Réalisée par : Encadrée par :

ELBOUBKRAOUI FARID Mr. ELBERMI LAHCEN

DODO TAHIROU ABDOUL SALAM

**Détection de Logiciels Malveillants par Classification d'Images Binaires avec des Réseaux de Neurones Convolutionnels (CNN)**

# Contexte

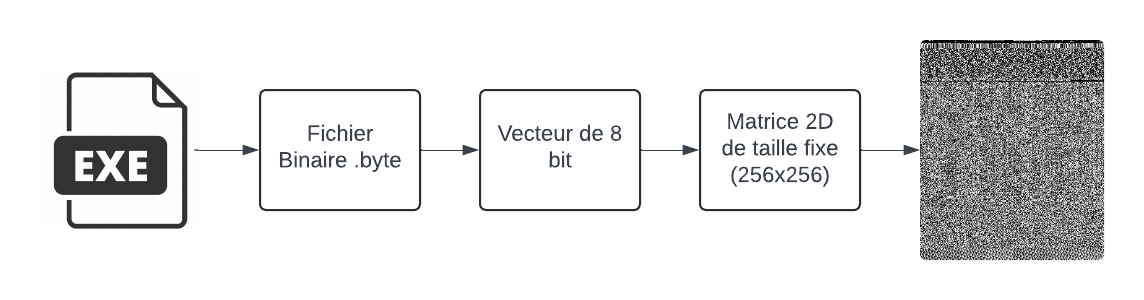
Les logiciels malveillants, ou "malwares", sont des programmes conçus pour causer des dommages ou des activités malveillantes sur des systèmes informatiques. Traditionnellement, la détection de malwares repose sur des méthodes comme l'analyse de signature ou l'analyse comportementale. Cependant, avec l'évolution constante des techniques de dissimulation et de polymorphisme, ces méthodes peuvent avoir des limites.

Une approche innovante consiste à transformer les fichiers binaires des malwares en images, puis à utiliser des techniques d'apprentissage automatique pour détecter des motifs ou des caractéristiques distinctes dans ces images. C'est ici que les CNN entrent en jeu.

# Transformation des fichier binaires en images

Transformation des binaires en images

Pour transformer un fichier binaire en image, l'idée est de lire le contenu du fichier et de le convertir en valeurs de pixels. on prend les octets du fichier binaire et on les mappe sur une grille de valeurs, chaque octet représentant un pixel d'une certaine intensité. Les images ainsi créées peuvent être de différentes tailles, selon la taille du fichier binaire et la manière dont vous choisissez de les représenter (par exemple, comme images en niveaux de gris ou en couleurs).



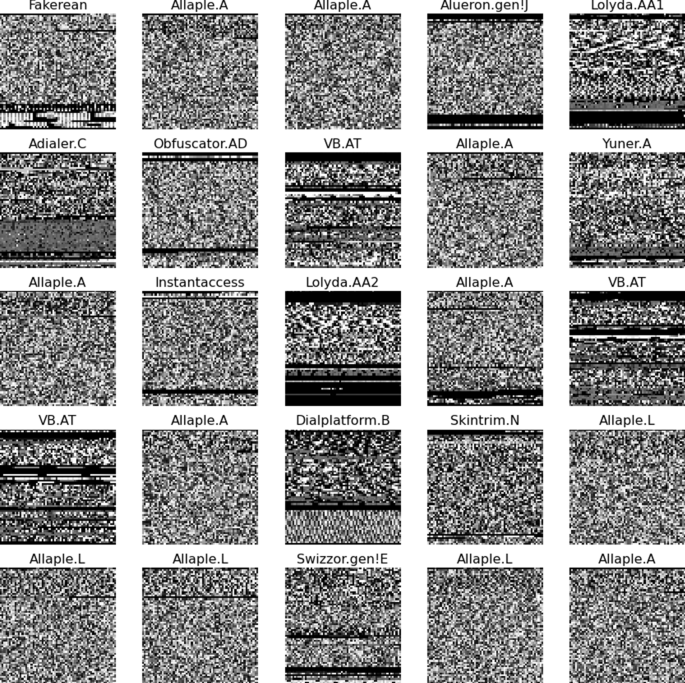
* 00000000 = 0 cad un pixel noir
* 11111111 = 255 cad un pixel blanc

# Utilisation des CNN pour la classification

Les CNN sont des réseaux de neurones particulièrement efficaces pour analyser des images, car ils ont la capacité de détecter des motifs visuels à différents niveaux de complexité. Voici comment ils sont utilisés pour la classification des images de malwares :

Prétraitement : Les images binaires des **malwares** et des programmes **légitimes** sont générées. Des techniques de normalisation (Compression, Padding …) peuvent être appliquées pour s'assurer que toutes les images ont des caractéristiques cohérentes.

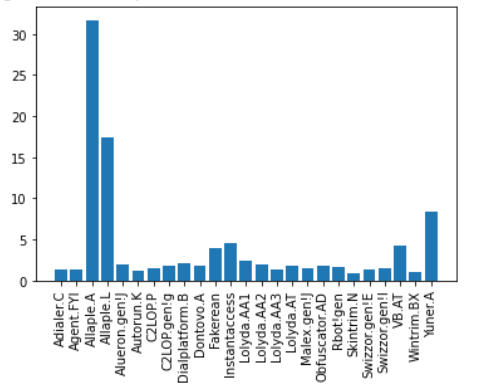
Exemple :



Constitution du dataset **:** Les images sont étiquetées en fonction de leur classification, créant un ensemble de données d'entraînement, Validation et de test. (80% – 10% - 10%)

On utilise l’ensemble de données Malimg contient 9 339 images byteplot de logiciels malveillants provenant de 25 familles différentes.

|  |  |  |  |
| --- | --- | --- | --- |
| # | Nom de Class | Type de Malware | Échantillons |
| 1 | Adailer | Dialer | 122 |
| 2 | Agent.FYI | Backdoor | 116 |
| 3 | Allaple.A | Worm | 2949 |
| 4 | Allaple.L | Worm | 1591 |
| 5 | Alueron.gen!J | Worm | 198 |
| 6 | AutoRun.K | Worm :AutoIT | 106 |
| 7 | C2LOP.P | Trojan | 146 |
| 8 | C2LOP.gen!g | Trojan | 200 |
| 9 | Dialplatform.B | Dialer | 177 |
| 10 | Dontovo.A | Trojan Download | 162 |
| 11 | Fakerean | Rogue | 381 |
| 12 | Instanccess | Dialer | 431 |
| 13 | Lolyda.AA1 | Password Stealer | 213 |
| 14 | Lolyda.AA2 | Password Stealer | 184 |
| 15 | Lolyda.AA3 | Password Stealer | 123 |
| 16 | Lolyda.AT | Password Stealer | 159 |
| 17 | Malex.gen!J | Trojan | 136 |
| 18 | Obfuscator.AD | Trojan Download | 142 |
| 19 | Rbot!gen | Backdoor | 158 |
| 20 | Skintrim.N | Trojan | 80 |
| 21 | Swizzor.gen !E | Trojan Download | 128 |
| 22 | Swizzor.gen !l | Trojan Download | 132 |
| 23 | VB.AT | Worm | 408 |
| 24 | Wintrim.BX | Trojan Download | 97 |
| 25 | Yuner.A | Worm | 800 |

****

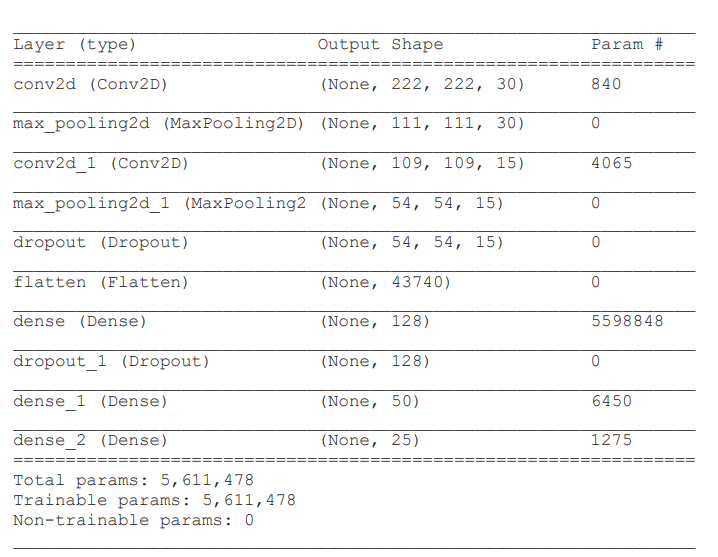
Entraînement du modèle **:** Un modèle CNN est construit avec des couches de convolution, de pooling (sous-échantillonnage), et des couches entièrement connectées. Le modèle est entraîné avec l'ensemble de données d'entraînement pour *apprendre les caractéristiques des images de malwares.*



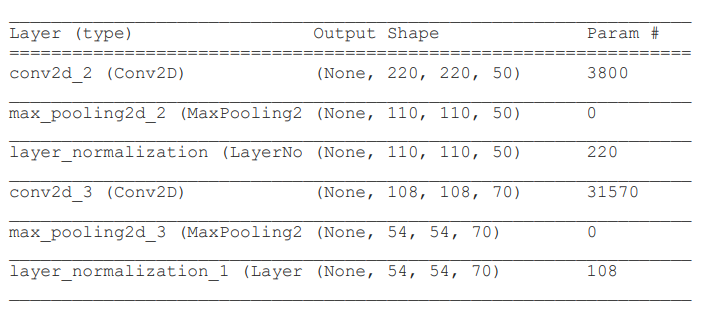
**Validation et test :** Le modèle est validé avec un ensemble de *données de test* pour mesurer sa performance. Des métriques **comme la précision, le rappel, et la F1-score** peuvent être utilisées pour évaluer l'efficacité du modèle.

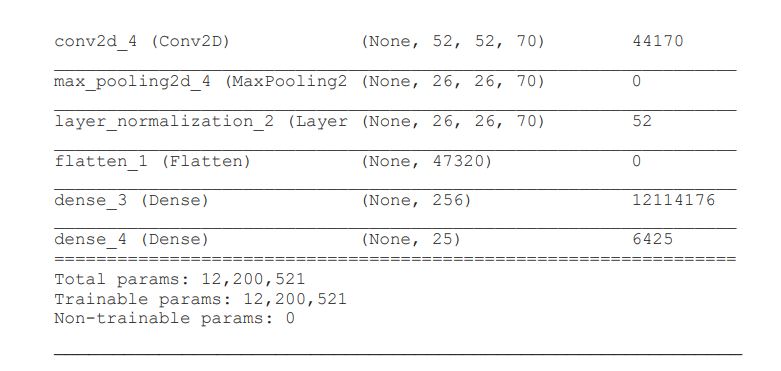
# Architecture Proposée :

Model 1 :

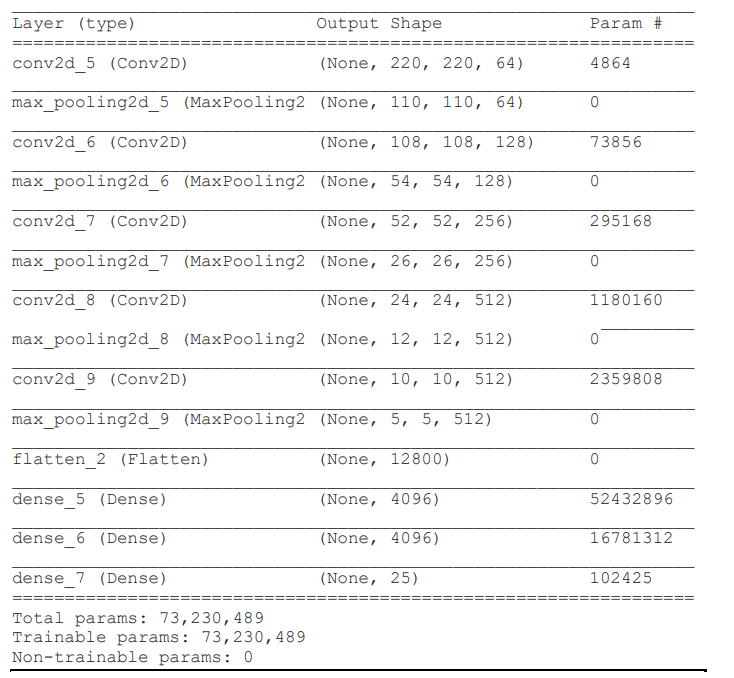


Model 2 :

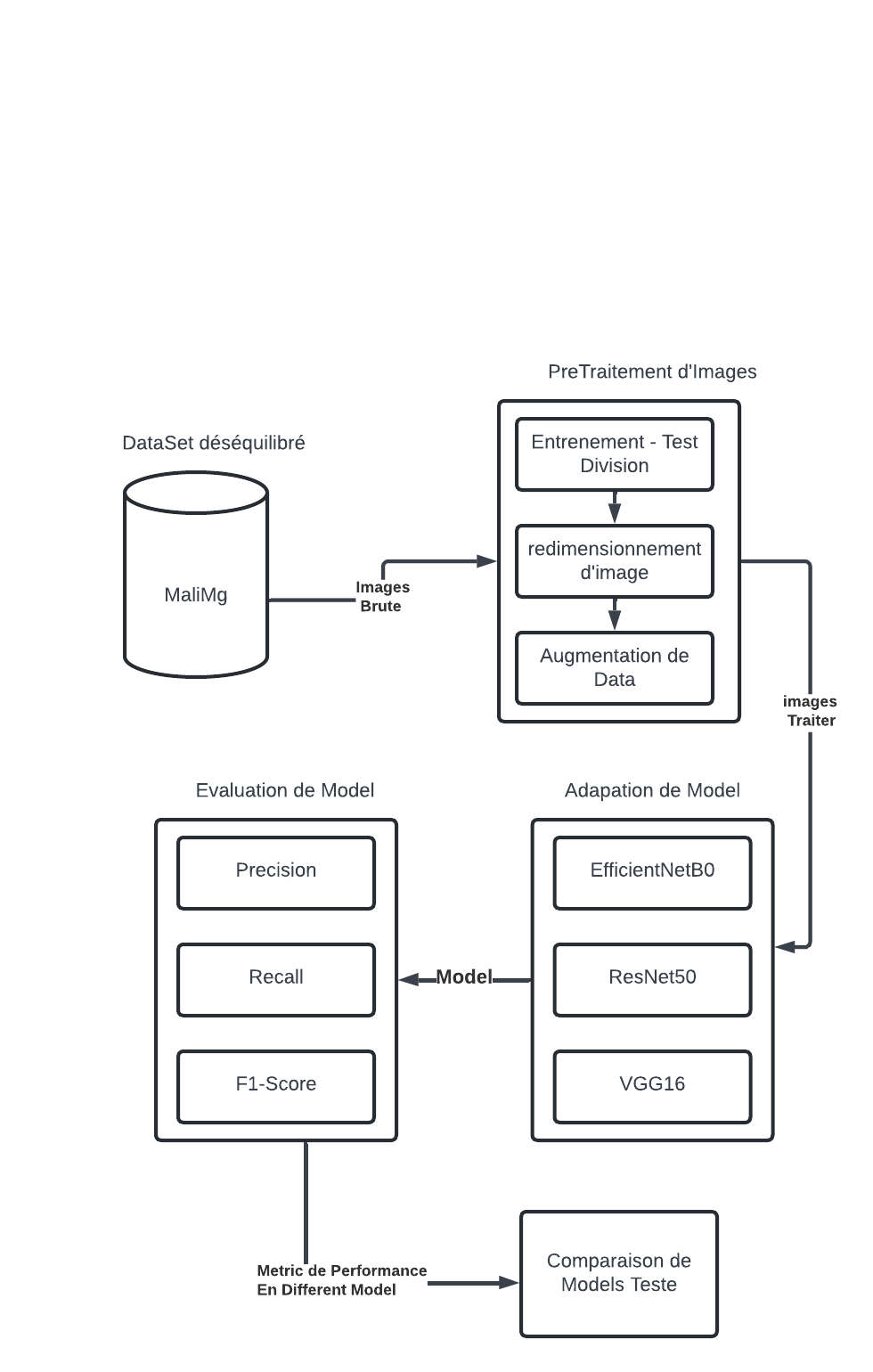




Model 3 :



# Diagramme d’Architecture :



# Réalisation de projet :

**Etape 1 :**

* Code de conversion de fichier exécutable vers une image niveau Gris
* Préparation et normalisation de dataSet

**Etape 2 :**

* Construction de model CNN de base
* Evaluation et Adaptation de Model
* L’ajout de classe : Logiciel Légitime
* Construction de 3 model CNN pour étude Comparatif

**Etape 3 :**

* Evaluation de model CNN
* Comparaison des Architectures tester

**Etape 4 :**

* Implémentation et Déploiement de Model Logiciel de Détections sous forme Plateforme web

**Extension de Projet :**

* L’Implémentation de Data Set : Malives équilibre qui contient des images RGB

<https://web.cs.hacettepe.edu.tr/~selman/malevis/>

* Expérimentation avec Data Set Mélangé (MaliMg + MaliVes)