiabilité du réseau de neurones [16]. Le recadrage permet de ne plus voir le défaut dans sa globalité dans certains cas. Pourvoir reconnaître un défaut sans avoir à le voir en entier améliore la robustesse de la détection.
— Altération de la teinte avec un angle compris entre $15^o$ et $45^o$. L'angle a été choisi pour couvrir les couleurs naturelles du bois : du marron jusqu'au vert.
— Zoom aléatoire entre 0 et 30% avec non-conservation de l'aspect. La non conservation de l'aspect permet d'apprendre au réseau que les noeuds peuvent avoir des formes ovales plutôt que parfaitement rondes par exemple.

Le décalage ne fut pas utilisé puisque nous avons estimé qu'il peut créer des artefacts ressemblant à des défauts.

Le tableau suivant résume les possibilités de transformations, au final l'algorithme peut créer plus de 184 000 000 versions différentes d'une image.

| Transformation | Méthode | Possibilités |
|---|---|---|
| Rotation | Keras | 10 |
| Luminosité | Keras | >80 |
| Retournement horizontal | Keras | 2 |
| Retournement vertical | Keras | 2 |
| Recadrage aléatoire | Personnelle | >64 |
| Changement de teinte | Personnelle | 30 |
| Zoom aléatoire | Keras | >30 |
| Total | | > 184 000 000 |

**Table 3.1** – Les transformations opérées pour l'augmentation

6. La bibliothèque Keras permet d'interagir avec les algorithmes de réseaux de neurones profonds et de machine learning, notamment Tensorflow3, Theano, Microsoft Cognitive Toolkit4 ou PlaidML.

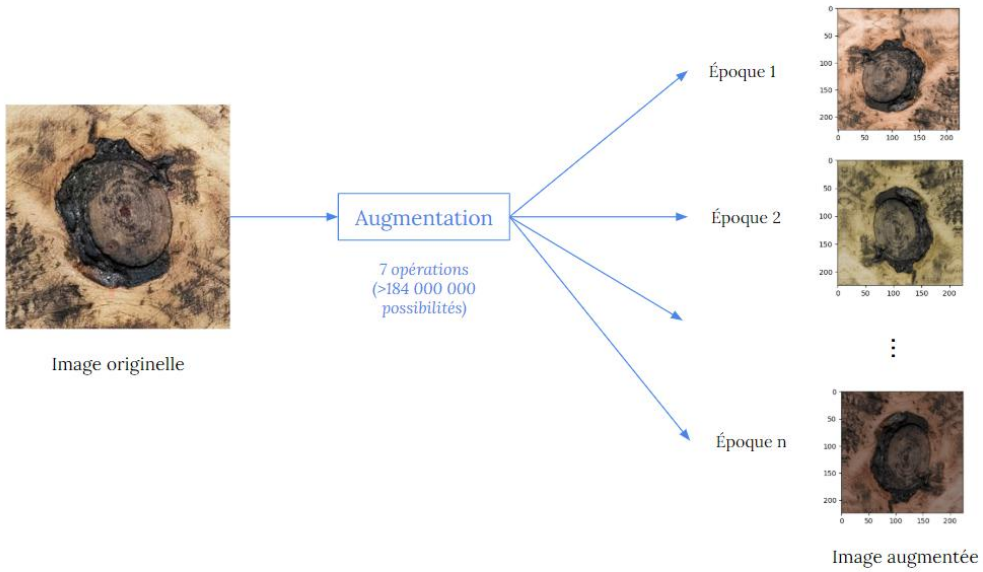



**Figure 3.2** – Processus d'augmentation au fil des époques

## 3.3 Préparation des données

D'après les dernières recherches sur les réseaux de neurones, une des meilleures méthodes pour normaliser les images est de soustraire la valeur moyenne des canaux RGB à chaque pixel pour centrer les données puis de diviser ces 3 nouvelles valeurs R.G.B centrées par l'écart type de chaque canal sur l'image [16], [18], [19], [20]. La distribution de ces données ressemble alors à une Gaussienne centrée en zéro. Cette normalisation permet d'assurer que les paramètres d'entrées *(les pixels)* ont une distribution similaire pour chaque image. Ce processus permet d'améliorer la convergence du réseau de neurones durant l'entraînement.

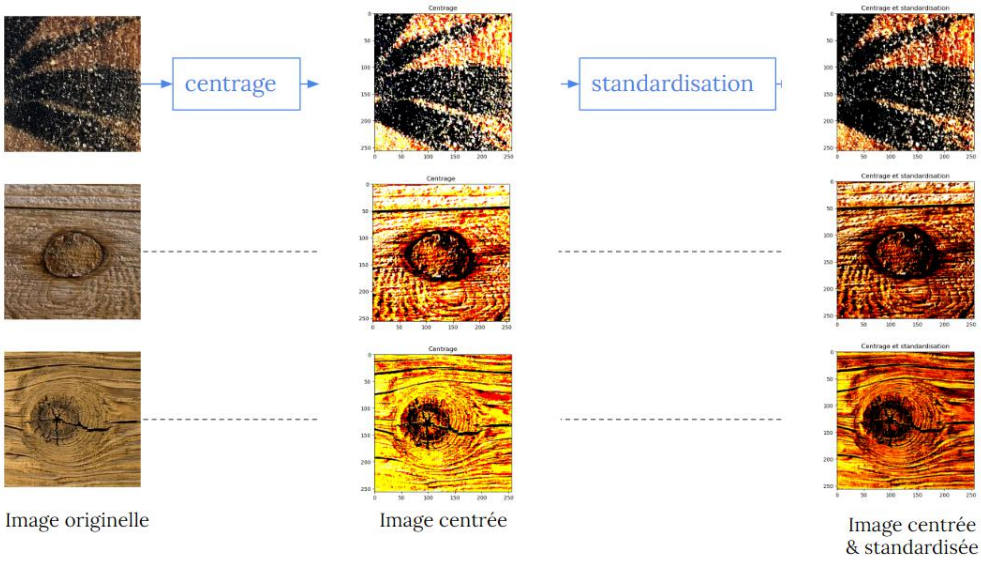



**Figure 3.3** – Processus de normalisation

## 3.4 La base de données en résumé

Finalement les caractéristiques des bases de données selon les algorithmes **CNN-1** et **CNN-2** sont résumés dans les tableaux suivants :

| Base | Images par classe (2) | Augmentation |
|---|---|---|
| Entraînement | 160 | Oui |
| Validation | 40 | Non |
| Test | 50 | Non |

**Table 3.2** – Base de données de CNN1

| Base | Images par classes (6) | Augmentation |
|---|---|---|
| Entraînement | 32 | Oui |
| Validation | 8 | Non |
| Test | 10 | Non |

**Table 3.3** – Base de données de CNN2

Les règles utilisées pour divisier la base de données en 3 parties sont les suivantes :

- Base de 400 images. Cette partie représente 80% de la base totale. Cette partie est subdivisée en 2 autres parties :
  - Base d'entraînement : 80% des 400 images. Cette base est utilisée pour déterminer les poids des connexions entre neurones.
  - Base de validation . 20% des 400 images. Cette base est utilisée pour régler les hyperparamètres [7] du modèle.
- Base de test comprenant plus de 100 images. Cela représente environ 20% de la base de données. Cette base fournit une évaluation impartiale du modèle finale entraîné sur la base des 400 images. Dans notre cas la base de test n'est jamais utilisée pour l'apprentissage.

7. En apprentissage automatique, un hyperparamètre est un paramètre dont la valeur est utilisée pour contrôler le processus d'apprentissage

# Chapitre 4

# Le réseau de neurones

## 4.1 Introduction aux réseaux de neurones convolutifs

Les réseaux de neurones sont des outils mathématiques pour résoudre des problèmes d'optimisation. Leur nom a été donné suite à la ressemblance structurelle entre les neurones biologiques et artificiels. Cependant leurs fonctionnements diffèrent et certains préfèrent appeler un neurone artificiel : une unité de calcul. Un réseau de neurone artificiel est composé de couches de neurones qui s'assemblent pour former une certaine structure. Dans notre étude nous nous limiterons aux réseaux de neurones convolutifs [1] séquentiels. Les réseaux convolutifs tirent leur nom de l'opération mathématique opérée par certaines couches de neurones : la convolution [2]. Cette opération est pertinente pour l'analyse des signaux et des images. Le terme séquentiel signifie que les couches de neurones sont disposées avec une structure dite en série.

### 4.1.1 Le neurone artificiel ou unité de calcul

Un neurone est l'unité de base de calcul d'un réseau de neurones. Il prend comme entrée un vecteur $\mathbf{x}$ et produit un vecteur $\mathbf{y}$ tel que $\mathbf{y} = f(\mathbf{x})$, f étant la fonction de combinaison réalisée par le neurone. Ce vecteur $\mathbf{y}$ est par la suite donné à une fonction d'activation $\varphi$ qui déterminera la sortie du neurone. Les neurones sont nommés par rapport aux noms des fonctions de combinaison *(convolution, pooling, etc)* et d'activation *(ReLU, Swish, Softmax, ...)*.

1. En apprentissage automatique, un réseau de neurones convolutifs ou réseau de neurones à convolution (en anglais CNN ou ConvNet pour Convolutional Neural Networks) est un type de réseau de neurones artificiels acycliques (feed-forward), dans lequel le motif de connexion entre les neurones est inspiré par le cortex visuel des animaux. Les neurones de cette région du cerveau sont arrangés de sorte qu'ils correspondent à des régions qui se chevauchent lors du pavage du champ visuel. Leur fonctionnement est inspiré par les processus biologiques, ils consistent en un empilage multicouche de perceptrons, dont le but est de pré-traiter de petites quantités d'informations.

2. La convolution, ou produit de convolution, dans le domaine du traitement d'image est une généralisation du filtre moyenneur.

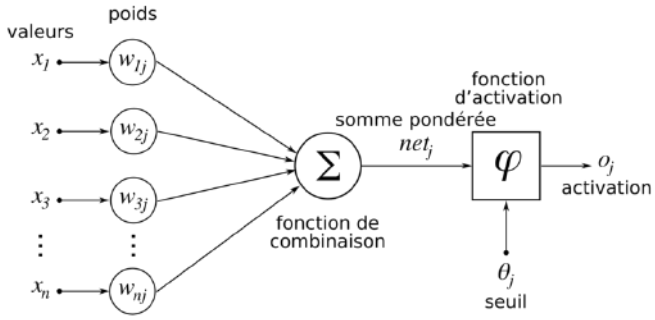



**Figure 4.1** – Schéma du neurone artificiel

Dans le fonctionnement d'un réseau de neurones, l'entrée $\mathbf{x}$ est en fait pondérée par des poids. En effet, dans le vecteur $\mathbf{x} = (x_1, ..., x_n)$, les valeurs $x_i$ représentent les sorties des neurones précédents. Ces valeurs sont pondérées par les poids $W_{ij}$ [3]. Ainsi le neurone reçoit alors comme entrée le vecteur $\mathbf{W.x} = (W_{1j}.x_1, ..., W_{nj}.x_n)$.

Le neurone est ensuite activé par l'intermédiaire d'une fonction d'activation. Historiquement, la fonction *sigmoïde* fut très utilisée comme fonction d'activation. Cependant, cette fonction possède deux principaux inconvénients durant l'apprentissage :

— A cause de la saturation de cette fonction aux valeurs extrêmes, la valeur du gradient local (dérivée de cette fonction sigmoïde) est quasiment nulle sur ces zones extrêmes. Durant la *rétro-propagation du gradient* [4] l'information est alors "tuée" par ce gradient quasi-nul.
— Cette fonction n'est pas centrée en zéro, ce qui peut amener une mauvaise dynamique lors de la mise à jour du gradient pour la détermination des poids.

Le problème de centrage en zéro fut fixé en utilisant la fonction *tanh*. Mais cette fonction présentait toujours le même problème de saturation et de gradient nul. L'introduction de la fonction linéaire rectifiée *(ReLU)* régla ce problème. La fonction ReLU est définie ainsi : $f(x) = max(0, x)$, ce qui fait que l'activation est tout simplement nivelé à 0. Voici ces avantages et inconvénients :

— Accélération de la convergence de la descente de gradient stochastique en comparaison à la fonction sigmoïde.
— Implication d'opérations mathématiques moins coûteuse que pour les fonctions *sigmoïde* et *tanh*.
— Les neurones ReLU peuvent être fragiles durant l'apprentissage et n'être jamais activé si le taux d'apprentissage est trop élevé.

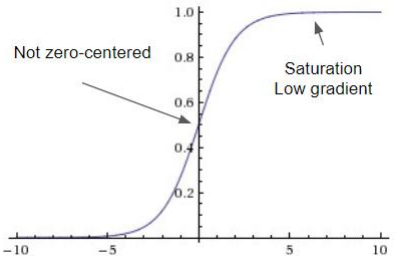



**Figure 4.2** – *Sigmoïde*

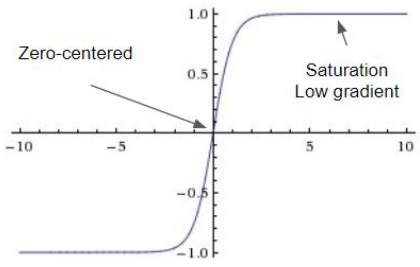



**Figure 4.3** – *Tanh*

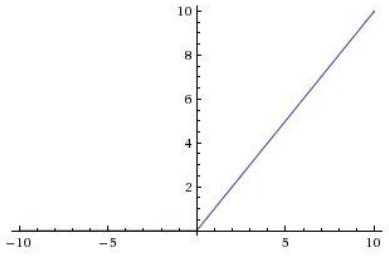



**Figure 4.4** – *ReLU*

3. Le poids $W_{ij}$ est le poids de pondération pour la connexion entre le neurone i et le neurone j.
4. En statistiques, la rétro-propagation du gradient est une méthode pour calculer le gradient de l'erreur pour chaque neurone d'un réseau de neurones, de la dernière couche vers la première.

### 4.1.2 Les couches de neurones

Une couche de neurones est constituée de plusieurs neurones. Il existe différents types de couche de neurones. Dans la reconnaissance d'image, les couches utilisées sont :

— **La couche convolutive**. La convolution est une opération mathématique utilisée en traitement du signal et servant à trouver des motifs. Les paramètres les plus importants sont la taille du filtre convolutif et le déplacement de celui-ci. Pour des images à 3 niveaux de couleurs (RGB) la taille du filtre est souvent $3 \times 3 \times 3$. Le filtre agit alors sur une fenêtre de pixels de taille $3 \times 3$ et sur les 3 canaux RGB. Le déplacement du filtre est généralement fixé à un pixel.
— **Couche de regroupement**. La couche de regroupement est utilisée immédiatement après la couche de convolution pour réduire la hauteur et la largeur de la couche précédente. L'opération de regroupement est réalisé en retenant la valeur maximale de pixel d'une fenêtre de taille $2 \times 2$.
— **Couche entièrement connectée**. Chaque neurone est connecté à tous les neurones de la couche précédente.

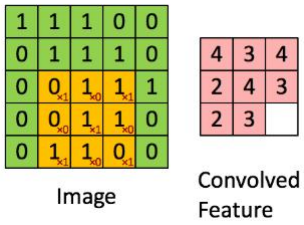



**Figure 4.5** – Principe de la convolution

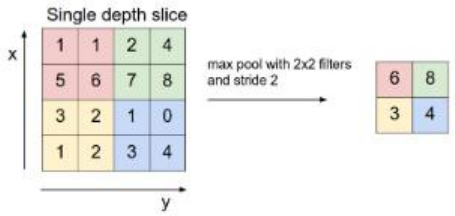



**Figure 4.6** – Principe du regroupement

Concernant le nombre de neurones présents, choisir un grand nombre de neurones permet au réseau de modéliser des fonctions plus complexes mais peut également entraîner du surapprentissage. Il arrive qu'un réseau de neurones très performant apprenne le bruit des données au lieu des relations sous-jacentes. Face à ce problème, il est intéressant d'utiliser l'augmentation artificielle. Puisque cette augmentation réalise des transformations préservant les étiquettes des images cela aide jusqu'à un certain point le réseau de neurones à apprendre ces relations sous-jacentes et non le bruit. On pourrait penser que pour des bases de données réduites il est plus aisé de réduire le nombre de neurones, et donc de paramètres, afin d'éviter le surapprentissage. En pratique il est préférable de contrôler le surapprentissage grâce à la régularisation, l'augmentation artificielle et d'autres techniques plutôt que de diminuer la complexité de notre réseau. Diminuer le nombre de neurones c'est risquer d'utiliser un réseau non capable d'apprendre certaines relations plus complexes. Pour aller plus loin, en termes d'optimisation du modèle : les petits réseaux de neurones produisent des minimums locaux pour lesquels il est très facile de converger : cela peut amener à choisir un minimum local mauvais par rapport aux autres et présentant une perte d'information élevée. À l'inverse les réseaux de neurones plus grands produisent plus de minimum locaux mais ces minimums sont meilleurs en termes de perte [21].

### 4.1.3 La Structure du réseau

Les réseaux de neurones sont composés de plusieurs couches. Les réseaux de neurones *profonds* (composé de plus de 1 couche cachée[5]) ont tendances à être plus performants.

5. une couche cachée est une couche positionnée entre les couches d'entrée et de sortie.

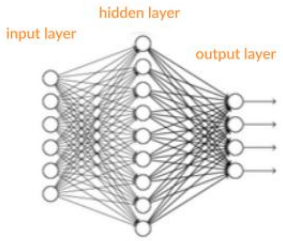



**Figure 4.7** – Réseau de neurones

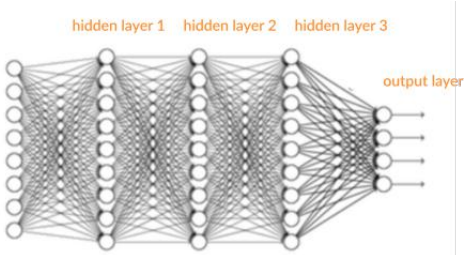



**Figure 4.8** – Réseau de neurones profond

Les réseaux de neurones sont généralement composé de 2 ou 3 couches, ajouter d'autres couches n'aide généralement pas. En revanche, concernant les réseaux de neurones convolutifs c'est différent. Puisque les images contiennent des structures hiérarchiques (un visage est composé de yeux, eux même composé de contours, etc) ajouter des couches prend du sens. En fait, il a été prouvé que la profondeur du réseau est extrêmement importante pour réaliser un système de reconnaissance d'image performant [22].

### 4.1.4 L'apprentissage

Une fois l'architecture du réseau déterminée, le modèle est entraîné. Le but de cette tâche est d'établir les paramètres (les poids des connexions) des neurones dans le but :

- dans le cas de **CNN1** distinguer les zones de défauts des zones de bois sains.
- dans le cas de **CNN2** classifier les images dans 6 classes.

La mise à jour des paramètres est réalisée par une descente de gradient en minimisant une fonction de perte. La fonction de perte est définie telle que la perte est réduite quand la précision du réseau augmente. Pendant l'apprentissage, pas toutes les données sont ingurgitées par le réseau en une seule fois. La base d'apprentissage est divisée en plusieurs lots, généralement de 16 à 32 images. L'apprentissage se déroule en plusieurs itérations jusqu'à ce que le réseau voit toutes les images une fois. Lorsque le réseau a vu toutes les images, on appelle cela une époque. L'apprentissage peut être fait sur plusieurs époques, généralement entre 50 et 200.

## 4.2 Les Architectures

Créer l'architecture d'un réseau de neurones nécessite beaucoup d'expériences. Durant ce stage, le réseau de neurones utilisé est basé sur une architecture similaire à celle proposée par Yan Lecun en choisissant la fonction ReLU comme fonction d'activation [23].

### 4.2.1 CNN1 - Le classificateur binaire

L'architecture est composée de 3 blocs *CONV-POOL*, chacun constitué par une couche de convolution et une couche de regroupement puis à cela s'ajoute deux couches entièrement connectées. La dernière couche n'est composée que d'un seul neurone dont l'activation est la fonction sigmoïde, ce qui est pertinent dans le cas d'un classificateur binaire.

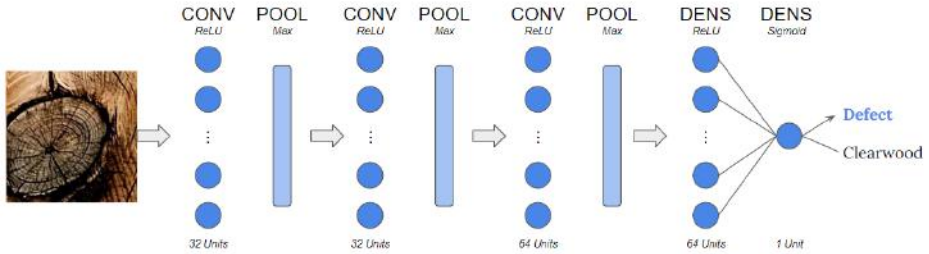



**Figure 4.9** – Architecture

Comme fonction de perte, l'entropie croisée a été choisie car elle convient à des problèmes de reconnaissance avec peu de classes, en l'occurrence il y a ici deux classes : zone de défaut et zone saine. Pour éviter le problème de décalage covariable interne [6] découvert par Ioffe et Szegedy en 2015 [20], nous utiliserons la normalisation par lots [7] fournie par l'API python Keras.

### Influence de l'augmentation

Pour discuter de l'efficacité de l'augmentation artificielle expliquée dans la section 3.2 un réseau de neurones [8] a été entraîné avec et sans augmentation sur assez d'époques pour voir quand apparaît le surapprentissage :

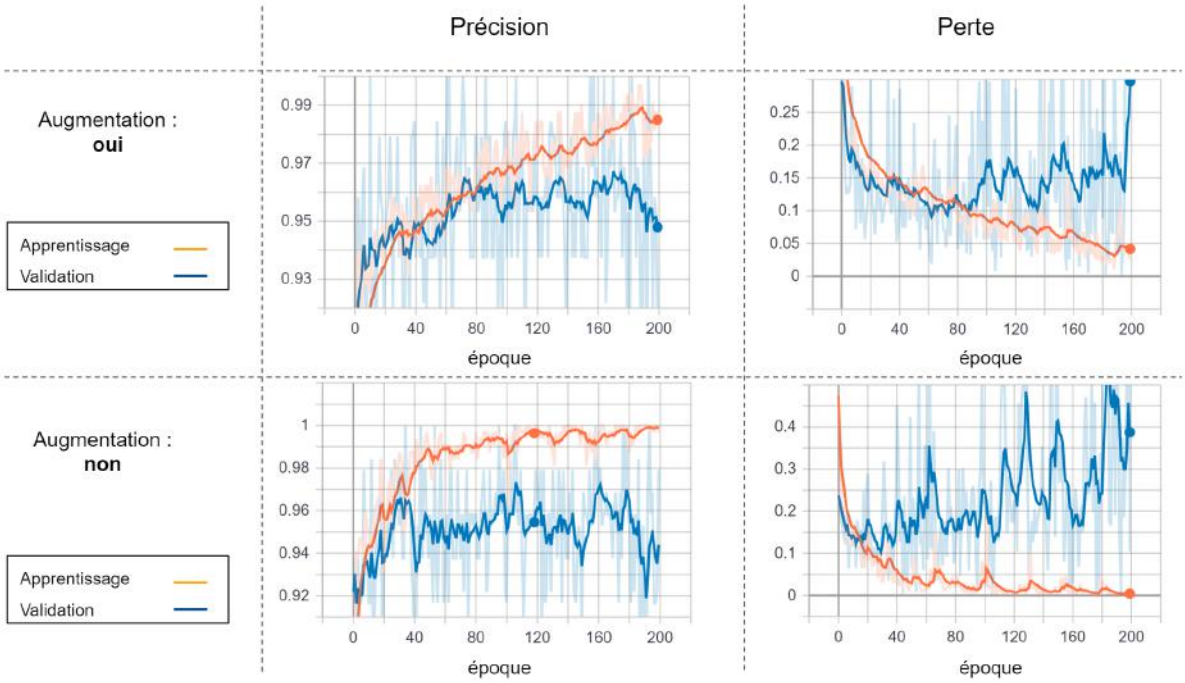



**Figure 4.10** – Comparaison de l'apprentissage avec et sans augmentation

6. Internal Covariate Shift en anglais : Changement dans la distribution des activations du réseau en raison du changement des paramètres du réseau pendant l'apprentissage.

7. Batch Normalization en anglais : La normalisation par lots (également connue sous le nom de norme par lots) est une technique pour améliorer la vitesse, les performances et la stabilité des réseaux de neurones artificiels. La normalisation des lots est utilisé pour normaliser la couche d'entrée en recentrant et en redimensionnant

8. Le réseau de neurones utilisé ici est l'architecture de base de Yan LeCun présentée dans la figure 4.12.

Sur la figure 4.10 sont présentées la précision ainsi que la perte [9] du réseau durant l'apprentissage et la validation. En abscisse se trouve le nombre d'époques. Il est intéressant de remarquer que l'augmentation permet d'entraîner le réseau sur plus d'époques avant de voir l'apparition du surapprentissage. En effet, quand il y a surapprentissage la précision de l'apprentissage augmente alors que la précision de la validation stagne, de même la perte de la validation augmente tandis que la perte de l'apprentissage continue à diminuer. Après cette limite dépassée : le réseau apprend des relations intrinsèques à la base de données d'apprentissage qui ne peuvent être généralisées à de nouvelles bases de données. Avec augmentation, la précision de l'apprentissage continue à augmenter après la $80^{me}$ époque alors que la précision de la validation ne progresse plus : il y a surapprentissage. Ces courbes sont la preuve que l'augmentation permet d'enrichir la base de données et de contrôler le surapprentissage. Sans augmentation, le phénomène arrive plus tôt : environ à la $40^{me}$ époque. Avec augmentation, le réseau peut être entraîné sur plus d'époques avant que le sur-apprentissage survienne. L'apprentissage est alors meilleur : la précision de la validation est meilleure.

### Contrôle de l'apprentissage

Une des étapes importantes lors de l'apprentissage est d'identifier les meilleurs hyperparamètres pour le problème posé, cela implique souvent beaucoup d'expérimentations. Dans cette étude, les hyperparmaètres *(nombre de neurones par couche, optimisateur, taux de relâchement, etc)* ont été réglés en utilisant TensorBoard [10]. En partant de l'architecture de Yan LeCun, plusieurs hyperparamètres ont été réglés :

- La profondeur du réseau. Un bloc CONV-POOL peut être ajouté au réseau portant à 4 le nombre de couches de convolution. Théoriquement cela revient à rajouter une échelle de description spatiale. En ajoutant ce bloque le réseau possédera une échelle d'analyse des hiérarchies de l'image supplémentaire. Cela devrait aider le réseau à repérer des motifs plus complexes sur les images.
- Nombre de neurones de certaines couches. En effet le nombre de neurones des 2 ou 3 premières couches selon la profondeur du réseau sont variables : les nombres de neurones sont de 16, 32 ou 64 par couche.
- Élagage des connexions [11] : méthode qui permet d'éviter le surapprentissage tout en limitant la complexité du modèle. Elle consiste à supprimer des connexions (ou synapses), des entrées ou des neurones du réseau une fois l'apprentissage terminé. En pratique, les éléments qui ont la plus petite influence sur l'erreur de sortie du réseau sont supprimés.
- La taille du filtre de la première couche de convolution : un filtre de taille $3 \times 3 \times 3$ ou un filtre de taille $5 \times 5 \times 3$.

Les apprentissages ont été lancés sur 100 époques avec les paramètres suivants :

- Optimisateur [12] pour réseau profond : *Adam* [13].
- Fonction perte : *Cross-Entropy*.

La valeur de 100 époques a été déterminée selon plusieurs essais d'architectures. Cette valeur permet aux architectures testées de converger sans trop produire de surapprentissage.

9. valeur de la fonction perte pour l'optimisation.
10. TensorBoard est un plug-in python qui fournit la visualisation et l'outillage nécessaire à l'expérimentation d'apprentissage automatique :
    - Suivi et visualisation de mesures telles que la perte et la précision.
    - Visualisation du graphe du modèle (opérations et couches).
    - Affichage des histogrammes des poids, des biais ou d'autres tenseurs au fil du temps
11. Dropout en anglais.
12. L'optimisateur Contrôle le taux d'apprentissage afin d'avoir une convergence rapide.
13. Adaptive Moment Estimation.

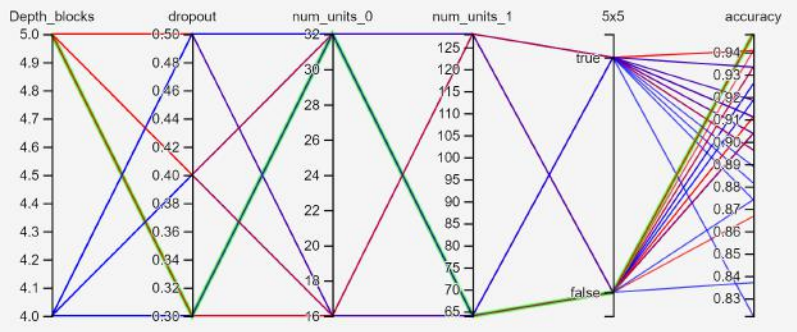



**Figure 4.11** – Réglage des hyperparamètres.

Sur la figure 4.11 est représenté la précision du réseau de neurones en fonction de ces paramètres. La couleur des traits corresponds à la profondeur du réseau, nous remarquons que les réseaux plus profonds en rouge obtiennent de meilleurs scores en moyenne. La présence du filtre $5 \times 5$ ne semble pas améliorer la précision en moyenne, au contraire. La meilleure configuration possède une précision de 0.95 et apparaît en trait vert. La configuration détaillée est la suivante :

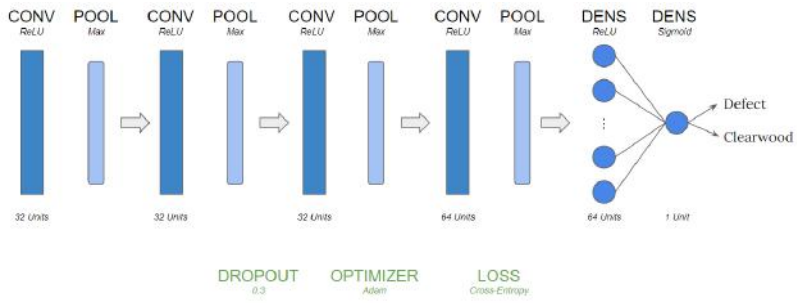



**Figure 4.12** – Architecture CNN1

### Amélioration de la précision

Pour améliorer la précision des réseaux de neurones, l'équipe d'ImageNet [16] réalise plusieurs prédictions sur une même image en effectuant quelques transformations : à partir d'une image de $256 \times 256$ pixels ils ont extrait 5 images recadrées (les 4 coins et le centre) de $224 \times 224$ pixels puis ils ont appliqué un retournement vertical sur chaque image, ils ont ainsi obtenu 10 images. Ces 10 images sont analysées par le réseau de neurones puis le score moyen de ces 10 images est calculé pour donner la prédiction finale du système de reconnaissance.

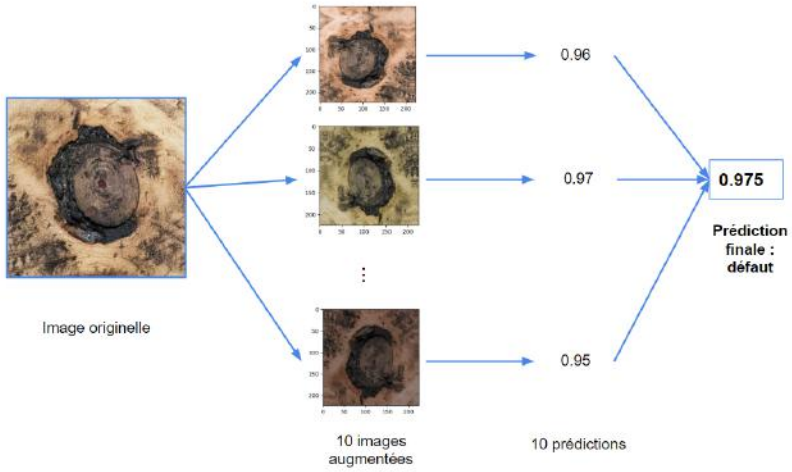



**Figure 4.13** – Prédiction moyennée sur plusieurs images

Nous avons utilisé une technique similaire, dans notre cas nous utilisons toutes les transformations utilisées pour l'augmentation et produisons 10 images avec des transformations régies de façon aléatoires. Plusieurs tests ont été effectués pour connaître le nombre d'images à utiliser pour réaliser la prédiction finale, les données sont résumées dans le tableau 4.1. Sur 10 simulations sur la base test de 135 images, la précision ainsi que la durée du calcul ont été collectées.

| Nombre d'images utilisées | Précision | Durée (s) |
|---|---|---|
| 1 | 93.65 % | 4.8 |
| 2 | 94.29 % | 8.0 |
| 5 | 94.44 % | 32.2 |
| 10 | 94.59 % | 72.6 |
| 20 | 94.74 % | 152.4 |
| 30 | 94.74 % | 232.2 |

**Table 4.1** – Utilisation de plusieurs images pour la prédiction finale

En moyenne, la précision augmente légèrement avec le nombre d'images utilisées pour la prédiction finale. Le nombre de 10 images a été choisi car au-delà l'amélioration de la précision n'est pas significative et le temps de calcul est grandement augmenté. Pour faire le parallèle avec la vision humaine, cette technique revient à observer selon plusieurs angles, selon des lumières différentes ou à des distances différentes l'objet avant de déterminer sa nature.

### 4.2.2 CNN2 - Classificateur multi-classes

Ce deuxième classificateur, plus complexe, est basé sur l'architecture proposée par Dan Nelson [14]. Ce classificateur devra être capable de distinguer 6 catégories : *bois sain, noeud vif, noeud mort, fente, échauffure/tâche et poche de résine.* La base de donnée étant divisée cette fois-ci en 6 classes, il y a beaucoup moins d'images par classe : 50 par classe (voir le tableau résumé 3.3).

14. Il s'agit d'un data scientist spécialisé dans les réseaux de neurones.

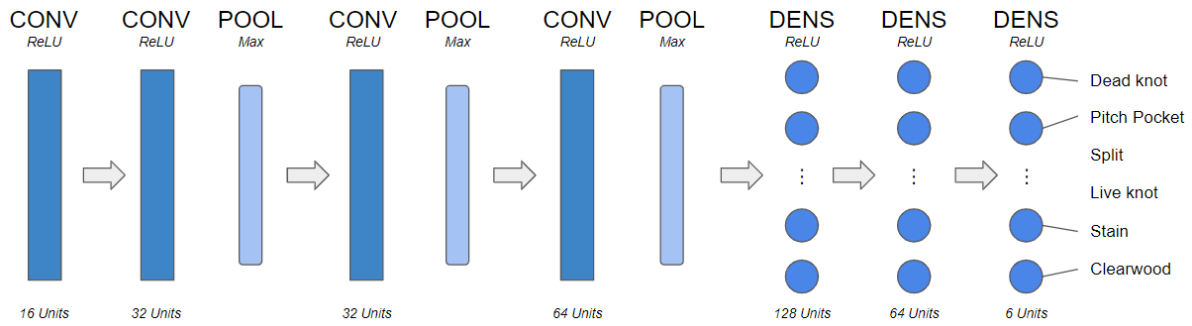



**Figure 4.14** – Architecture CNN2

Malheureusement la base de données est trop petite pour pouvoir entraîner un classificateur à 6 classes. Le réseau possède plus de 25 000 000 de paramètres (CNN1 en possède un peu plus de 600 000). L'entraînement aboutit à une précision inférieure à 15% pour l'apprentissage (cf figure 4.15).

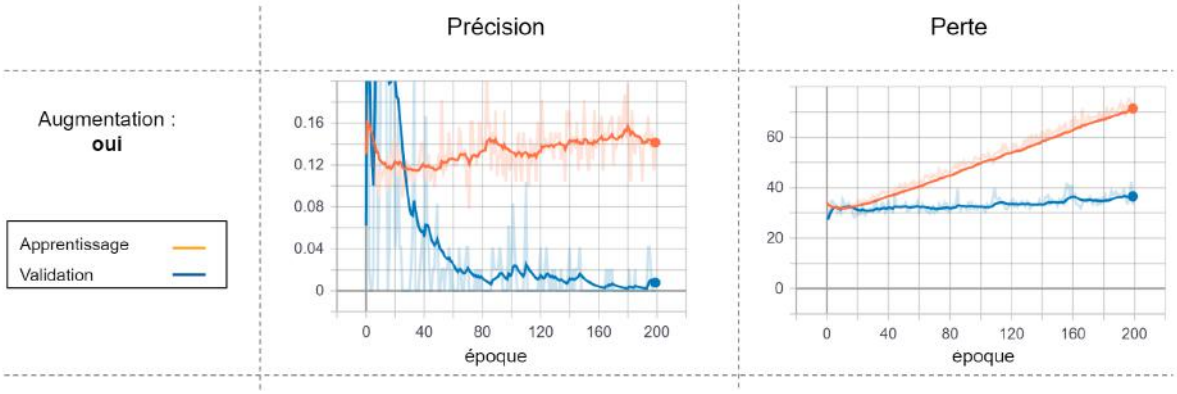



**Figure 4.15** – Apprentissage de CNN2

Dans ce cas là il n'y a simplement rien à faire, la taille de la base de donnée est l'élément limitant. Pour reconnaître les défauts, il est possible en revanche de fonctionner avec plusieurs classificateurs binaires à l'instar de K. Suzuki et al. (2003) dans le domaine médical [13].

### 4.2.3 RCNN - Classificateur multi-classes

Pour remplacer le classificateur multi-classes CNN2, un réseau de plusieurs classificateurs binaires pourrait être développé. En effet l'image serait d'abord analysée par le CNN1, l'image serait alors catégorisée comme défaut ou bois sain. Si l'image est reconnue comme un défaut alors elle est à nouveau analysée par de multiples classificateurs binaires pour déterminer de quel type de défaut il s'agit. Les classificateurs binaires ont moins de paramètres, il est donc plus simple de les entraîner sur des petites bases de données en comparaison d'un classificateur multi-classes.

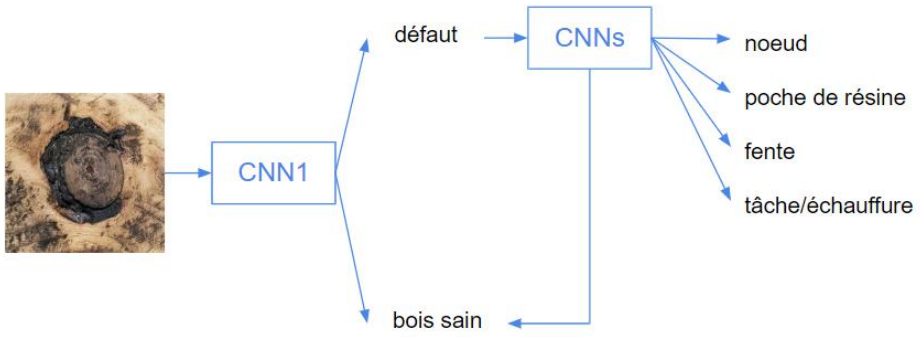



**Figure 4.16** – RCNN

Le bloc CNNs contient 4 classificateurs binaires :

- CNN3 : distinction noeuds - bois sain
- CNN4 : distinction poche de résine - bois sain
- CNN5 : distinction fente - bois sain
- CNN6 : distinction tâche/échauffure - bois sain

Chaque classificateur fonctionne en parallèle et donne un score compris entre 0 et 1. La probabilité que l'image soit un défaut est $p_{\text{défaut}} = s$, où $s$ est le score. La probabilité que l'image soit du bois sain est $p_{\text{bois sain}} = 1 - s$. Finalement, 4 probabilités de bois sain sont observées, elles sont moyennées. Par la suite le système déclare la catégorie via la probabilité la plus grande. Par exemple, sur la figure 4.17 les probabilités sont : **Noeud** 0.87, poche de résine 0.56, tâche/échauffure 0.52 et bois sain 0.43. Le système retient donc que l'image appartient à la catégorie Noeud.

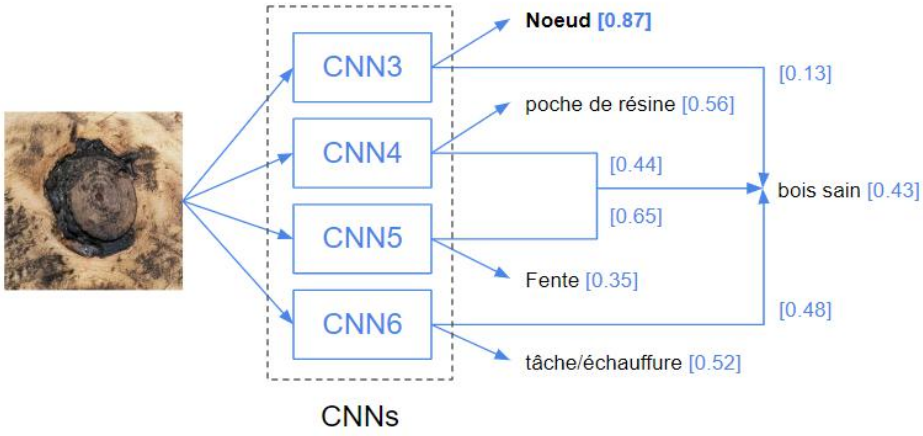



**Figure 4.17** – CNNs

Je n'ai pas eu le temps de développer ce système mais cela paraît être une alternative intéressante. Tout dépend de la précision des classificateurs entraînés sur des petites bases de données.

# Chapitre 5

# Résultats et Perspectives

## 5.1 Résultats de l'étude

### 5.1.1 Capacité à diminuer les faux positifs

Le développement du classificateur binaire a été réalisé avec la motivation de créer un algorithme capable de diminuer le taux de faux positifs. Pour rappel, au plus il y a de faux positifs au moins la classification du bois est précise par la suite. Cela peut mener à des pertes matérielles et économiques. Pour tester cette capacité, plusieurs banques de 80 images [1] ont été créés. Ces banques contiennent des défauts et des zones de bois saines. Le but est de tester la capacité de l'algorithme à détecter ces zones de bois saines et à les enlever de la banque d'image.

Plusieurs configurations de mélange ont été testées : 95%, 90%, 85% et 80% de défauts contenus dans la banque de 80 images. Ce taux est appelé taux de vrais positifs, il s'agit du nombre de défauts contenus dans une base de données ne devant contenir que des défauts. Une banque de 80 images est créée en choisissant des images aléatoirement parmi les 135 images de la base de test [2]. Pour chaque configuration, 5 banques de 80 images sont créées. Pour noter l'efficacité de ce réseau de neurones à cette tâche : les taux de *vrai positif*[3] avant passage et après passage de réseau de neurones sont comparés. Le taux de perte de vrai positif après passage du filtre est également analysé pour s'assurer que le filtre ne produit pas un nombre trop grand de *faux négatifs*[4].

| Taux de vrai positif avant filtrage | Taux de vrai positif après filtrage [5] | Perte de vrai positif [6] |
|---|---|---|
| 80.00% | 98.92% | 2.50 % |
| 85.00% | 97.71% | 2.65 % |
| 90.00% | 98.74% | 1.94 % |
| 95.00% | 99.74% | 2.63 % |

**Table 5.1** – Résultats du filtre à faux positifs moyennées sur 5 versions de chaque base de données pour 4 configurations différentes avant filtrage

1. Le nombre 80 a été choisi par rapport au nombre d'images de défauts disponibles dans la base de test : il faut assez d'images de défauts pour constituer une banque d'image avec 95% de défauts.
2. Les images de la base de validation et apprentissage ne sont pas utilisées car le réseau a appris sur ces deux banques d'images.
3. Un vrai positif est un défaut détecté comme étant un défaut.
4. Un faux négatif est un défaut détecté comme étant du bois sain.
5. Calculé de la façon suivante : $\frac{\text{Nombre de vrais positifs après filtrage}}{\text{Nombre d'images après filtrage}}$.
6. Calculé de la façon suivante : $\frac{\text{Nombre de vrais positifs perdus}}{\text{Nombre d'images avant filtrage}}$.

Le tableau 5.1 présente les résultats du passage du filtre à faux positifs pour différentes configurations avant filtrage : 80%, 85%, 90% et 95% de défauts contenues dans la base de données. Les résultats sont concluants, le filtre parvient à filtrer la plupart des zones de bois saines ce qui se traduit par une augmentation significative du taux de vrai positif après filtrage. Les images de bois sains que l'algorithme ne parvient par à filtrer ont été extraite de la base de données. Il s'avère que ces images sont très ambiguës, même pour un œil averti il est difficile de dire s'il s'agit de bois sain ou si un défaut est présent.


**Figure 5.1** – Images non détectées par le filtre - faux positifs : : bois sain détecté comme bois avec défaut

Sur la figure 5.1 sont présentées les images qui causent les faux positifs après filtrage. L'essence de bois de l'image 1 possède des cernes d'accroissement avec un fort contraste qui peuvent être identifiées comme des échauffures par le réseau. L'image 2 contient des parties plus foncées qui peuvent être des tâches d'origine fongique. L'image 3 contient le pourtour d'un noeud qui semble chargé en résine, cela peut être identifié comme étant une poche de résine. L'image 4 semble contenir un noeud. Ces images sont difficiles à classer. Elles ont d'ailleurs été classées dans la catégorie *bois sain* manuellement. Il est possible que le classement manuel soit source d'erreur et que cela ait une influence dans l'apprentissage. Par rapport aux images fournies l'algorithme identifie ces images comme ayant un défaut. Certaines de ces images *(3,4)* auraient pu être classées manuellement dans la catégorie défaut. Un des problème ici est donc l'erreur de classement manuel, une solution à envisager est de réaliser le classement manuel par plusieurs personnes pour diminuer ce taux d'erreur. L'autre problème pour les images (1,2) est que le bois sain peut avoir un aspect vraiment proche de certains défauts notamment les échauffures, seule une base de données plus grande et plus variée pourrait aider à résoudre ce problème.

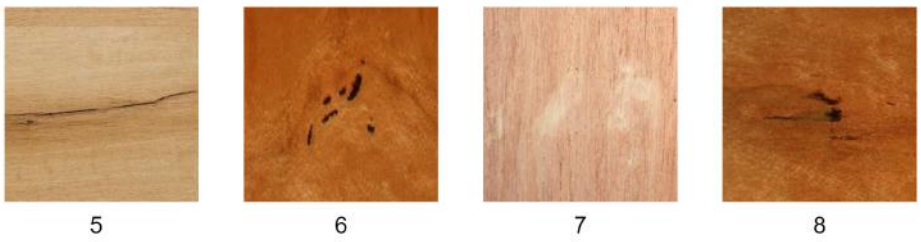



**Figure 5.2** – Images non détectées par le filtre - faux négatifs : bois avec défaut détecté comme bois sain

Sur la figure 5.2 sont présentées les images qui causent les faux négatifs après filtrage. Sur toutes ces images les défauts représentent une partie minime de l'image et le bois entourant le défaut semble être en excellent état. Cette fois, il est facile pour l'oeil humain de distinguer si ces images présentent des défauts ou non, l'image 5 possède une fente, l'image 6 et 8 possèdent des petites poches de résine. L'analyse de l'image 7 est plus délicate, de légères tâches blanches sont présentes. Concernant l'image 5, le réseau n'a pas dû détecter la fente car celle-ci est plus fine que les fentes présentes durant l'apprentissage, idem pour les poches de résines : cet aspect en amas n'est pas très présent dans la base d'apprentissage. En d'autres termes, le taux de faux négatifs pourrait être diminué grâce à une base de données plus grande et plus riche en "cas particulier". Des extraits de la base de données sont présents en Annexe A.

En outre, le réseau de neurones fonctionne très bien et l'identification est opérationnelle. Les capacités du réseau de neurones sont proches de celle d'un humain concernant les faux positifs puisque celui-ci à buter sur des images qui sont difficiles à classer ou qui ne peuvent être classées. En revanche, une base de données plus grande [7] pourrait diminuer le taux de faux négatifs ainsi que celui de faux positifs.

### 5.1.2 Capacité à agrandir la base de données

Le classificateur binaire peut également être utilisé pour agrandir la base de données qui a servi à son apprentissage. En effet, grâce à sa bonne précision supérieure à 90%, ce classificateur peut être utilisé pour classer de nouvelles images en deux catégories avec une certaine confiance : *bois sain* et *défauts* . Cela permettrait d'agrandir la base de données d'apprentissage afin de réaliser un nouvel apprentissage sur cette base de données agrandie. Ce nouvel apprentissage permettrait d'avoir un réseau plus précis en gardant la même architecture ou d'améliorer la précision en modifiant la profondeur du réseau. En effet, au plus la base de données est grande, au mieux un réseau à grand nombres de paramètres (et donc plus profond) peut être entraîné. Pour rappel, dans la section 4.2.1 nous avons vu que la profondeur du réseau jouait un rôle important dans la capacité du réseau à reconnaître des motifs complexes.

## 5.2 Conclusion

Dans cette étude, une base de données a été créée pour développer un réseau de neurones capable de distinguer une image de défaut et une image de bois sain. Cette base de données limitée en taille a permis de développer le classificateur binaire de façon satisfaisante. En utilisant plusieurs méthodes telles que l'augmentation pour l'apprentissage et l'augmentation pour la prédiction, l'algorithme atteint une précision largement supérieure à 90%. Face à ce succès, le développement d'un réseau de neurones plus complexe capable de reconnaître des catégories de défaut a été entrepris. Malheureusement, la taille de la base de données n'a pas permis de développer un modèle si complexe avec autant de paramètres. Face à ce problème, un système de plusieurs classificateurs binaires a été proposé mais non développé.

Malgré l'échec du deuxième réseau de neurones à cause d'une base de données trop petite, les principaux objectifs du stages sont remplis. Le but principal de ce stage était d'appliquer une technique de machine learning au service de la caractérisation du bois. Le classificateur binaire répond exactement à cette attente. Grâce à une base de données limitée il est capable de distinguer avec une précision suffisante les zones de bois saines des zones de défauts. La précision du réseau permet de l'utiliser comme filtre à faux positifs. Un problème récurrent dans la détection de défauts et qui est à l'origine, rappelons le, de pertes matérielles et économiques. Grâce à la base de données constituée qui présente de multiple essences de bois, le réseau de neurones devrait être généralisable à d'autres base de données avec une précision similaire . L'algorithme développé à base de CNN1 pour créer de nouvelles bases de données peut être très pratique pour des scieries ou autres industries qui n'utilisent qu'une seule essence de bois. En effet, cet algorithme pourrait être utilisé pour créer des base de données contenant une seule essence. Puis le réseau de neurones CNN1 pourrait être entraîné une nouvelle fois sur cette base de données. Le réseau de neurones serait alors plus performants sur cette essence.

Personnellement, ce stage m'a permis de me former au Machine Learning et au Machine Vision, domaines que je connaissais peu. J'ai pu comprendre toutes les étapes nécessaires au développement d'un outil de machine learning, notamment les points essentiels que sont l'acquisition et la préparation des données. J'ai pu également m'initier à plusieurs techniques pour contrer les effets néfastes d'une petite base de données. L'augmentation a permis de diversifier les images à chaque époque afin d'éviter le surapprentissage. Le contrôle de l'apprentissage et des hyperparamètres via tensorboard m'a permis d'optimiser le réseau de

7. Les bases de données généralement utilisées pour les réseaux de neurones présentent plus de 1M d'images [16] par classe contre 250 dans notre cas

neurones par rapport au problème donné. De surcroît, l'utilisation de l'augmentation dans la prédiction m'a permis d'améliorer légèrement la précision du réseau. Enfin, la manipulation de l'architecture de réseau de neurones (cf : section 4.2.1) m'a permis d'appréhender le processus de reconnaissance de motif via les couches de convolution et regroupement.

## 5.3 Perspectives

Ce travail a abouti sur la création d'un filtre à faux positifs performant. Cet outil pourrait être utilisé dans les scieries, plate-formes de réemploi ou tout autre établissement ayant recours au classement du bois. En effet, ce classement est la plupart du temps réalisé manuellement et rapidement. Utiliser le réseau de neurones permettrait d'obtenir un classement plus juste et régulier tout en garantissant un débit comparable voire meilleur. Le système à mettre en place pour analyser le bois est simple : un tapis roulant où passe le bois et une caméra reliée à une unité centrale qui réalise l'analyse en temps réel, la puissance de calcul nécessaire n'est pas conséquente mais dépend du débit souhaité. Grâce à ce dispositif, l'entreprise pourrait gérer de manière plus pertinente le risque client et le risque producteur : les faux positifs (bois sain considéré comme étant un défaut et pouvant amener à une dépréciation de classe du bois voir à sa perte) sont responsables du risque producteur, c'est-à-dire de la perte matérielle pour le producteur, à l'inverse les faux négatifs (défaut détecté comme étant du bois sain et pouvant amener à une sur-appréciation de la classe du bois et un mécontentement du client) sont responsables du risque client, c'est-à-dire d'une mauvaise évaluation du bois. Connaissant les taux de faux positifs et faux négatifs, l'entreprise pourrait statistiquement adopté la stratégie qui lui convient pour ajuster le rapport risque client/producteur. Enfin, le système pourrait également être utilisable via une application smartphone utilisant le module photo de celui-ci. Cela permettrait à des charpentiers ou tout autre corps de métier devant juger la qualité du bois de réemploi de réaliser un diagnostic complémentaire aux techniques utilisées aujourd'hui sur le terrain.

# Bibliographie


[1] J. L. Sandoz. Grading of construction timber by ultrasound*. *Wood Sci. Technol.*, 23 :95–108, 1989.

[2] McMillin C.W. Lin K. Vasquez-Espinosa Conners, R.W. Identifying and locating surface defects in wood : part of an automated lumber processing system. *Transactions on Pattern Analysis and Machine Intelligence PAMI-5*, (6) :573–583, 1983.

[3] Laurendeau D. Gagnon R. Lepage, R. Extraction of texture features with a multiresolution neural network. *Proceedings of the 3rd SPIE Conference on Applications of Artificial Neural Networks.*, 1709, 1992.

[4] Funck J.W. Brunner-C.C. Forrer J.B., Butler D.A. Image sweep-and-mark algorithms - part i. *Forest Products Journal*, 38 :75–79, 1988.

[5] Peleg S. Werman, M. Min-max operators in texture analysis. *Analysis and Machine Intelligence PAMI-7*, (6) :730–733, 1985.

[6] Funck J.W. Butler D.A., Brunner C.C. A dual-threshold image sweep-and-mark algorithm for defect detection in veneer. *Forest Products Journal*, 39(5) :25–28, 1989.

[7] Claudio A. Perez Gonzalo A. Ruz, Pablo A.Estévez. A neurofuzzy color image segmentation method for wood surface defect detection. *Forest Product Journal*, 55(4) :52–58, 2005.

[8] Patrick K. Simpson. Fuzzy min-max neural networks-part 1 : Classification. *Transations on neural networks*, 3(5), 1992.

[9] Patrick K. Simpson. Fuzzy min-max neural networks-part 2 : Clustering. *Transations on fuzzy systems*, 1(1), 1993.

[10] S Haykin. Neural networks, a comprehensive foundation. *Macmillan, Englewood, Cliffs*, page 696, 1994.

[11] D.T Pham and R.J Alcock. Automated grading and defect detection :areview. *Forest products journal*, 48(4) :34–42, 1998.

[12] Eric Goles Pablo A. Estévez, Claudio A. Perez. Genetic input selection to a neural classifier for defect classification of radiata pine boards. *Forest products journal*, 53(7/8) :87–94, 2003.

[13] Feng Li Shusuke Sone Kunio Doi Kenji Suzuki, Samuel G. Armato III. Massive training artificial neural network (mtann) for reduction of false positives in computerized detection of lung nodules in low-dose computed tomography. *Medical Physics*, 30 :1602–1617, 2003.

[14] Alcock R.J. Pham D.T. *Advances in Manufacturing.*, chapter 8 - Recent Developments in Automated Visual Inspection of Wood Boards. 1999.

[15] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. LSUN : construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[17] Yi Shan Qingqing Dang Gang Sun Ren Wu, Shengen Yan. Deep image : Scaling up image recognition. 2015.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[19] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[20] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[21] Danfei Xu Fei-Fei Li, Ranjay Krishna. Neural networks part 1 : Setting up the architecture.

[22] Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep ? *CoRR*, abs/1312.6184, 2013.

[23] Yoshua Bengio Patrick Haffner Yann LeCun, Léon Bottou. Gradient-based learning applied to document recognition. 1998.

# Annexe A

# Échantillons de la base de données


**Figure A.0.1** – Bois sain


**Figure A.0.2** – Noeuds morts


**Figure A.0.3** – Noeuds vifs


**Figure A.0.4** – Fentes

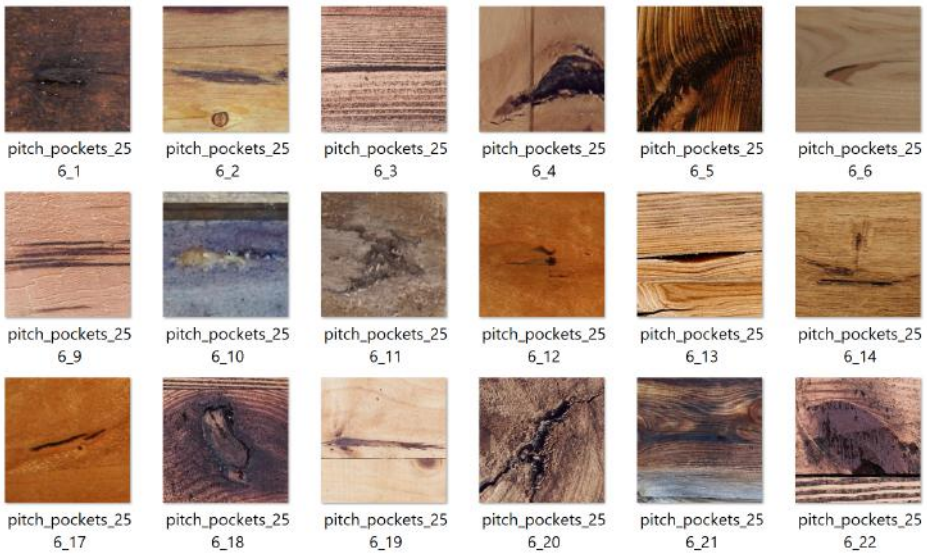



**Figure A.0.5** – Pôches de résine

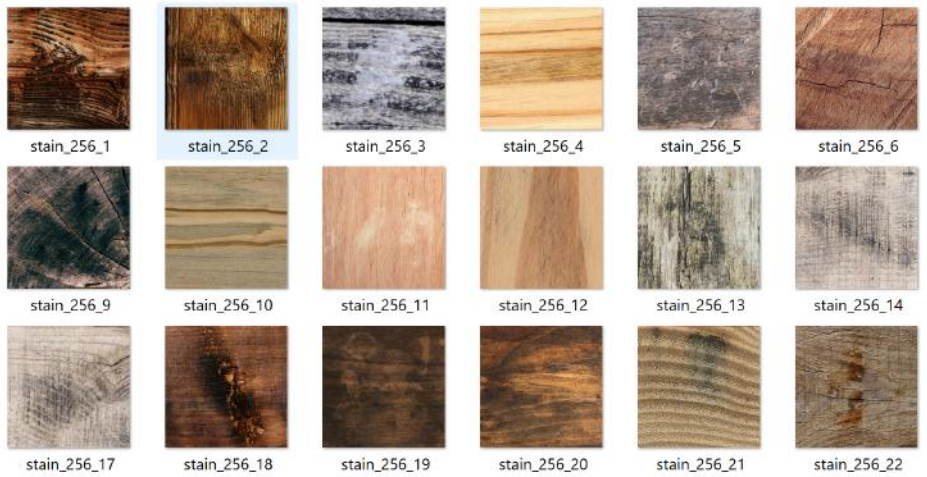



**Figure A.0.6** – Echauffures/tâches