Thèse de doctorat
Pour obtenir le grade de Docteur de
L'UNIVERSITÉ POLYTECHNIQUE HAUTS-DE-FRANCE
et l'INSA HAUTS-DE-FRANCE

Apprentissage profond non-supervisé : Application à la détection de situations anormales dans l'environnement du train autonome

Discipline, spécialité selon la liste des spécialités pour lesquelles l'École Doctorale est accréditée :

Informatique

Présentée et soutenue par **AMINE BOUSSIK**
Le 15/12/2023, à Valenciennes

Ecole doctorale : Polytechnique Hauts-de-France (ED PHF n°635)
Laboratoire : Laboratoire d'Automatique, de Mécanique et d'Informatique Industrielles et Humaines (LAMIH - UMR CNRS 8201)

JURY

Mme. LAETITIA JOURDAN	Présidente du jury,	Université de Lille
M. SMAIL NIAR	Directeur de thèse,	UPHF
M. YASSINE RUICHEK	Rapporteur,	UTBM
M. NOREDDINE ABGHOUR	Rapporteur,	Université Hassan II
M. LOTFI ABDI	Co-encadrant,	IRT Railenium
M. ABDELMALIK TALEB-AHMED	Co-directeur de thèse,	UPHF

Doctoral Thesis
To obtain the Doctoral degree from
POLYTECHNIC UNIVERSITY OF HAUTS-DE-FRANCE
and INSA HAUTS-DE-FRANCE

Unsupervised Deep Learning : Application to the Detection of Abnormal Situations in the Environment of Autonomous Trains

Discipline, specialty according to the list of specialties for which the doctoral school is accredited :

Computer Science

Presented and defended by **AMINE BOUSSIK**
On 15/12/2023, at Valenciennes

Doctoral School : Polytechnique Hauts-de-France (ED PHF n°635)
Laboratory : Laboratory of Industrial and Human Automation control Mechanical engineering and Computer science (LAMIH - UMR CNRS 8201)

JURY

Mrs. LAETITIA JOURDAN	Jury president,	University of Lille
Mr. SMAIL NIAR	Thesis director,	UPHF
Mr. YASSINE RUICHEK	Reviewer,	UTBM
Mr. NOREDDINE ABGHOUR	Reviewer,	Hassan II University
Mr. LOTFI ABDI	Co-supervisor,	IRT Railenium
Mr. ABDELMALIK TALEB-AHMED	Co-director of thesis,	UPHF

Résumé

La thèse aborde les défis du monitoring de l'environnement et de détection des anomalies, notamment des obstacles, pour un train de fret autonome. Bien que traditionnellement, les transports ferroviaires étaient sous la supervision humaine, les trains autonomes offrent des perspectives d'avantages en termes de coûts, de temps et de sécurité. Néanmoins, leur exploitation dans des environnements complexes pose d'importants enjeux de sûreté. Au lieu d'une approche supervisée nécessitant des données annotées onéreuses et limitées, cette recherche adopte une technique non supervisée, utilisant des données non étiquetées pour détecter les anomalies en s'appuyant sur des techniques capables d'identifier les comportements atypiques.

Deux modèles de surveillance environnementale sont présentés : le premier, basé sur un autoencodeur convolutionnel (CAE), est dédié à l'identification d'obstacles sur la voie principale ; le second, une version avancée incorporant le transformeur de vision (ViT), se concentre sur la surveillance générale de l'environnement. Tous deux exploitent des techniques d'apprentissage non supervisé pour la détection d'anomalies.

Les résultats montrent que la méthode mise en avant apporte des éléments pertinents pour la surveillance de l'environnement du train de fret autonome, ayant un potentiel pour renforcer sa fiabilité et sécurité. L'utilisation de techniques non supervisées démontre ainsi l'utilité et la pertinence de leur adoption dans un contexte d'application pour le train autonome.

Mots clés : intelligence artificielle, apprentissage profond, train autonome, détection d'anomalies, surveillance de l'environnement, train de fret autonome, transport ferroviaire, approche supervisée, approche non supervisée, algorithmes d'apprentissage, transformeur de vision, convolution.

Abstract

The thesis addresses the challenges of monitoring the environment and detecting anomalies, especially obstacles, for an autonomous freight train. Although traditionally, rail transport was under human supervision, autonomous trains offer potential advantages in terms of costs, time, and safety. However, their operation in complex environments poses significant safety concerns. Instead of a supervised approach that requires costly and limited annotated data, this research adopts an unsupervised technique, using unlabeled data to detect anomalies based on methods capable of identifying atypical behaviors.

Two environmental surveillance models are presented : the first, based on a convolutional autoencoder (CAE), is dedicated to identifying obstacles on the main track ; the second, an advanced version incorporating the vision transformer (ViT), focuses on overall environmental surveillance. Both employ unsupervised learning techniques for anomaly detection.

The results show that the highlighted method offers relevant insights for monitoring the environment of the autonomous freight train, holding potential to enhance its reliability and safety. The use of unsupervised techniques thus showcases the utility and relevance of their adoption in an application context for the autonomous train.

Keywords : artificial intelligence, deep learning, autonomous train, anomaly detection, environment monitoring, autonomous freight train, rail transport, supervised approach, unsupervised approach, learning algorithms, vision transformers, convolution.

Remerciements

Je souhaite prendre un instant pour exprimer ma sincère reconnaissance envers toutes les personnes qui ont joué un rôle crucial dans l'élaboration de cette thèse. En premier lieu, une pensée particulière à mes parents pour leur soutien sans faille, leur affection et l'inspiration qu'ils m'ont apportées, me permettant d'avancer avec détermination.

Je tiens également à exprimer ma gratitude envers tous les membres du jury, notamment Mme. Laetitia Jourdan en tant que présidente du jury et examinatrice de la thèse, ainsi que M. Yassine Ruichek et M. Noreddine Abghour en tant que rapporteurs de thèse.

Mes remerciements s'adressent également à mon directeur de thèse, M. Smail Niar, pour son accompagnement constant au cours de cette quête académique. Ses encouragements, ses retours constructifs et son aspiration à l'excellence ont été des piliers sur lesquels je me suis appuyé.

Je tiens aussi à exprimer ma gratitude à mon co-directeur de thèse, M. Abdelmalik Taleb-Ahmed, pour son expertise et ses conseils avisés, indispensables à la rédaction de ce travail.

Mes remerciements s'adressent également à M. Lotfi Abdi et M. Wael Ben Messaoud, mes encadrants professionnels. Leur générosité à partager leurs connaissances, leurs expériences, ainsi que leur accompagnement ont été essentiels pour contextualiser mes recherches et les appliquer dans un cadre concret et pertinent.

Ma gratitude va vers M. Sébastien Lefebvre, chef de projet du programme train de fret autonome, pour son expertise essentielle à ce projet. Je remercie aussi chaleureusement l'équipe de l'IRT Railenium pour leur professionnalisme et leur collaboration, qui ont renforcé la pertinence de ce travail.

Enfin, une pensée à tous mes amis qui m'ont soutenu moralement et qui ont été un véritable socle d'amitié durant ce parcours. Ma gratitude envers chacune de ces personnes est immense, tant pour leur apport à cette thèse que pour leur place dans ma vie. J'aimerais particulièrement remercier Rachid Kinta, Youssef Guedira, Mehdi Bouhamidi, Nabila Guennouni ainsi que ma fiancée Chaimae Raji pour leur soutien et leur amitié.

Sommaire

1 Introduction **1**

1.1 Contexte de la thèse . . . 1
1.2 Motivations et objectives . . . 2
1.3 Contributions . . . 3
1.4 Plan de la thèse . . . 4

2 Autonomie ferroviaire via l'IA **5**

2.1 Introduction . . . 5
2.2 Projet Train de Fret Autonome (TFA) . . . 5
2.2.1 Classification des niveaux d'automatisation dans les systèmes de véhicules autonomes . . . 7
2.2.1.1 Autonomisation : Définition . . . 7
2.2.1.2 Système Autonome . . . 10
2.2.2 Briques du système autonome ferroviaire TFA . . . 12
2.2.3 Contraintes et défis . . . 13
2.3 Intelligence artificielle et apprentissage profond . . . 14
2.3.1 Apprentissage automatique et apprentissage profond . . . 16
2.3.1.1 Fonctionnement . . . 19
2.3.1.2 Architecture des réseaux de neurones artificiels (ANN) . . . 22
2.3.1.3 Architecture des réseaux de neurones convolutionnels (CNN) 24
2.3.1.4 Exemples de datasets de vision . . . 25
2.3.1.5 Exemples d'architectures des CNNs . . . 31
2.4 Détection d'anomalies . . . 35
2.4.1 Définition . . . 35
2.4.2 Difficultés et défis . . . 38
2.4.3 Approches utilisées et output . . . 39
2.4.4 Modèles globaux de détection d'anomalies . . . 40
2.4.5 Modèles pour divers domaines . . . 42
2.4.6 Domaines d'application . . . 43
2.5 Conclusion . . . 44

3 Etat de l'art **45**

3.1 Introduction . . . 45
3.2 Détection d'obstacles ferroviaires via l'apprentissage profond . . . 48

3.2.1 Détection via apprentissage supervisé . . . 48
3.2.2 Détection par apprentissage non-supervisé . . . 58
3.3 Détection par d'autres approches . . . 64
3.4 Conclusion . . . 71
3.4.1 Synthétisation des travaux . . . 72

4 Détection d'obstacles ferroviaires sur la voie principale 76
4.1 Introduction . . . 76
4.2 Architecture proposée . . . 78
4.2.1 Autoencodeur convolutionnel . . . 78
4.2.2 Etude exploratoire . . . 79
4.2.2.1 Exploration des fonctions de pertes . . . 79
4.2.2.2 Exploration des fonctions d'activations . . . 81
4.2.2.3 Exploration des optimiseurs . . . 83
4.3 Dataset . . . 86
4.3.1 Manipulation du dataset . . . 88
4.3.2 Pretraitement du dataset . . . 89
4.3.3 Extraction des Régions d'intérêts . . . 90
4.3.3.1 Génération des boîtes englobantes . . . 91
4.3.3.2 Extraction des clusters de la voie de circulation principale 96
4.3.3.3 Délimitation de la RoI . . . 98
4.3.4 Dataset d'entraînement . . . 99
4.3.5 Dataset de validation . . . 100
4.4 Résultats expérimentaux . . . 103
4.4.1 Critères de sélection et gap-score . . . 104
4.4.2 Méthode Multi-critères TOPSIS . . . 105
4.4.3 Hyperparamètres . . . 106
4.4.4 Résultats . . . 107
4.4.4.1 Evaluation et classement des modèles . . . 108
4.4.4.2 Test sur un scénario réel . . . 109
4.5 Conclusion . . . 112

5 Détection au niveau de la scène ferroviaire 113
5.1 Introduction . . . 113
5.2 Composante transformeur de vision . . . 114
5.2.1 Définition de l'attention dans les transformeurs . . . 114
5.2.2 Le mécanisme de l'attention . . . 115
5.3 Autoencodeur à base de transformeur de vision . . . 118
5.3.1 Encodeur ViT . . . 118
5.3.2 Décodeur . . . 120

5.3.3 Architecture globale . . . 121
5.4 Datasets . . . 122
5.4.1 Dataset d'entraînement . . . 122
5.4.2 Dataset de test . . . 125
5.5 Résultats expérimentaux . . . 127
5.5.1 Résultats qualitatifs . . . 129
5.5.2 Résultats quantitatifs . . . 130
5.6 Conclusion . . . 136

6 Conclusion **137**
6.1 Conclusion . . . 137
6.2 Travaux futurs et améliorations . . . 138

Table des figures

1 Locomotive utilisée sur le projet TFA avec les capteurs montés. 7
2 Définition des niveaux d'automatisation SAE [9]. 8
3 Définition des grades d'automatisation ferroviaire GoA [10]. 9
4 Schéma fonctionnel global pour le fonctionnement opérationnel d'un véhicule autonome. 11
5 Les différents domaines et disciplines de l'IA (deep learning & machine learning) [23]. 16
6 La différence entre l'intelligence artificielle, l'apprentissage automatique et l'apprentissage profond [24]. 17
7 La classification des algorithmes d'apprentissage automatique et des sous-catégories de chaque classe avec certains exemples de cas d'utilisation données aux niveau de chaque sous-catégorie [25]. 19
8 La distribution des erreurs de la fonction $\mathcal{E}$. 21
9 Architecture d'un réseau de neurones artificiel [26]. 23
10 Analogie entre les neurones artificiels et un véritable neurone du cerveau humain [27]. 24
11 Exemple d'une opération de convolution a) et une opération de déconvolution b) [28]. 25
12 Exemple d'une architecture d'un CNN pour classifier les images intitulée VGG-16 [54]. 25
13 Aperçu du dataset MNIST [46]. 26
14 Aperçu du dataset CIFAR [47]. 27
15 Aperçu du dataset ImageNet [48]. 28
16 Aperçu du dataset COCO [128]. 28
17 Aperçu du dataset LfW [49]. 29
18 Aperçu du dataset ADE20K [50]. 29
19 Aperçu du dataset KITTI [129]. 30
20 Aperçu du dataset RailSem19 [218]. 30
21 Aperçu de l'architecture LeNet [51]. 31
22 Aperçu de l'architecture AlexNet [53]. 31
23 Aperçu de l'architecture VGG [54]. 32
24 Un diagramme simplifié de l'architecture GoogleNet [56]. 33
25 Aperçu de l'architecture ResNet, spécifiquement ResNet-18 [115]. 33

26 Aperçu de l'architecture DETR [57]. 34
27 Aperçu de l'architecture YOLO (spécifiquement YOLOV2) [123]. 34
28 Aperçu de l'architecture U-Net [59]. 35
29 Exemple d'anomalies ponctuelles en relation avec les fraudes bancaires. . . 36
30 Exemple d'une anomalie contextuelle : La zone bleue indique des températures normales, la verte une baisse durant l'hiver (janvier à mars), mais une chute drastique des températures en juillet-aout (partie C1) suggère une anomalie contextuelle. 37
31 Exemple d'une anomalie collective [61]. 38
32 Évolution des recherches sur la détection d'obstacles ferroviaire et la voiture autonome de 2010 à 2022 - Scopus 47
33 Évolution générale des recherches sur l'apprentissage profond et les techniques de traitement d'images 2004 à 2022 - Scopus 47
34 Évolution de la recherche de 2010 à 2023 : Analyse comparative des travaux sur la détection d'obstacles ferroviaires et de voiture autonome par approche IA, ainsi que de la détection globale des obstacles dans les deux domaines. 47
35 Résultats de détection basée sur UAVs [106] 49
36 Résultats de la détection des obstacles par la méthode décrite dans [118] . 50
37 Aperçu des différentes classes dans le dataset utilisé dans [118] 50
38 Performances du modèle FE-SSD [109] 51
39 Performances d'Improved-YOLOv4 dans les RoIs [122] 52
40 Résultats de détection sur des données synthétique avec la méthode multitâche [117] 53
41 Résultat de détection du travail proposé [120] 53
42 Résultat de détection du travail proposé [131] 54
43 Résultat de détection du travail proposé [132] 55
44 Résultat de détection de la méthode proposée sur des scènes ferroviaires avec des voyageurs [136] 56
45 Résultat de détection de la méthode proposée sur des scènes ferroviaires avec des voyageurs et des travailleurs [136] 57
46 Figure montrant les résultats de l'approche utilisée [209] 57
47 Aperçu des différentes classes à détecter par la composante supervisée du travail proposé [142] 59
48 Résultat de détection du travail proposé [142] 59
49 Aperçu de la méthode décrite dans [151] 60

50 Image exemple où l'approche décrite dans [151] détecte l'obstacle correctement contrairement aux méthodes de base. La première ligne montre les cartes de segmentation prédites : le gris pour les zones externes, le noir pour la voie et le blanc pour la détection d'anomalies. La seconde ligne indique la localisation de l'anomalie 61
51 Résultat de détection du travail proposé [150] 62
52 Résultat de détection du travail proposé [156] 63
53 Description de la méthode utilisée et résultats de la détection du travail proposé [196] 65
54 Figure du capteur de détection LROD [204] 66
55 Description des résultats de la détection du travail proposé [208] 67
56 Figure de l'approche basée sur des capteurs physiques TRINETRA [213] . 68
57 Figure montrant les résultats de l'approche utilisée [197] 69
58 Figure montrant les résultats de l'approche utilisée [205] 70
59 Figure montrant les résultats de l'approche utilisée [201] 71

60 Description de l'architecture de l'autoencodeur convolutionnel utilisée dans cette étude exploratoire 79
61 Figure montrant les différentes formes graphiques des fonctions d'activations : ReLU, ELU, SELU, Mish, Swish, Tanh, Mila et Sharkfin 83
62 Description du fonctionnement de l'algorithme Adam à l'aide d'un pseudo-code. Les paramètres du pseudo-code sont : α est le pas utilisée pour ajuster la matrice de poids W, et ∇_W représente la dérivée directionnelle de W [234] 84
63 Description du fonctionnement de l'algorithme RAdam à l'aide d'un pseudo-code [235] 85
64 Description du fonctionnement de l'algorithme NovoGrad à l'aide d'un pseudo-code. Les paramètres du pseudo-code sont : Le taux d'apprentissage λ_0, les moments β_1, β_2, le facteur de décroissance des poids d, et le nombre d'étapes T [236] 86
65 Aperçu des images de scènes ferroviaires du dataset railsem19 [218] 87
66 Description des différentes classes (instances) présentes dans le dataset railsem19 [218] 87
67 Description des différentes classes (instances) présentes dans le dataset railsem19 [218] 88
68 Résumé des étapes suivies pour élaborer le pipeline de prétraitement du dataset railsem19 89
69 Exemples d'images supprimées lors de la creation du sous-dataset de railsem19 90
70 Exemples d'images normalisées illustrant le cas "normal" 90

71 Description graphique de l'architecture du modèle YOLOV3-tiny [125] . . . 92
72 Etapes suivies lors de la création des boîtes englobantes à partir des masques railsem19, la première ligne montre des images de scènes ferroviaires railsem19, la deuxième ligne montre des images des masques correspondants, la troisième ligne montre la génération d'annotation de boîtes englobantes à partir des masques des voies de circulation. 95
73 Résultats de détection du modèle YOLOV3-Tiny sur une vidéo tout le long du trajet Toulouse Matabiau - Paris Montparnasse. 96
74 Figure montrant les étapes de filtrage des boîtes englobantes. Les figures de la première colonne représentent les images résultantes de la détection avec YOLOV3-Tiny. La deuxième colonne montre les différents clusters générés avec l'algorithme HCA [183], avec imposition des contraintes pour la séléction du cluster représentatif de la voie de circulation principale (en rouge). La troisième colonne montre la même scène ferroviaire initiale avec suppression des boîtes englobantes des autres voies de circulation. 98
75 Génération d'une forme pour délimiter les RoIs à partir des clusters de la voie de circulation principale et extraction d'un masque 99
76 Aperçu du dataset résultant. La première ligne montre les images originales railsem19. La deuxième ligne montre les masques extraits à partir du pipeline de filtrage pour extraire les RoIs. La troisème ligne montre les images des RoIs. 100
77 Fonctionnement du framework GP-GAN [149] : Pour une image composite x, on génère d'abord une version simplifiée, l'image de basse résolution $\tilde{x}_l$, grâce au Blending GAN $G(x)$. Cette version utilise x_1 comme entrée, qui représente le niveau le moins détaillé de l'image à partir d'une structure appelée pyramide laplacienne. L'ajustement de cette image utilise un modèle mathématique, l'équation de Gaussian-Poisson $H(x_h)$, qui est régulée par les contraintes $C(x_h)$ et $P(x_h)$ pour préserver les éléments essentiels tout en modifiant les détails. À partir de cette opération, une image plus détaillée, $\tilde{x}_{1h}$, est obtenue. 102
78 Aperçu du dataset de validation. La première ligne d'images montre les images de validation sans obstacles (backgrounds). La deuxième ligne d'images montre les mêmes backgrounds avec incrustation GP-GAN [149] (incrustations). La troisème ligne montres les RoIs d'images incrustées. 103

79 Aperçu des résultats qualitatifs montrant quatre scénarios différents. La première et la troisième ligne d'images montrent quatre exemples d'images de backgrounds sans aucun obstacle suivis de leur région d'intérêt et de la reconstruction de la même région par le meilleur modèle. La deuxième et dernière ligne d'images montrent les homologues incrustées des backgrounds précédemment montrés, chacun avec leurs régions d'intérêt respectives et les reconstructions par le même modèle.. 110
80 Résultat de détection sur le premier segment du travail proposé : l'axe x illustre l'ID des frames de la vidéo, tandis que l'axe y indique le gap-score en pourcentage (%) 111
81 Résultat de détection sur le deuxième segment du travail proposé : l'axe x illustre l'ID des frames de la vidéo, tandis que l'axe y indique le gap-score en pourcentage (%) 112

82 Tête d'auto-attention par produit scalaire (Scaled Dot-product Self Attention head) 117
83 Auto-attention multi-têtes (Multi-head self-attention) utilisant plusieurs têtes d'auto-attention par produit scalaire 118
84 Représentation d'un bloc ViT 120
85 Représentation du decodeur à base de convolutions 120
86 Architecture globale hybride ViT 121
87 Exemples d'images à partir des acquisitions vidéos des baselines 123
88 Exemples d'images du dataset après prétraitement 124
89 Autres exemples d'images du dataset d'entraînement 125
90 Exemples d'images prétraitées illustrant le dataset de test 126
91 Résultats de détection d'anomalies sur le dataset via des heatmaps. Les zones bleues indiquent des erreurs faibles, signifiant probablement aucune anomalie, tandis que les zones rouges signalent des erreurs élevées, pointant des anomalies. Ces couleurs facilitent une analyse visuelle rapide des anomalies 129
92 Exemples d'images générées représentants des masques d'obstacles ainsi que les images originales 132
93 La courbe ROC moyenne pour l'ensemble du dataset de test. La courbe illustre la performance du modèle en traçant le taux de vrais positifs (TPR ou sensibilité) en fonction du taux de faux positifs (FPR). La ligne diagonale représente une prédiction aléatoire ; un modèle performant se situe au-dessus de cette ligne, et un modèle non performant en dessous. 133

94 La zone sous la courbe ROC (AUROC) pour l'ensemble du dataset de test. L'AUROC représente la probabilité qu'une anomalie détectée (via la heatmap) soit classée au-dessus d'une zone non-anormale (mask). La zone colorée illustre l'aire sous la courbe ROC, qui quantifie la capacité globale du modèle à détecter les anomalies. 134
95 Comparaison du modèle proposée avec d'autres modèles. 135

Liste des tableaux

2.1 Table des valeurs d'entrée et de sortie désirées representant un exemple d'un dataset labelisé simplifié. 20
2.2 Synthèse des modèles pour la détection d'anomalies. 42
2.3 Domaines d'application de la détection d'anomalies 44

3.1 Synthèse des travaux mettant en jeu des approches supervisées. 73
3.2 Synthèse des travaux mettant en jeu des approches non-supervisées. 74
3.3 Synthèse des travaux mettant en jeu des approches basées sur le traitement d'images et l'utilisation de capteurs physiques. 75

4.1 Caractéristiques du jeu de données railsem19 88
4.2 Structure et paramètres du réseau YOLOv3-tiny [125] 93
4.3 Comparaison des performances de différentes variantes de YOLO (YOLO V1, YOLOV3-Tiny, YOLOV3), ainsi que les modèles de ségmentation d'instances (DB-Swin-L, SoTR, Mask R-CNN). (↑) : Plus élevé est meilleur, (↓) : Plus bas est meilleur. Les modèles de détection d'objets (YOLO) sont entraînés et utilisé par le GPU Pascal Titan X, tandis que les modèles de ségmentation d'instances (∗) utilisent le GPU RTX 3090 [191, 192] 93
4.4 Résumé du dataset de validation généré. On distingue un set de validation qui se compose uniquement d'images sans obstacles afin de juger les performances du modèle. De plus, à l'aide du modèle GP-GAN [149] on fixe 19 types de backgrounds, chacun avec un nombre d'incrustations qui varie entre 10 et 20. 103
4.5 Résumé des hyperparamètres 107
4.6 Résultats de l'évaluation basée sur les modèles générés (top 10) avec une importance égale pour les deux critères avec les poids suivants, $w_1 = 0.5$ pour les scores d'écart et $w_2 = 0.5$ pour les scores positifs. 109
4.7 Résultats de l'évaluation basée sur les modèles générés (top 10) avec plus d'importance accordée au nombre de scores positifs pour chaque arrière-plan avec les poids suivants, $w_1 = 0.1$ pour les scores d'écart et $w_2 = 0.9$ pour les scores positifs. 111

5.1 Synthèse des segments d'anomalies et événements divers dans les vidéos des baselines 8, 8 Bis et 9 126
5.2 Résumé des hyperparamètres et configurations 128

5.3 Comparaison des AUROCs et gain relatif des modèles par rapport au modèle hybride ViT. 135

Nomenclature

AE Autoencoder (EN) / Autoencodeur (FR)

ANN Artificial Neural Networks (EN) / Réseaux de Neurones Artificiels (FR)

ATO Automatic Train Operation (EN) / Fonctionnement Automatique de Trains (FR)

AUPRC Area Under the Precision-Recall Curve (EN) / Aire sous la courbe de la précision-rappel (FR)

AUROC Area Under the ROC Curve (EN) / Aire sous la courbe ROC (FR)

CAE Convolutional Autoencoder (EN) / Autoencodeur Convolutionnel (FR)

CIFAR Canadian Institute For Advanced Research (EN) / Institut Canadien pour la Recherche Avancée (FR)

CMOS Complementary Metal-Oxide-Semiconductor (EN) / Semi-conducteur Complémentaire à Oxyde Métallique (FR)

CNN Convolutional Neural Network (EN) / Réseau Neuronal Convolutif (FR)

COCO Common Objects in Context (EN)

CVAE Convolutional Variational Autoencoder (EN) / Autoencodeur Variationnel Convolutif (FR)

DBSCAN Density-Based Spatial Clustering of Applications with Noise (EN) / Regroupement spatial basé sur la densité des applications avec bruit (FR)

DETR DEtection TRansformers (EN)

DL Deep Learning (EN) / Apprentissage Profond (FR)

DNN Deep Neural Network (EN) / Réseau Neuronal Profond (FR)

DSR Dual Subspace Re-projection (EN)

EDO Environment Obstacle Detection (EN) / Environnement Détection d'Obstacles (FR)

ELU Exponential Linear Unit (EN) / Unité Linéaire Exponentielle (FR)

FLOPS Floating Point Operations Per Second (EN) / Opérations Flottantes par Seconde (FR)

FN False Negatives (EN) / Faux Négatifs (FR)

FP False Positives (EN) / Faux Positifs (FR)

FPR False Positive Rate (EN) / Taux de Faux Positifs (FR)

FPS Frames Per Second (EN) / Images Par Seconde (FR)

GAN Generative Adversarial Network (EN) / Réseau Adversarial Génératif (FR)

GD Gradient Descent (EN) / Descente de Gradient (FR)

GISE Global Information Surveillance Environment (EN) / Environnement Global de Surveillance de l'Information (FR)

GISM Global Information Surveillance Manager (EN) / Gestionnaire Global de Surveillance de l'Information (FR)

GPGAN Gaussian-Poisson GAN (EN) / Gaussian-Poisson GAN (FR)

GPS Global Positioning System (EN) / Système de Positionnement Global (FR)

GPU Graphics Processing Unit (EN) / Unité de Traitement Graphique (FR)

HCA Hierarchical Clustering (EN) / Regroupement hiérarchique (FR)

IA Artificial Intelligence (EN) / Intelligence Artificielle (FR)

IEC International Electrotechnical Commission (EN) / Commission Électrotechnique Internationale (FR)

IEEE Institute of Electrical and Electronics Engineers (EN) / Institut des Ingénieurs Électriciens et Électroniciens (FR)

IEMN Institute of Electronics, Microelectronics, and Nanotechnology (EN) / Institut d'électronique de microélectronique et de nanotechnologie (FR)

ILSVRC ImageNet Large Scale Visual Recognition Challenge (EN)

IP Image Processing (EN) / Traitement d'Image (FR)

IPM Inverse Perspective Mapping (EN) / Mappage de Perspective Inverse (FR)

IR Infrared (EN) / Infrarouge (FR)

IRT Technological Research Institute (EN) / Institut de Recherche Technologique (FR)

KITTI Karlsruhe Institute of Technology and Toyota Technological Institute (EN) / Institut de Technologie de Karlsruhe et Institut Technologique de Toyota (FR)

KNN K Nearest Neighbors (EN) / K Plus Proches Voisins (FR)

LAMIH Laboratory of Automatic Control, Mechanics, and Industrial and Human Computer Science (EN) / Laboratoire d'Automatique, de Mécanique et d'Informatique Industrielles et Humaines (FR)

LFW Labeled Faces in the Wild (EN)

$LIDAR$ Light Detection and Ranging (EN) / Détection et Télémétrie par Lumière (FR)

LOF Local Outlier Factor (EN) / Facteur d'Outlier Local (FR)

$LROD$ Long Range Obstacle Detection (EN)

$LSTM$ Long Short-Term Memory Cell (EN) / Cellule de Longue Mémoire à court Terme (FR)

MAE Mean Absolute Error (EN) / Erreur Absolue Moyenne (FR)

$MCDM$ Multi-Criteria Decision Making (EN) / Prise de Décision Multi-Critère (FR)

ML Machine Learning (EN) / Apprentissage Automatique (FR)

MLP Multi-Layer Perceptron (EN) / Perceptron Multi-Couche (FR)

$MNIST$ Modified National Institute of Standards and Technology (EN) / Institut National des Normes et de la Technologie (FR)

$MS-SSIM$ Multi-Scale Structural Similarity Index (EN) / Indice de Similitude Structurelle Multi-Échelle (FR)

MSA Multi-head Self-Attention (EN) / Auto-attention Multi-têtes (FR)

MSE Mean Squared Error (EN) / Erreur Quadratique Moyenne (FR)

NAV Navigation (EN) / Navigation (FR)

NLP Natural Language Processing (EN) / Traitement Automatique du Langage Naturel (FR)

NS Unsupervised (EN) / Non-Supervisé (FR)

NVD Normalized Vector Distance (EN) / Distance de Vecteur Normalisée (FR)

$OC-SVM$ One-Class Support Vector Machine (EN) / Machine à Vecteurs de Support à Une Classe (FR)

$OTSU$ Otsu's Thresholding Method (EN) / Méthode de Seuillage d'Otsu (FR)

PAL Protection Abstraction Layer (EN) / Couche d'Abstraction de Protection (FR)

$PETA$ PEdestrian Attribute (EN)

$PSNR$ Peak Signal-to-Noise Ratio (EN) / Rapport Signal sur Bruit en Pic (FR)

$RCNN$ Regional Convolutional Neural Network (EN) / Réseau Neuronal Convolutif Régional (FR)

$RFOD$ Rail Foreign Obstacle Detection (EN)

RGB Red-Green-Blue (EN) / Rouge-Vert-Bleu (FR)

ROC Receiver Operating Characteristic (EN) / Caractéristique de Fonctionnement du Récepteur (FR)

RROD RailRoad Obstacle Detection (EN)

SA Self Attention (EN) / Auto-attention (FR)

SAE Sparse Autoencoder (EN) / Autoencodeur épars (FR)

SELU Scaled Exponential Linear Unit (EN) / Unité Linéaire Évolutive Auto-Normalisée (FR)

SGD Stochastic Gradient Descent (EN) / Descente de Gradient Stochastique (FR)

SIAL Signal Interpretation and Abstraction Layer (EN) / Couche d'Interprétation et d'Abstraction de la Signalisation (FR)

SNCF National Society of French Railways (EN) / Société Nationale des Chemins de fer Français (FR)

SS Semi-Supervised (EN) / Semi-Supervisé (FR)

SSD Single Shot Detector (EN) / Détecteur Single Shot (FR)

SSIM Structural Similarity Index (EN) / Indice de Similarité Structurelle (FR)

SUAL Supervision Abstraction Layer (EN) / Couche d'Abstraction de Supervision (FR)

SVM Support Vector Machine (EN) / Machine à Vecteurs de Support (FR)

TFA Autonomous Freight Train (EN) / Train de Fret Autonome (FR)

TN True Negatives (EN) / Vrais Négatifs (FR)

TOPSIS Technique for Order of Preference by Similarity to Ideal Solution (EN) / Technique de Préférence par Similarité à une Solution Idéale (FR)

TP True Positives (EN) / Vrais Positifs (FR)

TPR True Positive Rate (EN) / Taux de Vrais Positifs (FR)

UAV Unmanned Aerial Vehicle (EN) / Véhicule Aérien Sans Pilote (FR)

UPHF Polytechnic University of Hauts-de-France (EN) / Université Polytechnique Hauts-de-France (FR)

VAE Variational Autoencoder (EN) / Autoencodeur Variationnel (FR)

VGG Visual Geometry Group (EN) / Groupe de Géométrie Visuelle (FR)

ViT Vision Transformer (EN) / Transformeur de Vision (FR)

YOLO You Only Look Once Algorithm (EN)

Chapitre 1

Introduction

1.1 Contexte de la thèse

La croissance technologique a joué un rôle crucial dans la révolution des systèmes de transport. Une application clé est l'utilisation des systèmes de détection d'obstacles dans le transport ferroviaire. La détection d'obstacles a connu une transformation significative au cours de son histoire, passant de techniques rudimentaires telles que l'inspection manuelle et les capteurs infrarouges passifs, à des méthodes sophistiquées comme la vision par ordinateur et l'intelligence artificielle (IA) [1]. Cette thèse vise à étudier l'application des techniques d'apprentissage non supervisé dans la détection d'obstacles et le monitoring de l'environnement pour les trains autonomes.

L'histoire de la détection d'obstacles remonte à l'origine des systèmes de transport. Les techniques traditionnelles de détection d'obstacles utilisaient la vigilance humaine et des dispositifs sensoriels simples. Le développement rapide de la technologie au fil des années a facilité l'intégration de mécanismes de détection avancés. Par exemple, la recherche sur les systèmes de sécurité des véhicules à la fin du 20ème siècle a introduit des capteurs comme le radar et le LiDAR pour la détection d'obstacles, une technique qui est devenue courante dans l'industrie automobile [2].

La tendance à l'automatisation à l'ère moderne nécessite des moyens plus sophistiqués pour un fonctionnement efficace et sûr. Cette nécessité est particulièrement critique dans le contexte des systèmes ferroviaires, qui ont un haut potentiel pour des accidents catastrophiques en cas de défaillance dans la détection ou l'évitement d'obstacles. L'IA, spécifiquement l'apprentissage profond, a été identifié comme une solution potentielle à ces défis. Les récentes avancées de cette technologie offrent la possibilité d'apprendre des motifs complexes, de s'adapter à de nouveaux scénarios et de faire des prédictions précises, ce qui peut améliorer considérablement l'efficacité et la sécurité des opérations ferroviaires [3].

L'apprentissage profond, et plus spécifiquement l'apprentissage non supervisé, offre des promesses dans ce contexte. Contrairement aux techniques d'apprentissage supervisé, l'apprentissage non supervisé ne nécessite pas de données étiquetées pour l'entraînement, ce qui peut être un avantage dans les scénarios où l'obtention de données étiquetées est difficile ou coûteuse [4]. Dans le contexte de la détection d'obstacles et le monitoring de

l'environnement, cela pourrait signifier la capacité d'apprendre à partir de données brutes, d'identifier des obstacles et des conditions potentiellement dangereuses sans avoir besoin d'un étiquetage manuel des données.

Diverses études de recherche ont soutenu l'application de l'apprentissage non supervisé dans le secteur des transports. Des travaux récents ont démontré le potentiel des techniques d'apprentissage non supervisé dans divers aspects de la sécurité ferroviaire, tels que la surveillance de l'état des voies [5] et la maintenance prédictive [7]. Cependant, l'application spécifique de ces techniques à la détection d'obstacles et au monitoring de l'environnement pour les trains autonomes est un domaine qui mérite une investigation plus approfondie.

1.2 Motivations et objectives

Cette thèse s'inscrit à la croisée d'un enjeu industriel majeur et d'un intérêt croissant pour l'application de technologies de pointe dans le domaine du transport. À l'heure actuelle, l'industrie du transport est en pleine mutation, stimulée par des innovations technologiques disruptives et une volonté grandissante d'automatisation. Le secteur ferroviaire n'échappe pas à cette tendance.

Le travail de recherche se place donc dans le cadre du projet intitulé Train de Fret Autonome (TFA). Ce projet est le fruit d'une collaboration multidisciplinaire impliquant plusieurs acteurs industriels de premier plan, tels que la SNCF, Thales, Alstom et Capgemini Engineering, mais aussi des partenaires académiques tels que l'Université Polytechnique Hauts-de-France (UPHF). La thèse bénéficiera également de l'expertise de plusieurs laboratoires de recherche renommés, dont le Laboratoire d'Automatique de Mécanique et d'Informatique Industrielles et Humaines (LAMIH) et l'Institut d'Électronique de Microélectronique et de Nanotechnologie (IEMN).

En tant que thèse à caractère industriel appliqué, l'un des principaux défis à relever concerne le suivi précis et efficace de l'environnement du train autonome, en d'autres termes, la voie de circulation principale et l'emprise ferroviaire. Ainsi, la motivation qui sous-tend ce travail de recherche réside dans la volonté d'aborder ce défi en exploitant les méthodes d'apprentissage profond non supervisé, afin d'améliorer le monitoring de l'environnement dans le contexte des trains autonomes. C'est une occasion inestimable d'adapter l'utilisation de l'IA dans un cadre industriel et de contribuer de manière significative à l'évolution des systèmes de transport.

L'objectif principal de cette thèse est de développer et d'améliorer les techniques d'apprentissage profond non supervisé pour le monitoring de l'environnement autour des voies ferrées. Plus précisément, cette thèse vise à :

— Comprendre et évaluer l'efficacité des techniques d'apprentissage profond non supervisé existantes dans le contexte du suivi de l'environnement pour les trains

autonomes.

— Développer de nouvelles méthodes et architectures basées sur l'apprentissage profond non supervisé qui peuvent améliorer la précision et l'efficacité du monitoring de l'environnement.

— Intégrer l'IA avec le système matériel spécialisé dans la détection d'obstacles intitulé Environnement Détection d'Obstacles (EDO), pour créer un système de surveillance environnementale synergique et robuste.

1.3 Contributions

La thèse vise à réaliser des contributions notables dans le domaine du transport ferroviaire autonome.

La première contribution est une analyse exhaustive de l'état de l'art, qui passe en revue les travaux récents sur l'application de l'IA dans le domaine ferroviaire. Cette analyse inclut à la fois les méthodes basées sur l'apprentissage profond et les approches plus traditionnelles, telles que le traitement d'images et les méthodes basées sur les capteurs. Cet étude approfondie, qui se manifeste sous la forme d'une revue, établit une base solide pour les travaux de recherche subséquents. Les détails de cette contribution sont disponibles dans la section 3.

Ce travail a été concrétisé par un article de journal : Amine, Boussik, et al. "Comparative Assessment of Deep Learning, Image Processing, and Sensor-Based Approaches for Railway Obstacle Detection : A Comprehensive Review". IEEE Neural Networks.

La deuxième contribution est une nouvelle méthode de détection d'anomalies basée sur un autoencodeur convolutionnel. Cela permet un suivi plus précis de la voie principale de circulation. En partant d'un ensemble de données publiques, RailSem19 [218], notre approche se distingue de l'état de l'art par une étude exploratoire et l'application d'une nouvelle méthode d'évaluation. Cela nous a permis d'extraire les modèles les plus performants, en combinant des ensembles de données synthétiques et réels pour le test. Plus de détail sont disponibles dans la section 4.

Cette contribution a été concrétisée par plusieurs conférences :

— Conférence internationale : Boussik, Amine, et al. "Railway Obstacle Detection Using Unsupervised Learning : An Exploratory Study." 2021 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2021.

— Conférence internationale : Amine, Boussik, et al. "Vision-based railway track extraction and obstacle detection using deep learning for autonomous train." The 2nd International Workshop on Artificial Intelligence for RAILwayS (AI4RAILS). 2021.

— Conférence nationale : Amine, Boussik, et al. "Détection des obstacles dans l'environnement du train autonome par approche machine learning". In 50ème Congrès

ATEC ITS France 2022.

Enfin, la troisième contribution de la thèse concerne l'amélioration des autoencodeurs par l'introduction de composantes de pointe en matière de vision par ordinateur, notamment les transformeurs de vision (ViT) [134]. Nous avons adapté une architecture d'autoencodeurs, entraîné et testé notre modèle sur un dataset d'images de scènes ferroviaires unique. Ce dataset a été regroupé à partir de baselines de test effectuées sur le matériel roulant pour valider le fonctionnement des capteurs, ce qui illustre l'application pratique et la pertinence de notre travail pour le secteur ferroviaire. Une présentation plus détaillée de cette contribution peut être consultée dans la section 5.

Cette contribution a été concrétisée par une Conférence internationale (en cours) : Amine, Boussik, et al. "Autonomous Train's Environment Monitoring using Unsupervised Learning Based on Vision Transformers.".

1.4 Plan de la thèse

La thèse commence par une introduction générale au nivau du chapitre 1, qui décrit le contexte de l'étude, les motivations, les objectifs, les contributions ainsi que le plan de la thèse. Elle poursuit avec le Chapitre 2, ce dernier met l'accent sur le projet TFA, le système autonome et ses composants, ainsi que sur les contraintes liées au projet TFA. Il met également en lumière l'importance de l'intelligence artificielle, l'apprentissage profond ainsi que la détection d'anomalies dans ce contexte.

Le Chapitre 3 traite l'état de l'art, y compris la détection des obstacles ferroviaires et le monitoring de l'environnement par l'apprentissage profond, et la détection par capteurs physiques et traitement d'images.

Le Chapitre 4 se consacre à la détection des obstacles ferroviaires sur la voie de circulation principale en utilisant un autoencodeur convolutionnel. Ce chapitre présente une étude détaillée de l'architecture proposée, la préparation du jeu de données, et les résultats expérimentaux obtenus.

Dans le Chapitre 5, l'accent est mis sur la détection au niveau de la scène ferroviaire à l'aide d'un autoencodeur à base de transformeur de vision. Les détails sur l'architecture et les résultats sont également présentés dans ce chapitre.

Enfin, le Chapitre 6 conclut la thèse en proposant des directions pour des travaux futurs.

Chapitre 2

Autonomie ferroviaire via l'IA

2.1 Introduction

De nos jours, la technologie fait des progrès significatifs. Des secteurs divers et variés sont impactés, y compris le transport ferroviaire. Face à ces changements, l'intégration de l'IA dans les systèmes ferroviaires devient une perspective de plus en plus pertinente. Ce chapitre met l'accent sur cette intersection, en se concentrant particulièrement sur le projet Train de Fret Autonome (TFA).

La première partie de ce chapitre est dédiée à l'étude du concept de train de fret autonome. Nous y examinerons l'idée du système autonome et les divers éléments qui composent un tel système dans le contexte ferroviaire. En outre, un éclairage spécifique sera apporté aux défis du projet et aux contraintes liées à la thèse.

La deuxième partie du chapitre se concentre sur la définition et les aspects de l'intelligence artificielle. Elle explore des concepts tels que l'apprentissage automatique (*Machine Learning*), l'apprentissage profond, les techniques d'entraînement associées, ainsi que les différents types de modèles spécialisés en vision par ordinateur, comme les CNNs, et présente des exemples de datasets de vision publiques.

Enfin, la troisième partie se penche sur un aspect crucial lié au thème du monitoring de l'environnement : la détection d'anomalies. Nous définirons ce concept, ses objectifs, les approches généralement utilisées pour y parvenir, ainsi que les modèles spécifiquement conçus pour cette tâche.

À l'issue de ce chapitre, nous esquisserons une conclusion récapitulative, faisant le point sur les acquis et en dégageant les perspectives futures. Ce tour d'horizon, bien que non exhaustif, a pour vocation d'offrir une vision claire et concise du rôle potentiel de l'IA dans l'évolution vers un système ferroviaire autonome.

2.2 Projet Train de Fret Autonome (TFA)

Le projet français collaboratif TFA est une initiative ambitieuse qui réunit plusieurs partenaires industriels de premier plan, dont la SNCF, Alstom Transport, Hitachi Rail STS, Capgemini Engineering, et Airbus Protect ainsi qu'une multitude d'acteurs scientifiques tels que l'IRT Railenium, l'université polytechnique Hauts-de-France, le Laboratoire

d'Automatique de Mécanique et d'Informatique Industrielles et Humaines (LAMIH) et l'Institut d'Électronique de Microélectronique et de Nanotechnologie (IEMN). L'objectif ultime de ce projet, lancé en 2019, est de concevoir un prototype opérationnel d'un train de fret autonome d'ici 2023. Ce train devrait être capable de circuler en mode GOA4 (voir section 2.2.1) sans nécessiter d'intervention humaine. Pour atteindre cet objectif, plusieurs éléments clés doivent être développés, dont une architecture robuste pour le train autonome et différents modules qui le composent.

Le projet fait face à des défis technologiques stimulants. Pour certains aspects du train autonome, des solutions technologiques existent déjà et nécessitent simplement des ajustements et des intégrations. Toutefois, d'autres éléments vont au-delà de ce qui existe actuellement ou sont encore insuffisamment matures, ce qui demande donc des efforts de recherche supplémentaires. Parmi les domaines où l'innovation est particulièrement cruciale, on trouve le monitoring de l'environnement du train autonome, c'est-à-dire la brique de perception. Notre thèse se concentre sur une partie essentielle de la brique de perception du train de fret autonome. Elle met l'accent sur l'utilisation de l'apprentissage profond, plus précisément l'apprentissage non supervisé. Cette thèse vise à proposer des solutions pour garantir un monitoring efficace de l'environnement du train autonome à partir des capteurs montés sur la locomotive, comme indiqué dans la figure 1.

L'objectif principal de cette thèse est de développer des solutions innovantes basées sur l'apprentissage profond, en particulier l'apprentissage non supervisé, pour permettre au train de fret autonome de comprendre et interpréter son environnement. En identifiant avec précision les différents éléments présents à proximité des voies, ce système de perception fournira des données cruciales qui seront utilisées pour éclairer la prise de décision ultérieure du train autonome. Le périmètre de cette thèse ne se concentre pas sur la prise de décision du train de fret autonome, mais plutôt sur l'assurance d'une partie essentielle de la perception de l'environnement. Cette perception sera utilisée comme entrée pour la brique de prise de décision du système autonome.

En identifiant les différents éléments présents à proximité des voies, ce système de perception fournira des données essentielles qui seront utilisées pour guider la prise de décision du train autonome. Assurant ainsi une partie fondamentale de la perception, cette thèse renforcera la capacité du train de fret autonome à opérer de manière sûre et efficace dans un environnement ouvert et incertain, soit un espace où les conditions et les variables ne sont pas toujours prévisibles ou contrôlables, confrontant le train à des éléments extérieurs variés et imprévus. Les informations fournies par ce système de perception joueront un rôle déterminant dans l'amélioration des performances globales du train autonome et garantiront un haut niveau de sécurité lors de ses trajets.

Dans le cadre du projet TFA, l'objectif de cette thèse est d'affiner la détection des anomalies telles que les obstacles et aléas au sein de l'environnement du train de fret autonome, centré principalement sur la voie de circulation et le contexte adjacent. Plutôt

que d'interagir directement, le but est d'identifier et analyser ces irrégularités pour fournir des informations pertinentes à la brique de prise de décision du système autonome. Cette démarche s'inscrit dans une volonté d'améliorer la capacité du train à circuler avec une plus grande efficacité et fiabilité en tenant compte des aléas potentiels de son environnement.



FIGURE 1: Locomotive utilisée sur le projet TFA avec les capteurs montés.

2.2.1 Classification des niveaux d'automatisation dans les systèmes de véhicules autonomes

2.2.1.1 Autonomisation : Définition

Bien qu'il n'existe pas de standard unique, diverses classifications ont été créées pour évaluer le degré d'automatisation d'un véhicule. L'objectif est d'établir plusieurs repères qui réduisent progressivement le rôle de l'homme dans le pilotage du véhicule, ce qui nous aide à mieux comprendre l'évolution de la technologie. Dans cette discussion, nous examinerons deux classifications qui nous permettront d'élaborer une définition de l'autonomie.

Dans le secteur automobile, la *Society of Automotive Engineers* (SAE) a introduit 6 niveaux d'automatisation (*Levels of automation*) allant d'une voiture entièrement manuelle (niveau 0) à une voiture entièrement autonome (niveau 5) [9]. Cette classification est expliquée en détail dans la figure 2.

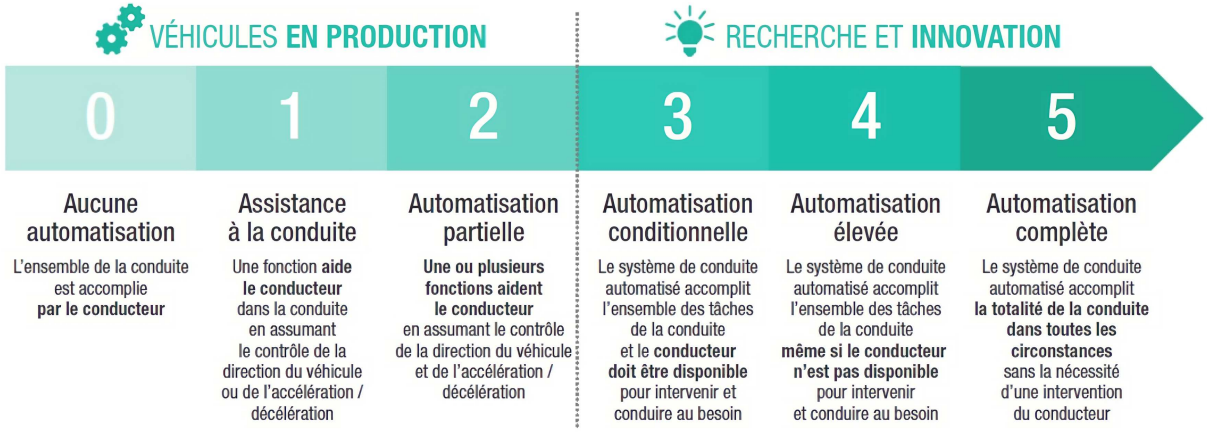

FIGURE 2: Définition des niveaux d'automatisation SAE [9].

Par ailleurs, dans le secteur ferroviaire, le degré d'automatisation *Grade of Automation* (GoA), défini par la norme internationale IEC 62290-1, propose 5 niveaux allant d'un train entièrement géré par l'homme (GoA0) à un train totalement autonome (GoA4) [10]. Cette classification est illustrée dans la figure 3.

Tâches / GoAs	Accélération/ Décélération	Départ/Arrêt aux stations/	Surveillance de l'environnement de conduite	Fermeture/ Ouverture des portes	Détection et gestion des urgences
GoA0 	Conducteur	Conducteur	Conducteur	Conducteur	Conducteur
GoA1 	Conducteur/ Système	Conducteur	Conducteur	Conducteur	Conducteur
GoA2 	Système/ Conducteur	Conducteur/ Système	Conducteur	Conducteur	Conducteur
GoA3 	Système	Système	Système	Contrôleur de train	Contrôleur de train
GoA4 	Système	Système	Système	Système	Système/ Personnel

FIGURE 3: Définition des grades d'automatisation ferroviaire GoA [10].

Le *Grade of Automation* (GoA) est un indicateur utilisé pour évaluer et classer le degré d'automatisation d'un véhicule ferroviaire. Ce niveau s'étend d'une absence d'automatisation à une automatisation complète. **GoA 0** se réfère à une absence totale d'automatisation, où toutes les actions et décisions sont prises par un conducteur humain. Le **GoA 1**, quant à lui, évoque un niveau d'automatisation minimal où certaines fonctions, telles que l'assistance au freinage ou la régulation de la vitesse, peuvent être automatisées, mais l'essentiel des tâches de conduite reste sous le contrôle de l'opérateur. À l'échelon **GoA 2**, bien que l'automatisation englobe plusieurs aspects de la conduite, le conducteur demeure essentiel pour surveiller l'environnement et reprendre le contrôle face à des situations imprévues ou dangereuses. Au niveau **GoA 3**, la surveillance constante de l'environnement par l'homme n'est plus requise. C'est le système qui assume cette fonction en identifiant les aléas potentiels et en les communiquant à l'opérateur, lequel doit toutefois demeurer à bord pour intervenir en cas d'urgence. Enfin, le **GoA 4** représente une automatisation totale, où la présence humaine à bord n'est plus nécessaire et où la prise

de décision, tout comme la perception de l'environnement, est intégralement gérée par le système automatisé.

Il est intéressant de noter que même si cette classification peut s'appliquer à différents types de véhicules (trains, tramways, métros), les ressources nécessaires pour atteindre un même niveau peuvent varier considérablement selon le type de véhicule. Par exemple, les métros circulant à un niveau **GoA4** sont présents dans nos villes depuis plusieurs décennies. Pour gérer les aléas, la solution a été de créer un environnement entièrement contrôlé en isolant complètement le véhicule. En revanche, dans le cas des trains, il serait impossible de cloisonner l'ensemble du réseau ferroviaire, et les niveaux **GoA 3/4** nécessitent le développement de technologies de perception et de décision.

Il est important de faire la distinction entre un véhicule autonome et un véhicule automatique, bien que ces termes soient souvent utilisés de manière interchangeable. Un véhicule automatique peut être capable d'effectuer un trajet sans intervention humaine, mais uniquement dans un environnement structuré et prévisible. Un véhicule autonome, en revanche, peut fonctionner dans un environnement non structuré, ouvert et susceptible de comporter des événements imprévisibles. Dans le secteur ferroviaire, un système automatique peut être capable de piloter un métro sans conducteur dès lors que l'environnement est bien isolé, par exemple en ajoutant des portes palières aux quais ou en opérant dans une zone entièrement fermée et sécurisée. Cependant, un tel système ne pourrait pas fonctionner sur un train, les voies ferrées n'étant ni complètement sécurisées ni fermées. La différence entre ces deux termes est finalement liée à la complexité de l'environnement dans lequel le véhicule évolue et à la capacité ou non des méthodes d'automatisation classiques à y répondre [11,12].

Par conséquent, nous définissons un véhicule autonome comme étant capable de fonctionner indépendamment sans intervention humaine, tandis qu'un véhicule automatique réalise certaines tâches de conduite de manière automatique mais peut nécessiter une surveillance humaine. Cette distinction souligne la différence entre la capacité d'opérer seul et la réalisation automatique de tâches spécifiques.

2.2.1.2 Système Autonome

Un système autonome se réfère à un dispositif qui peut fonctionner indépendamment sans intervention humaine. Pour cela, il requiert un ensemble de modules intégrés qui lui permettent de percevoir son environnement et d'agir en conséquence en prenant des décisions basées sur les perceptions. Le système autonome peut être appliqué à divers contextes, y compris les véhicules autonomes, les robots, les drones, et dans notre cas, les trains de fret autonomes. Le projet TFA propose un système qui opère de manière similaire. Basé sur le principe de rétroaction fondée sur une boucle Perception/Décision/Action [8], il comporte des modules pour la perception de l'environnement, la prise de décisions basées sur ces perceptions, et l'action basée sur ces décisions. Cela

comprend des modules pour la localisation, la cartographie, la gestion de mission, la communication, et le contrôle. De plus, il intègre des interactions avec des acteurs externes et internes, tels que les gestionnaires de maintenance, les gestionnaires d'infrastructure, et les usagers du train (Train service voyageurs).

Dans le contexte du projet TFA, la perception joue un rôle crucial dans la détection d'obstacles potentiels et le monitoring de l'environnement. Les modules de perception sont responsables de la collecte et l'interprétation des données environnementales, qui sont ensuite utilisées pour informer le processus de prise de décision. En fin de compte, ce processus permet au train de fret de fonctionner de manière autonome.

Ainsi, le développement d'un système autonome pour le projet TFA repose sur la mise en œuvre d'une architecture fonctionnelle sophistiquée qui permet au train de fonctionner de manière autonome, tout en garantissant un niveau élevé de sécurité et d'efficacité.

La figure 4 montre l'architecture inspirée fortement du modèle utilisé dans le cadre du projet Train de Fret Autonome, avec l'intégration supplémentaire d'intervenants internes typiquement présents dans les véhicules transportant des passagers.

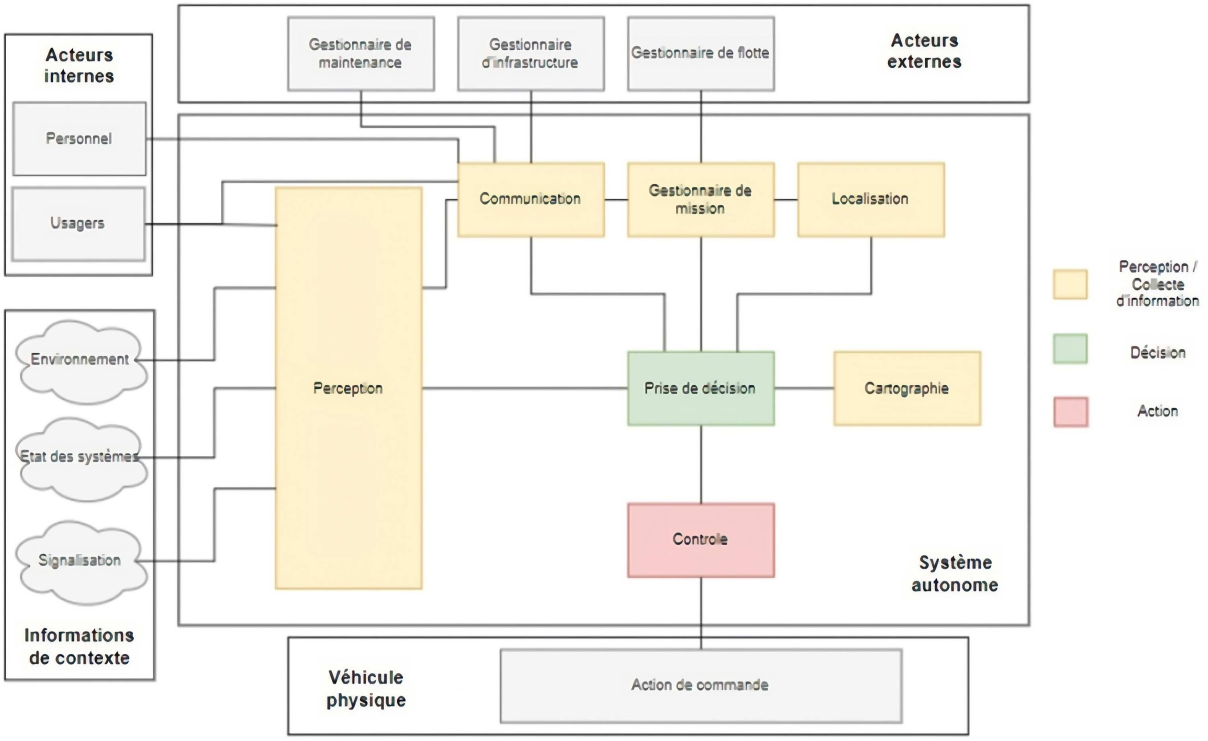

FIGURE 4: Schéma fonctionnel global pour le fonctionnement opérationnel d'un véhicule autonome.

Le Système Autonome se compose donc des éléments suivants :

— **Perception :** Il s'agit du système de capteurs qui facilite la détection, l'identification et la caractérisation des objets ou des phénomènes dans l'environnement physique du véhicule.

— **Communication :** Assure une liaison bidirectionnelle, permettant aux entités externes d'apporter des informations additionnelles et au véhicule de diffuser ses propres informations.
— **Gestion de mission :** Système qui établit et suit les objectifs de la mission, planifie et optimise le trajet.
— **Localisation :** Système GPS et autres capteurs permettant de déterminer la position exacte du véhicule et ses paramètres physiques.
— **Prise de décision :** Module qui évalue les données de la perception, la localisation et autres pour prendre des décisions opérationnelles.
— **Cartographie :** Système intégré qui fournit et met à jour les informations cartographiques nécessaires pour le fonctionnement du véhicule.
— **Contrôle :** Module qui exécute les directives du module de prise de décision, en respectant les différentes contraintes.
— **Environnement :** Ensemble des paramètres et conditions externes dans lesquels le véhicule évolue, comprenant la route, la météo, les autres véhicules, etc.
— **État des systèmes :** Moniteurs qui vérifient et rapportent l'état de santé des différents composants du véhicule.
— **Signalisation :** Il s'agit des informations externes liées à la circulation, incluant panneaux, affichage dynamique, marquages, feux de signalisation et télécommunications.
— **Gestionnaire de maintenance :** Système ou interface permettant d'interagir avec le véhicule pour les besoins de maintenance.
— **Gestionnaire d'infrastructure :** Système ou entité fournissant des informations sur l'infrastructure routière sur laquelle le véhicule évolue, telles que les conditions de route ou les travaux.
— **Gestionnaire de flotte :** Système de gestion centralisé lorsque le véhicule est intégré dans une flotte de véhicules.
— **Personnel :** Dans le cas de transports en commun, cela fait référence aux individus travaillant à bord, comme un contrôleur.
— **Usagers :** Il s'agit des passagers ou des individus transportés par le véhicule, avec lesquels le système autonome peut interagir ou pour lesquels il doit assurer la sécurité.

2.2.2 Briques du système autonome ferroviaire TFA

La plateforme du projet TFA se compose de plusieurs sous-systèmes interconnectés qui permettent l'automatisation complète de l'opération du train, chacun ayant une fonction spécifique pour assurer la sécurité, la fiabilité et l'efficacité des opérations. Dans ce contexte, notre thèse s'inscrit spécifiquement dans le sous-système GISE.

Voici une description détaillée de chaque sous-système :

— **NAV (Navigation)** : Ce sous-système est dédié à la géolocalisation du train, utilisant des capteurs et des systèmes satellites pour déterminer avec précision la position du train en temps réel.

— **SIAL (Signaling Interpretation and Abstraction Layer)** : Conçu pour détecter et interpréter la signalisation latérale, garantissant que le train est informé et répond correctement aux signaux le long de son trajet.

— **ATO (Automatic Train Control)** : L'ATO régule la vitesse du train en fonction de divers paramètres, tels que les conditions de la voie et le trafic, afin de garantir un fonctionnement optimal et sûr.

— **SUAL (Supervision Abstraction Layer)** : Fournit des informations relatives à la mission et au trajet, incluant les "polylines" des voies, qui représentent des tracés linéaires simplifiés du chemin que le train emprunte.

— **GISM (Global Information Surveillance Manager)** : Le GISM prend des décisions en se basant sur les incidents et les aléas rapportés par le sous-système de perception, principalement alimenté par le GISE.

— **PAL (Protection Abstraction Layer)** : Facilite la communication sécurisée entre les différents sous-systèmes pour garantir une coordination harmonieuse.

— **ATM (Autonomous Train Map)** : Offre une cartographie détaillée, indispensable pour une navigation précise et une planification de trajet optimale.

— **GISE (Global Information Surveillance Environment)** : Le GISE est chargé de détecter les aléas visuels en analysant l'environnement perçu par le système. Ce sous-système, essentiel pour la sécurité, est également l'objet principal de cette thèse, car il vise à identifier tout élément qui pourrait représenter un danger pour le train.

— **EDO (Environment Detection Obstacle)** : L'EDO se consacre à la détection d'obstacles grâce aux capteurs physiques montés sur le train, assurant une réaction prompte face à tout obstacle potentiel.

2.2.3 Contraintes et défis

L'élaboration du projet Train de Fret Autonome présente plusieurs contraintes et défis, lesquels déterminent l'approche requise pour la conception et l'implémentation des divers sous-systèmes.

Parmis les contraintes de ce projet en liaison avec cette thèse, on trouve :

— **Utilisation de capteurs RGB (caméra couleur)** : Pour ce travail de thèse, l'accent est mis sur l'exploitation des capteurs RGB, qui sont des caméras capturant des images en couleur. Bien que nous nous concentrons principalement sur ce type de capteur, d'autres types tels que les caméras infrarouges (IR) et le LiDAR

seront considérés pour les futures versions du prototype. Ces capteurs additionnels contribueront à l'amélioration des modèles de l'intelligence artificielle déjà en place.

— **Définition du monitoring de l'environnement** : Selon la SNCF, le monitoring de l'environnement englobe l'emprise ferroviaire, qui comprend la voie de circulation principale et l'ensemble des terrains adjacents sur lesquels les infrastructures ferroviaires sont situées. L'objectif majeur est de signaler toute anomalie (telle qu'une irrégularité ou une défaillance) ou aléa (comme un obstacle) divergeant du contexte habituel naturel.

Par ailleurs, le projet TFA est confronté à plusieurs défis significatifs, qui sont les suivants :

— **Manque de travaux de référence** : Peu de travaux s'intéressent à l'utilisation de l'IA dans la perception pour les trains autonomes. Cette carence rend difficile la comparaison de notre approche avec des travaux de référence existants.

— **Manque de données disponibles** : Il existe peu de datasets publiques disponibles spécifiquement pour la perception des trains autonomes. Cela rend difficile l'entraînement et la validation de nos modèles IA.

— **Absence de benchmarks** : Compte tenu du manque de travaux de référence, il est également difficile de mettre en place des benchmarks solides pour évaluer la performance de nos modèles.

Ces contraintes et défis influencent significativement l'approche de recherche et d'implémentation dans le cadre du projet TFA. Il est nécessaire de faire des efforts importants pour la collecte de données, l'exploration de nouvelles méthodes et la définition de critères d'évaluation pertinents.

2.3 Intelligence artificielle et apprentissage profond

L'intelligence artificielle, une notion originellement introduite par John McCarthy lors de la Conférence de Dartmouth en 1955 [13], a considérablement évolué depuis ses origines modestes. Elle se manifeste par la capacité d'une machine à simuler l'intelligence humaine, comprenant des branches telles que l'apprentissage automatique et l'apprentissage profond [14, 15]. Ces techniques constituent un élément essentiel dans le progrès de l'IA, bien que leur exploration exhaustive déborde du cadre de ce texte.

La vision par ordinateur (computer vision) [18], une spécialité majeure de l'IA, consiste à recueillir, analyser et comprendre des informations à partir d'images ou de séquences d'images. Dans ce contexte, l'IA a démontré une capacité sans égale à surpasser les performances humaines ; Certains réseaux de neurones convolutifs (CNN) [121], une catégorie de modèles d'apprentissage profond, ont la capacité d'identifier des objets dans des images avec une précision qui surpasse celle de l'homme [16].

La répercussion de l'IA est spécialement frappante dans le domaine des véhicules

autonomes. Selon [17] l'IA a permis l'élaboration de systèmes de perception, de décision et de contrôle qui assurent le fonctionnement sécurisé et efficace des véhicules autonomes. Les systèmes d'IA peuvent reconnaître et interpréter les situations routières, les signaux de circulation, les piétons et les autres véhicules, ce qui est indispensable pour une conduite sûre.

L'usage de l'IA dans les véhicules autonomes ne se limite pas à la perception et à la compréhension de l'environnement. L'IA est également exploitée pour prédire les comportements futurs des autres usagers de la route et prendre des décisions stratégiques concernant le positionnement du véhicule, le moment d'accélérer ou de freiner, et le moment de tourner [19].

L'IA a indéniablement provoqué une révolution dans le secteur automobile, surtout avec l'apparition des véhicules autonomes. Autrefois exclusivement dirigée par l'homme, la conduite est graduellement déléguée à des systèmes automatisés, un changement alimenté par l'IA. Cette transition a captivé l'intérêt de grands industriels comme Tesla, Google Waymo et Uber, qui investissent massivement dans la technologie de conduite autonome [78].

Il est également essentiel de souligner que l'IA ne se limite pas au domaine des véhicules autonomes ; elle est omniprésente dans de nombreux secteurs et disciplines. Par exemple, le traitement du langage naturel (NLP), une branche de l'IA qui facilite les interactions entre les ordinateurs et le langage humain, rendant possible des tâches telles que la traduction automatique, la reconnaissance vocale, et l'analyse des sentiments [79].

L'apprentissage par renforcement, un type d'IA comportementale, permet aux machines d'apprendre de manière autonome en interagissant avec leur environnement, ouvrant la voie à des applications dans la robotique où les systèmes doivent faire face à une multitude de situations imprévues [80].

La détection d'anomalies, une sous-branche de l'apprentissage automatique, est également largement utilisée dans plusieurs domaines. En se basant sur les techniques d'apprentissage automatique, elle permet d'identifier des événements qui ne correspondent pas à un schéma prévu ou typique. Par exemple, elle est couramment employée dans la maintenance prédictive pour anticiper les défaillances de machines dans l'industrie manufacturière. Elle est également utilisée en cybersécurité pour détecter les activités suspectes, en finance pour repérer les transactions frauduleuses, dans le secteur de la santé pour déceler des anomalies dans les images médicales [81].

Néanmoins, en dépit de ces progrès remarquables, l'IA est confrontée à divers défis. Les questions éthiques, les problèmes de sécurité et la nécessité d'une réglementation adéquate font partie des préoccupations majeures pour les chercheurs et les décideurs. Des enjeux éthiques, de responsabilité en cas d'accident, de sécurité informatique et l'adaptation de l'infrastructure routière sont parmi les problèmes importants à résoudre avant l'implémentation à grande échelle des véhicules autonomes [20,21].

En résumé, l'IA a bouleversé de nombreux aspects de notre société, depuis la vision par ordinateur jusqu'aux véhicules autonomes. Alors que nous continuons à explorer ses potentialités, il est essentiel de garder à l'esprit ses origines et les défis qui l'accompagnent, afin de mieux naviguer dans l'avenir de cette technologie prometteuse. La figure 5 montre quelques examples de branches de l'IA.

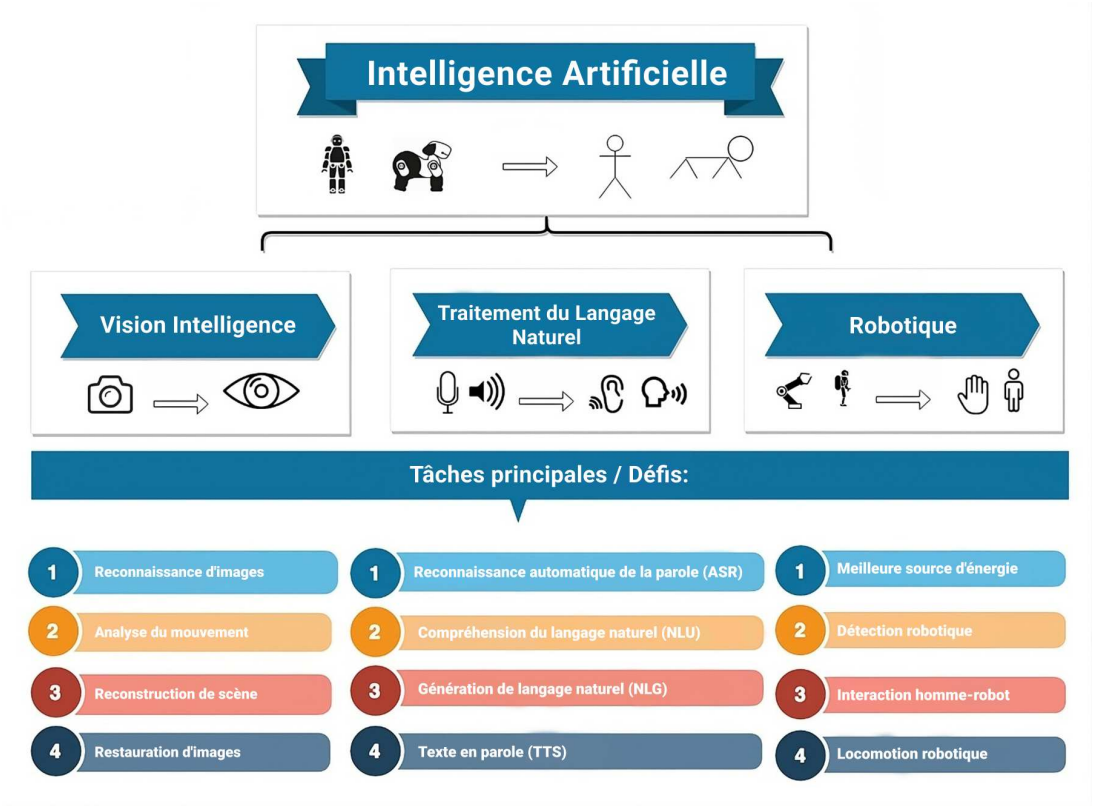

FIGURE 5: Les différents domaines et disciplines de l'IA (deep learning & machine learning) [23].

2.3.1 Apprentissage automatique et apprentissage profond

L'apprentissage automatique est une sous-catégorie de l'IA. L'IA englobe tout algorithme qui permet aux ordinateurs de reproduire le comportement humain comme on peut le voir dans la figure 6. Des exemples de ces algorithmes sont les systèmes basés sur des règles [22] ou des systèmes experts. Ces systèmes sont généralement utilisés pour le diagnostic médical où tous les résultats d'un problème (dans ces cas les maladies) sont énumérés avec toutes les entrées possibles : symptômes dans le cas du diagnostic médical [29]. Bien que ce principe soit utile dans certains domaines, il ne s'adapte pas à la taille du problème où l'expert humain doit transmettre une grande quantité d'informations à l'ordinateur. La capacité à apprendre sans être explicitement programmé était la suite des algorithmes de l'IA et est ce que nous appelons l'apprentissage automatique [30].

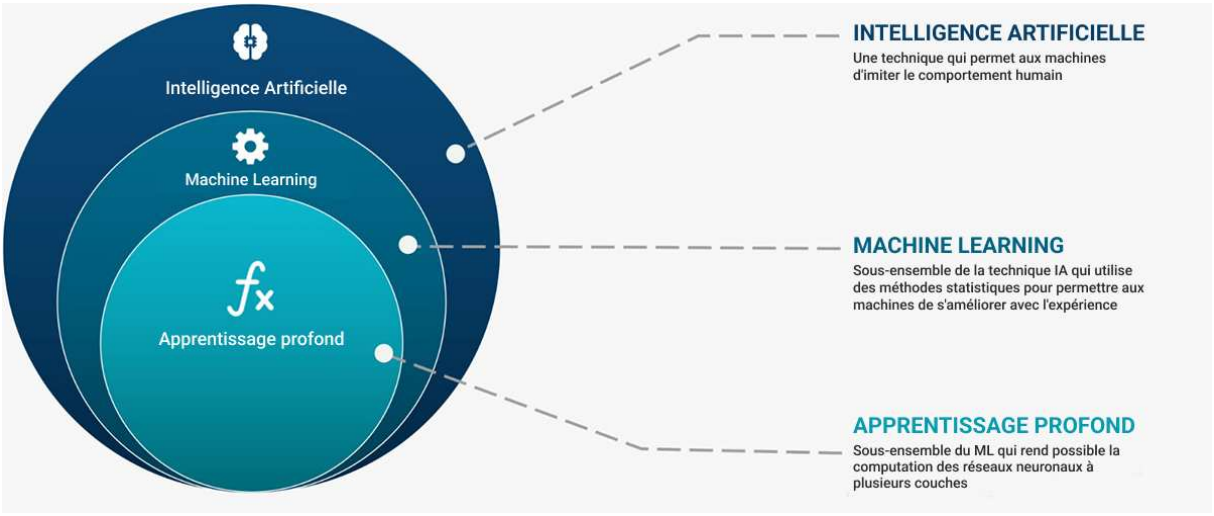

FIGURE 6: La différence entre l'intelligence artificielle, l'apprentissage automatique et l'apprentissage profond [24].

Dans une méthode traditionnelle, les développeurs donnent directement les instructions nécessaires pour accomplir une tâche spécifique, en détaillant chaque action que l'ordinateur doit exécuter. À l'opposé de cette approche, en apprentissage automatique, plutôt que de fournir des directives explicites, un modèle est entraîné pour aborder un problème déterminé lors de la phase d'entraînement. Cette phase est par la suite suivie de la phase d'inférence, durant laquelle le modèle, une fois entraîné, effectue des prédictions ou classifications sur des données nouvelles.

En fonction de la phase d'entraînement, de nombreux types d'algorithmes d'apprentissage automatique se présentent [31] :

- **Apprentissage supervisé :** Il s'agit d'une méthode où un modèle apprend à partir de données étiquetées. C'est-à-dire que chaque exemple de l'ensemble de données d'entraînement est associé à une étiquette ou une sortie. Pendant l'entraînement, l'algorithme cherche à minimiser les erreurs entre les prédictions du modèle et les véritables sorties.
- **Apprentissage non supervisé :** Il s'agit d'une méthode où un modèle apprend à partir de données non étiquetées. L'objectif de ces modèles est de comprendre les structures inhérentes aux données. Les algorithmes d'apprentissage non supervisé peuvent regrouper les données en catégories similaires, un processus connu sous le nom de "*clustering*".
- **Apprentissage semi-supervisé :** Il s'agit d'une méthode qui combine des aspects de l'apprentissage supervisé et non supervisé. Dans ce cas, le modèle est entraîné sur un mélange de données étiquetées et non étiquetées. Ces types d'algorithmes tentent d'apprendre la structure des données à partir de l'ensemble non étiqueté, tout en essayant de prédire les étiquettes de l'ensemble étiqueté.
- **Apprentissage par renforcement :** Il s'agit d'une méthode où un agent ap-

prend à se comporter dans un environnement en effectuant certaines actions et en recevant des récompenses/pénalités. Le but de l'agent est de trouver la politique qui maximise la récompense totale à long terme.

— **Apprentissage auto-supervisé :** L'apprentissage auto-supervisé est une méthode d'apprentissage automatique où le modèle est formé en utilisant des pseudo-étiquettes générées à partir des données d'entrée elles-mêmes, plutôt que des étiquettes fournies par des annotateurs humains. Ce paradigme tire parti de l'information inhérente aux données pour fournir un contexte qui aide le modèle à initialiser ses paramètres et à apprendre des structures de données complexes. Cela se fait généralement par le biais de tâches auxiliaires ou de prétextes qui préparent le modèle pour la tâche d'apprentissage réelle, qu'elle soit supervisée ou non supervisée. L'apprentissage auto-supervisé a montré son efficacité en produisant des résultats prometteurs dans divers domaines tels que le traitement audio et la reconnaissance vocale [32, 33].

Dans l'apprentissage profond, les modèles sont généralement caractérisés par leur profondeur, c'est-à-dire le nombre de couches présentes. Ces structures sont souvent désignées sous le terme de réseaux de neurones [34]. Chaque couche du réseau apprend des caractéristiques de plus en plus complexes à partir des données. Ainsi, un réseau profond est capable d'apprendre des représentations complexes des données.

En revanche, dans l'apprentissage automatique, les modèles peuvent être basés sur des méthodes statistiques directes sans utiliser plusieurs couches. Par exemple, un modèle de régression linéaire ne comporte qu'une seule couche, tandis que les arbres de décision et les forêts aléatoires peuvent être considérés comme n'ayant pas de couches, car ils segmentent simplement les données selon des critères spécifiques.

La principale différence entre l'apprentissage automatique et l'apprentissage profond réside dans la manière dont ils traitent leurs problèmes. Alors que l'apprentissage automatique nécessite souvent une intervention humaine pour identifier les caractéristiques appropriées à extraire des données pour résoudre efficacement un problème, l'apprentissage profond est capable de déterminer automatiquement quelles caractéristiques sont utiles pour la classification, ce qui rend le processus d'apprentissage beaucoup plus autonome et efficace [35].

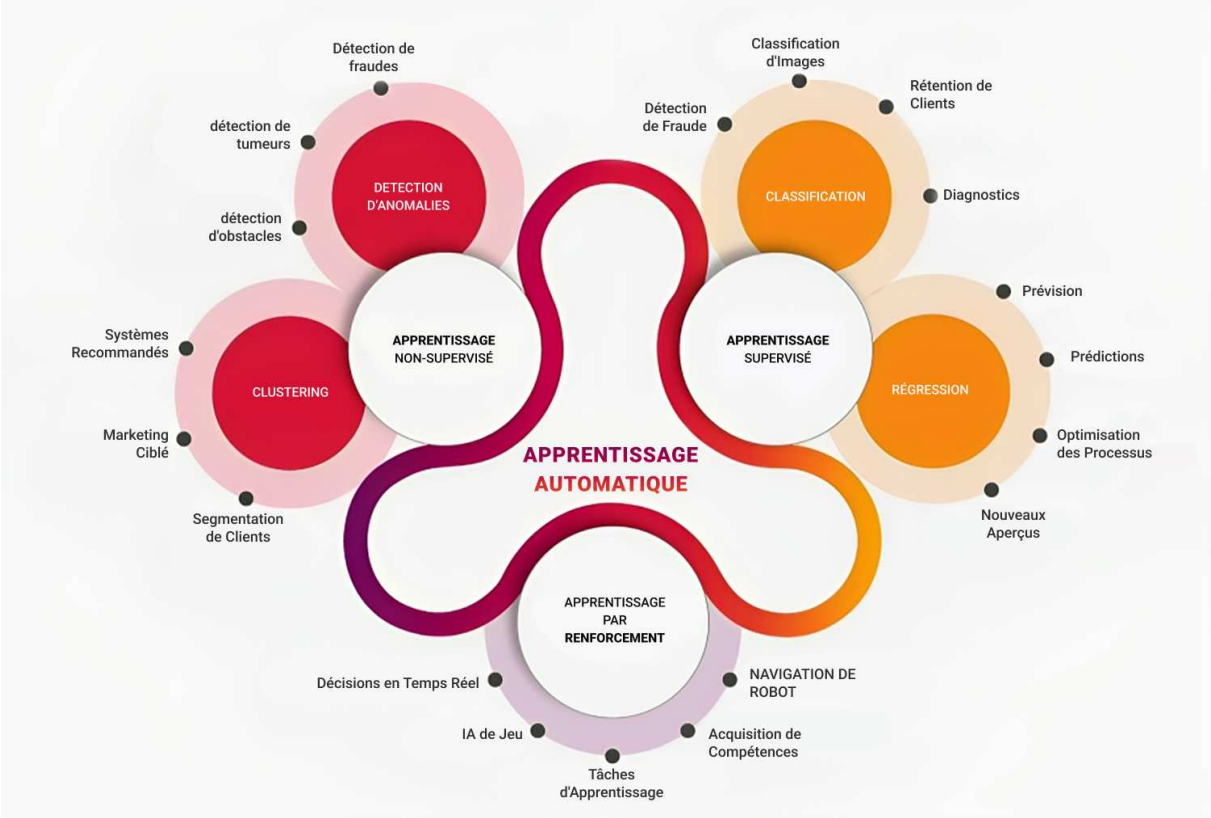

FIGURE 7: La classification des algorithmes d'apprentissage automatique et des sous-catégories de chaque classe avec certains exemples de cas d'utilisation données aux niveau de chaque sous-catégorie [25].

La Figure 7 montre les types les plus courants de cas d'utilisation de l'apprentissage automatique, à savoir, l'apprentissage supervisé, non supervisé et par renforcement. A l'exception de l'apprentissage par renforcement où aucun ensemble de données n'est présent, la précision de la méthode dépend fortement de la qualité des échantillons dans l'ensemble de données.

2.3.1.1 Fonctionnement

Les Réseaux de Neurones Artificiels (ANN) [34] font parti des algorithmes d'apprentissage profond qui imite le cerveau humain en répliquant, dans une certaine mesure, son architecture. Ils sont de loin les algorithmes d'apprentissage profond les plus utilisés. Pour cette raison, nous consacrons la section 2.3.1.2 à la présentation de cette catégorie d'algorithmes. La section 2.3.1.3 détaille une variante des ANNs qui traite principalement des images : les réseaux de neurones convolutionnels (CNNs) [121].

La capacité à gérer de grandes quantités de données a principalement été obtenue grâce à l'utilisation de l'algorithme de rétropropagation [36]. Cet algorithme est utilisé lors de la phase d'entraînement de presque tous les réseaux neuronaux. Il consiste à propager

les entrées à travers le réseau, les résultats seraient probablement erronés, l'erreur est quantifiée et sa valeur est propagée en arrière pour mettre à jour les paramètres du réseau.

Pour comprendre ce processus d'apprentissage, considérons une fonction $y = f(x) = ax + b$ qui prend une entrée x et produit une sortie y. En apprentissage automatique, le problème est de trouver les paramètres a et b afin que cette fonction génère une valeur désirée y pour une entrée x donnée. La relation entre x et y est représentée par une liste $\mathcal{L}$ de n paires $(x_i, y_i^{'})_{i=1->n}$ qui correspondent à la sortie désirée $y_i^{'}$ de la fonction f lorsque l'entrée est x_i.

Le processus d'apprentissage utilise la liste $\mathcal{L}$ pour trouver la meilleure paire de paramètres (a, b) afin d'augmenter la qualité de sortie de la fonction f. La qualité est mesurée en utilisant une fonction d'erreur $\mathcal{E}_i(a, b)$. Cette fonction prend la valeur actuelle de la paire de paramètres (a, b) et mesure la distance entre la sortie actuelle y_i et la sortie désirée $y_i^{'}$ de f pour une entrée donnée x_i de la liste $\mathcal{L}$.

$$\mathcal{E}_i(a, b) = |y_i - y_i^{'}| \tag{2.1}$$

Dans l'équation 2.1, nous estimons l'erreur, par choix, comme la valeur absolue de la différence entre les sorties y et les sorties désirées $y^{'}$ pour chaque valeur de $x \in \mathcal{L}$. Puisque $y_i = f(x_i) = ax_i + b$, l'équation 2.1 peut être écrite comme suit :

$$\mathcal{E}_i(a, b) = |ax_i + b - y_i^{'}| \tag{2.2}$$

Pour une paire de paramètres donnée (a, b), l'erreur totale $\mathcal{E}(a, b)$ serait calculée comme la somme sur toutes les paires $(x_i, y_i^{'}) \in \mathcal{L}$ comme suit :

$$\mathcal{E}(a, b) = \sum_{i=1}^{n} |ax_i + b - y_i^{'}| \tag{2.3}$$

Cette erreur peut être visualisée dans une grille tridimensionnelle en faisant varier a et b et en calculant l'erreur $\mathcal{E}$ chaque fois. Ensuite, nous considérons le problème comme indiqué précédemment, où $\mathcal{L}$ est la liste des paires du tableau 2.1.

Inputs (x)	1	-2	4	3	2	-1	-1
Labels (y)	2	5	17	10	5	2	2

TABLE 2.1: Table des valeurs d'entrée et de sortie désirées representant un exemple d'un dataset labelisé simplifié.

La figure 8 montre la distribution de la fonction d'erreur de l'équation 2.3. Ces valeurs représentent l'erreur sur la sortie de la fonction $f(x) = ax + b$ pour une paire donnée (a, b) et les sorties de la liste $\mathcal{L}$. Nous pouvons trouver visuellement le minimum de cette erreur dans les intervalles $(a, b) \in ([-6, 6], [-2, 10])$, chose qui représente l'objectif de cette phase. Ce minimum peut être local, cependant, il a été remarqué que la présence

d'un minimum local dans la fonction d'erreur ne semble pas être un problème majeur en pratique [37].

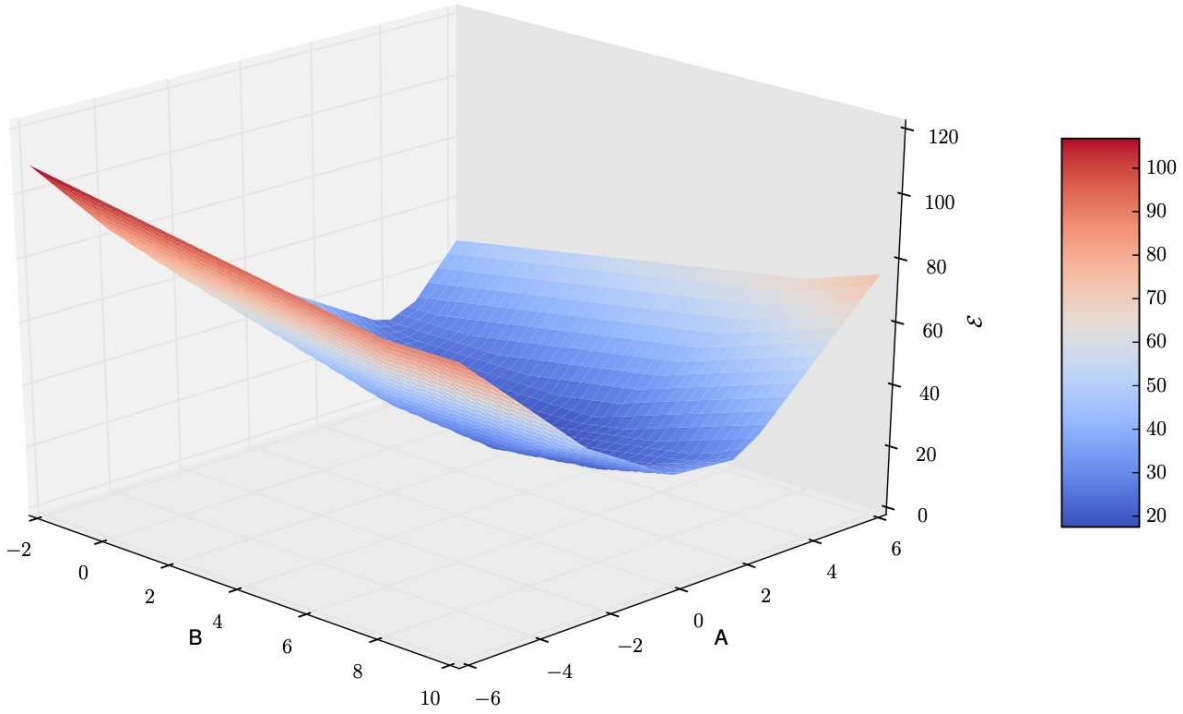

FIGURE 8: La distribution des erreurs de la fonction $\mathcal{E}$.

Cependant, cette visualisation pourrait ne pas être pratique. Elle est fortement dépendante du choix des intervalles et, pour une fonction complexe, il serait difficile de reconnaître un motif qui aide à localiser les minima. Par conséquent, afin d'augmenter la qualité de la sortie, nous devons trouver analytiquement une valeur pour (a, b) qui minimise cette fonction d'erreur. L'énoncé du problème est montré dans l'Équation 2.4.

$$\min_{(a,b)\in\mathbb{R}^2, i=1\to n} \mathcal{E}_i(a,b) = |ax_i + b - y_i^{'}| \tag{2.4}$$

Ce problème peut être résolu numériquement. Nous initialisons d'abord a et b de manière aléatoire à (a_0, b_0) et ensuite, nous mettons à jour leurs valeurs de manière itérative tout en réduisant l'ampleur de la fonction d'erreur. La mise à jour des poids (a et b) ne peut pas être arbitraire. Des méthodes numériques pour trouver le minimum pourraient être utilisées comme la méthode de Newton [38]. Concernant l'apprentissage automatique, l'optimiseur le plus courant est la descente de gradient (GD) [39] en raison de sa simplicité et de son efficacité. Semblable à la méthode de Newton [38], il s'appuie sur la pente, ou le gradient, de la fonction à un point donné. Si à un point a_0, la pente de $\mathcal{E}$ est négative, alors la fonction est en hausse. Puisque la fonction d'erreur prend deux arguments, la pente est calculée comme la dérivée partielle par rapport à chaque

paramètre. Les procédures de mise à jour sont montrées dans l'Équation 2.5.

$$\begin{aligned} a_{k+1} &= a_k - \frac{\partial E(a_k, b_k)}{\partial a}, \\ b_{k+1} &= b_k - \frac{\partial E(a_k, b_k)}{\partial b}. \end{aligned} \tag{2.5}$$

Par analogie avec l'apprentissage automatique, la liste $\mathcal{L}$ de notre exemple simplifié représente le dataset. La phase d'entraînement est supervisée puisque le dataset contient les entrées avec les sorties correspondantes, ou les étiquettes dans le cas d'une tâche de classification d'images par exemple. Les paramètres a et b représentent les poids. Un algorithme plus compliqué nécessiterait plus de deux paramètres pour expliquer des relations complexes, d'où le besoin d'un vecteur de poids et un f plus complexe. Dans un tel cas, la visualisation ne peut plus être envisagée car elle aboutit à un espace de dimension très large.

L'algorithme initial de descente de gradient pour l'apprentissage a été adapté et amélioré dans de nombreux travaux récents. Par exemple, dans l'Équation 2.5, un taux d'apprentissage α est introduit pour ralentir la descente vers le minimum. Cet ajout s'est avéré être primordial pour que l'algorithme converge vers le minimum. Une variation populaire est l'algorithme de Descente de Gradient Stochastique (SGD) [39].

2.3.1.2 Architecture des réseaux de neurones artificiels (ANN)

Les réseaux de neurones artificiels (ANN) [34] s'inspirent des neurones biologiques. Ils associent un ensemble d'entrées à des sorties via des éléments de calcul, les neurones. Ces neurones sont répartis en trois couches : d'entrée, cachée et de sortie, comme illustré à la Figure 9. Une analogie avec les neurones humains est présentée à la Figure 10. Le nombre de neurones dans les couches d'entrée et de sortie dépend du problème, tandis que celui de la couche cachée est une variable de conception. Plus de neurones ou de couches permettent des représentations plus complexes mais n'assurent pas une meilleure précision. Un neurone opère comme suit : chaque entrée x_i du vecteur X est multipliée par un poids w_i. Puis une fonction d'activation A est appliquée sur cette somme, décrite par l'équation :

$$O = A\left(\sum_{i=0}^{n} x_i \cdot w_i\right) \tag{2.6}$$

La fonction d'activation régule la sortie des neurones. Elle doit être dérivable pour permettre la mise à jour des poids en fonction des dérivées partielles. La sigmoïde, dont la sortie est entre [0, 1], est un choix possible. Une autre fonction courante est la ReLU [238], définie par :

$$\mathrm{ReLU}(x) = \begin{cases} x & \text{si } x > 0 \\ 0 & \text{sinon} \end{cases}$$

Les ANNs, malgré leur efficacité pour la classification et la régression, ont une limite : ils ne traitent pas bien les données structurées. Par exemple, avec l'Équation 2.6, intervertir deux neurones d'entrée ne change pas la sortie. C'est une contrainte pour des données comme les images où l'ordre des pixels importe. Cela a poussé à développer d'autres architectures valorisant cette structure.

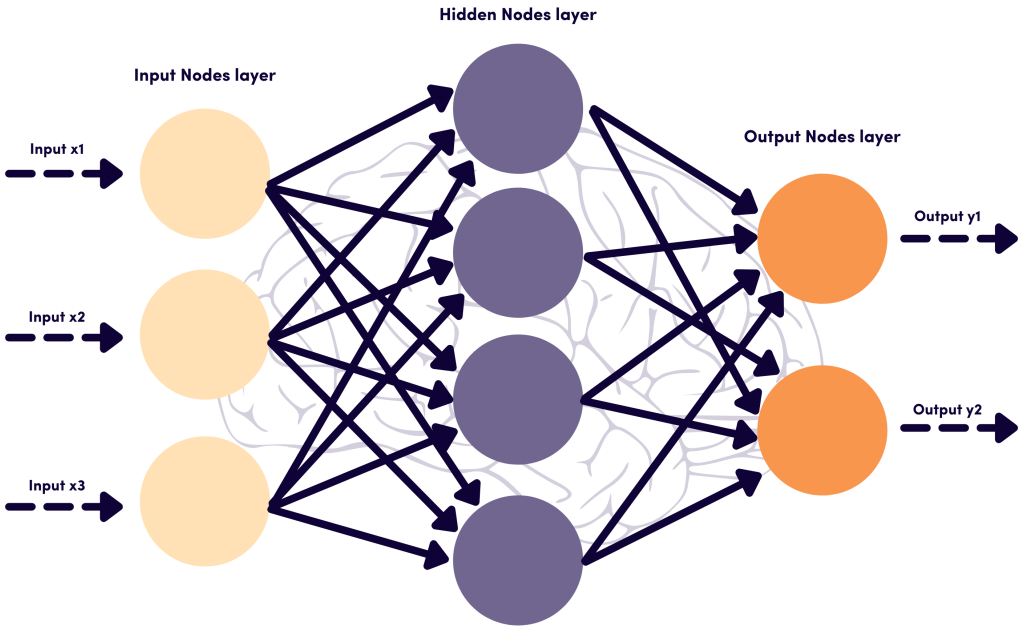

FIGURE 9: Architecture d'un réseau de neurones artificiel [26].

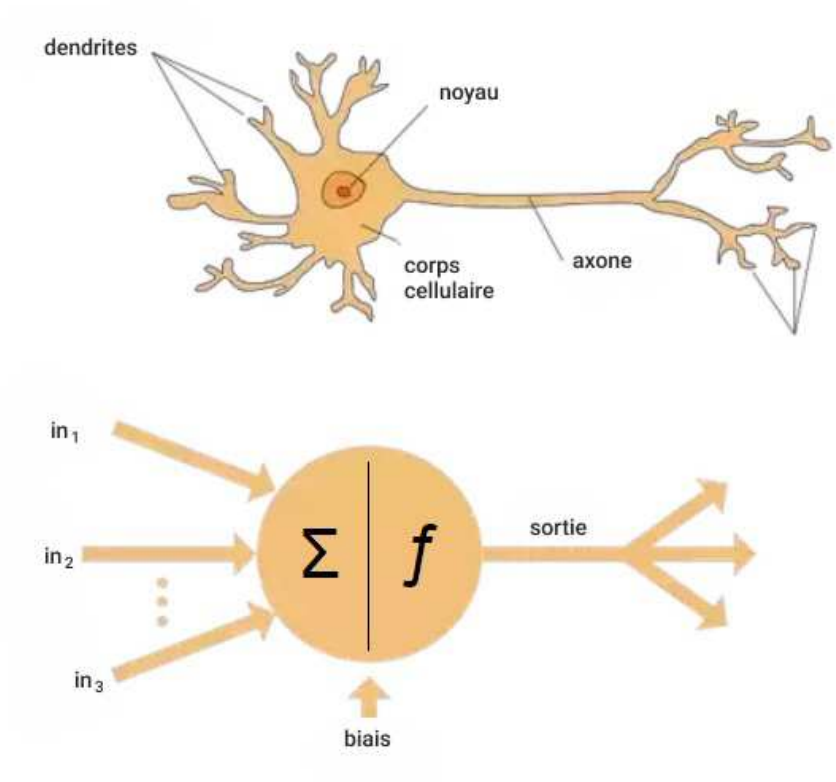

FIGURE 10: Analogie entre les neurones artificiels et un véritable neurone du cerveau humain [27].

2.3.1.3 Architecture des réseaux de neurones convolutionnels (CNN)

Les images, par nature, portent des structures intriquées où l'essence n'est pas dans un pixel unique, mais dans sa proximité. Grâce aux convolutions [42], il est possible d'extraire des caractéristiques de ces images sans recourir à des architectures vastes comme les ANN. Les CNNs [121] intègrent ces convolutions comme extracteurs de caractéristiques auto-formés, les faisant souvent reconnaître comme DNN [43]. L'apprentissage profond, axé sur les DNNs, s'inscrit dans l'apprentissage automatique, comme illustré à la Figure 6, avec sa particularité d'extraction automatique des caractéristiques. Les couches convolutives des CNNs traitent des cartes de caractéristiques F aux dimensions $W \times H \times C$ en utilisant des fenêtres $K \times K$. Contrairement aux ANN, elles partagent des poids, offrant différentes représentations de l'image. Ces représentations évoquent des algorithmes de filtrage comme Sobel [203]. Enchainant ces couches, on dégage des caractéristiques fines, ensuite classifiées via un ANN. Suite à cette extraction, des étapes d'agrégation, basées sur des techniques de regroupement telles que le maximum et le minimum [44], compressent les sorties. La déconvolution [45], processus inverse de la convolution, sert principalement à visualiser et redimensionner les sorties des CNNs. Les figures 11 et 12 illustrent respectivement cette opération et l'architecture VGG-16 [54].

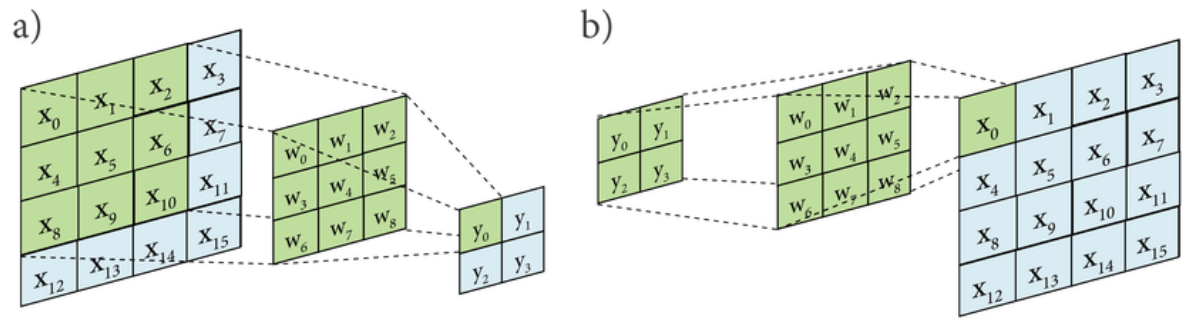

FIGURE 11: Exemple d'une opération de convolution a) et une opération de déconvolution b) [28].

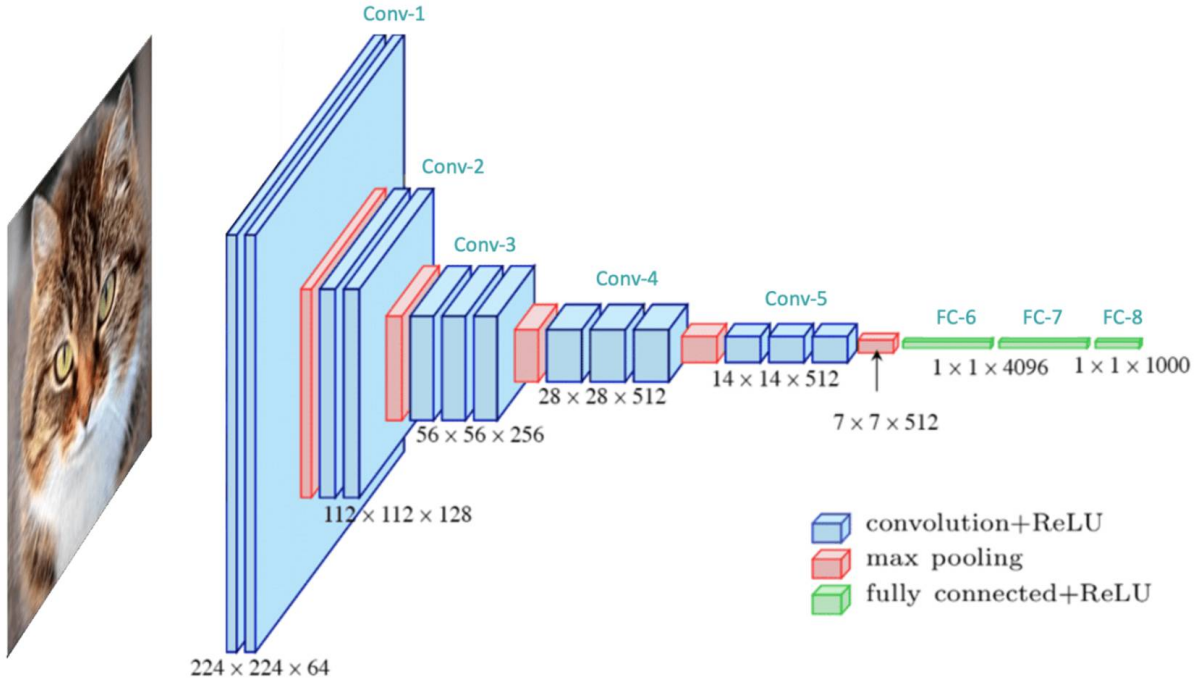

FIGURE 12: Exemple d'une architecture d'un CNN pour classifier les images intitulée VGG-16 [54].

2.3.1.4 Exemples de datasets de vision

La profusion des datasets disponibles a alimenté le succès des méthodes d'apprentissage automatique et profond. Les données de haute qualité que fournissent ces datasets assurent un entraînement efficace des modèles, conduisant à des modèles hautement précis. Voici quelques exemples des datasets de vision bien connus, chacun ayant des caractéristiques et des utilisations distinctes :

— MNIST est souvent l'un des premiers datasets utilisés pour l'entraînement des modèles d'apprentissage automatique. Il se compose de 60k images en niveaux de gris, chacune représentant un chiffre écrit à la main de résolution 28x28 pixels. Le dataset est généralement divisé en un ensemble d'apprentissage de 50k images et

un ensemble de test avec les 10k images restantes. L'objectif d'un modèle entraîné sur ce dataset est de classer correctement une image en entrée dans l'une des dix classes, chacune représentant un chiffre de 0 à 9 [46].

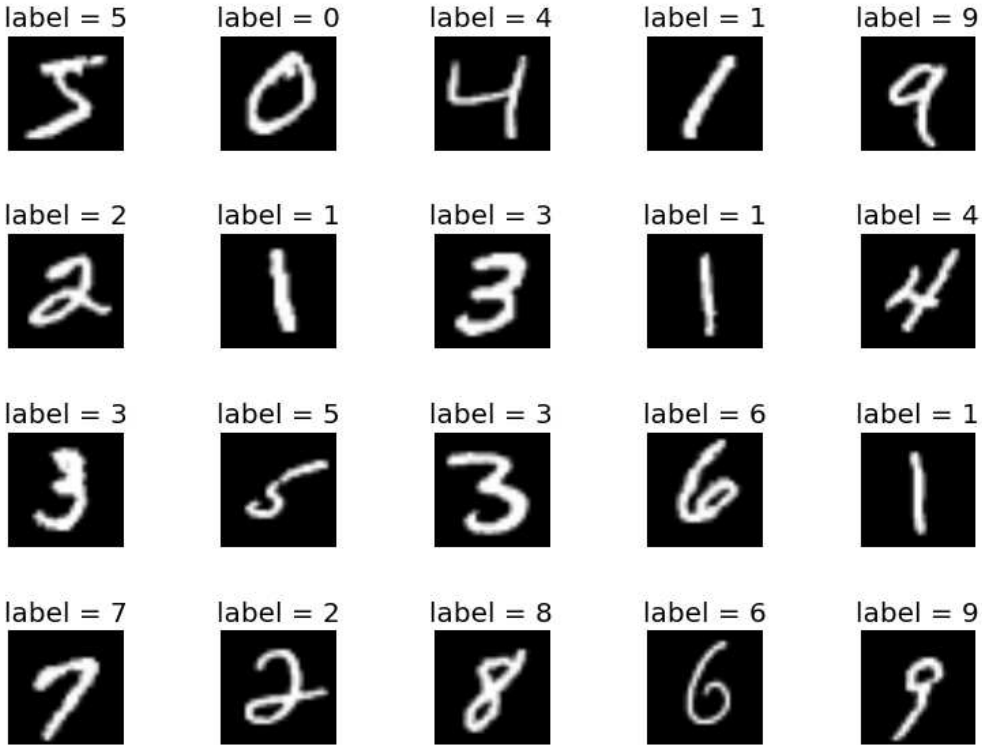

FIGURE 13: Aperçu du dataset MNIST [46].

— CIFAR-10 est un dataset constitué de 60k images en couleur de 32x32 réparties en 10 classes, avec 6k images par classe. Il comprend 50k images d'entraînement et 10k images de test. L'ensemble de données est divisé en cinq subsets d'entraînement et un subset de test, chaque subset contenant 10k images. Le subset de test contient exactement 1k images sélectionnées aléatoirement de chaque classe. Les subsets d'entraînement contiennent les images restantes dans un ordre aléatoire, mais certains subsets peuvent contenir plus d'images d'une classe que d'une autre. Au total, les subsets d'entraînement contiennent exactement 5k images de chaque classe.
CIFAR-100, quant à lui, est similaire à CIFAR-10, à l'exception qu'il comprend 100 classes contenant chacune 600 images. Il y a 500 images d'entraînement et 100 images de test par classe. [47].

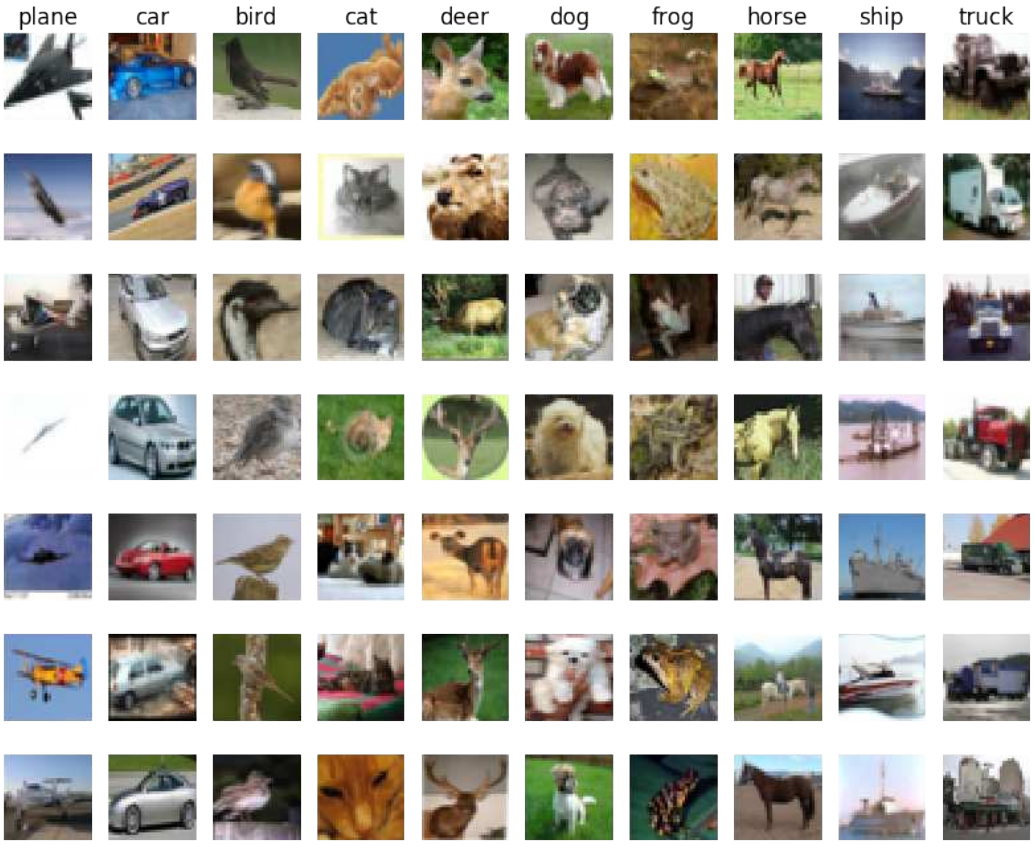

FIGURE 14: Aperçu du dataset CIFAR [47].

— ImageNet est un dataset organisé selon la hiérarchie de WordNet, représentant chaque nœud par des milliers d'images. Il comprend des données d'entraînement, de validation et des étiquettes, totalisant 1,2 million d'images réparties dans 1k catégories. Les données de validation et de test, distinctes de l'ensemble d'entraînement, comprennent 150k photos provenant de diverses sources. Cet ensemble de données a joué un rôle clé dans l'avancement de la recherche en vision par ordinateur et en apprentissage profond. [48].

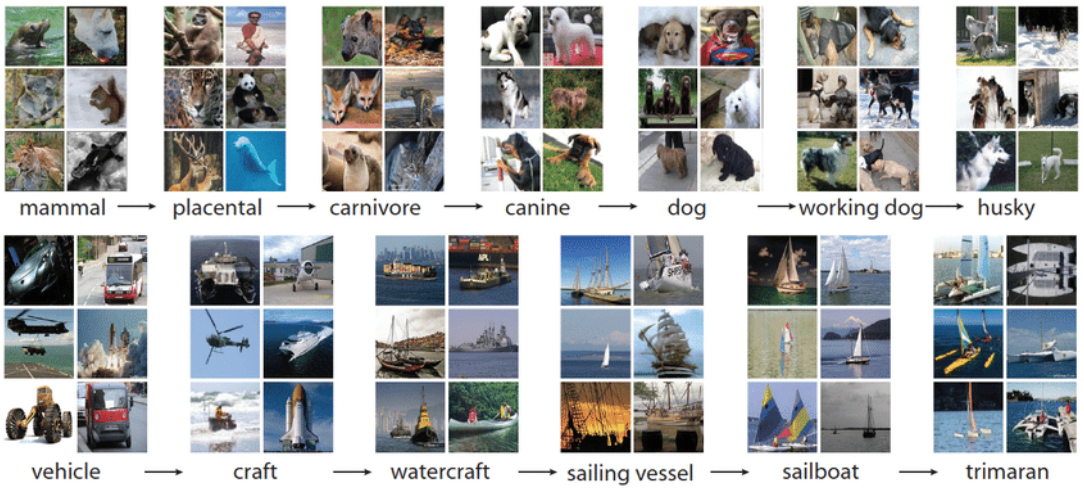

FIGURE 15: Aperçu du dataset ImageNet [48].

— COCO (Common Objects in Context), est un dataset d'images à grande échelle contenant 328k images d'objets. Il est destiné à la détection d'objets, à la segmentation d'instances et au légendage d'images. Ses annotations permettent d'entraîner des modèles d'apprentissage profond pour reconnaître, étiqueter et décrire des objets [128].

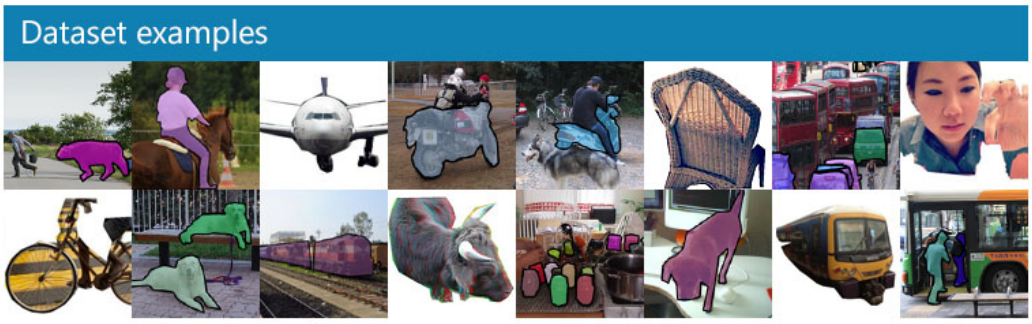

FIGURE 16: Aperçu du dataset COCO [128].

— LFW (Labeled Faces in the Wild) est un dataset d'images comprenant des photographies de visages, recueillies spécifiquement pour l'étude de la reconnaissance de visages. Il contient plus de 13k images de visages. Chaque visage est étiqueté par le nom de la personne dans l'image. 1680 personnes apparaissent distinctement dans deux photos ou plus [49].



FIGURE 17: Aperçu du dataset LfW [49].

— L'ensemble de données ADE20K est conçu pour la segmentation sémantique et comprend plus de 20k images annotées au niveau des pixels. Il couvre 150 catégories, avec un total de 27k images réparties sur 365 scènes. L'ensemble contient également des annotations détaillées pour plus de 700k objets uniques et près de 200k parties d'objets [50].



FIGURE 18: Aperçu du dataset ADE20K [50].

— KITTI est un ensemble de données construit à partir d'une plate-forme de conduite autonome, couvrant diverses tâches de vision comme la stéréo-vision, le flux optique, l'odométrie visuelle, etc. Cet ensemble comprend une partie dédiée à la détection d'objets, avec des images monochromes et des boîtes englobantes. Il contient 7481 images d'entraînement annotées avec des boîtes englobantes 3D, réparties entre les datasets de test (711 images), d'entraînement (6 347 images) et de validation (423 images) [129].

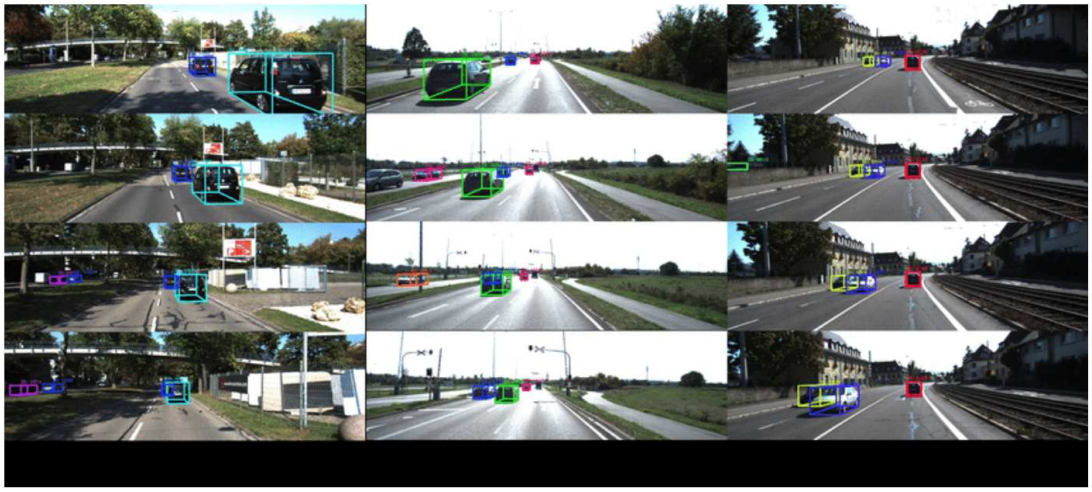

FIGURE 19: Aperçu du dataset KITTI [129].

Le dataset RailSem19 [218] se distingue dans la littérature de la vision par ordinateur. Unique en son genre, il est le seul centré sur le domaine ferroviaire et dédié à la segmentation de scènes ferroviaires. Doté d'une gamme riche d'images, il offre une immersion détaillée dans les environnements et composantes liés aux voies ferrées. RailSem19 est ainsi devenu une ressource précieuse pour les chercheurs et experts souhaitant affiner des algorithmes adaptés à cette niche du secteur des transports. Des informations complémentaires sont disponibles dans la section 4.3.



FIGURE 20: Aperçu du dataset RailSem19 [218].

Les datasets de vision sont le socle fondamental sur lequel reposent les avancées majeures en vision par ordinateur. Ils ne sont pas simplement de vastes collections d'images, mais des ressources méticuleusement organisées qui reflètent une gamme variée de scénarios et de contextes du monde réel. Cette diversité est essentielle, car elle expose les modèles à des situations multiples, permettant ainsi une meilleure généralisation. De plus, la disponibilité de classes variées et de dimensions d'images hétérogènes assure que les modèles peuvent être entraînés pour des nuances subtiles et des distinctions complexes. En essence, sans ces datasets riches et variés, la résolution efficace des défis de la vision par ordinateur serait inimaginable.

2.3.1.5 Exemples d'architectures des CNNs

— **LeNet-5** : La première architecture réussie d'un CNN a été introduite avec LeNet-5 [51], simplement appelé LeNet. LeNet a été conçu pour la reconnaissance de caractères isolés. LeNet se compose de deux parties : un extracteur de caractéristiques et un classificateur. L'extraction des caractéristiques est effectuée par deux couches convolutives avec de petits filtres de taille 5x5, suivies d'activations ReLU [238]. Chaque couche convolutive est suivie d'une opération de max pooling pour réduire la dimensionnalité. Le classificateur est composé de trois couches entièrement connectées, terminées par une activation Softmax [52]. LeNet-5 a atteint une précision de 99,2 % sur le dataset MNIST [46].

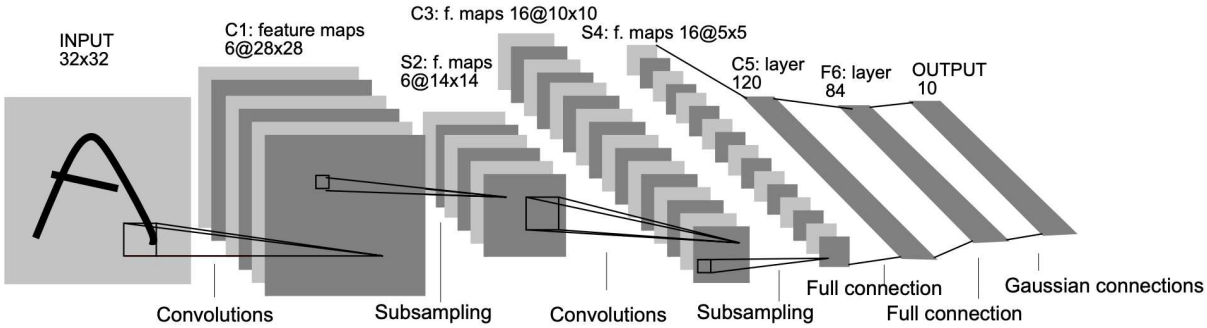

FIGURE 21: Aperçu de l'architecture LeNet [51].

— **AlexNet** : Le défi de reconnaissance visuelle à grande échelle ImageNet (ILSVRC) 2012 a marqué le premier succès majeur de l'apprentissage automatique. AlexNet [53], une architecture CNN, a remporté le défi de classification d'images en surpassant l'exactitude humaine. Cette étape a été franchie grâce à l'utilisation de 5 couches convolutionnelles séparées par des couches de maxpooling et de normalisation. La classification est réalisée à l'aide de 3 couches entièrement connectées avec une activation Softmax [52] à la fin.

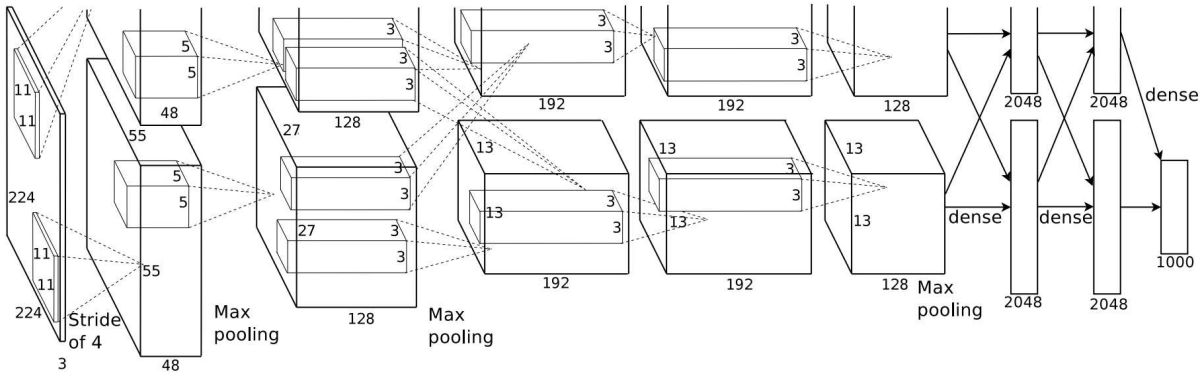

FIGURE 22: Aperçu de l'architecture AlexNet [53].

— **VGG-16** : VGG-16 [54] est une amélioration de AlexNet. Avec plus de couches et une architecture similaire, la principale différence réside dans la taille des noyaux. Les couches convolutionnelles utilisent des noyaux de 3x3 à tous les niveaux pour une réutilisation matérielle plus facile. VGG-16 est particulièrement utile pour la classification d'images.

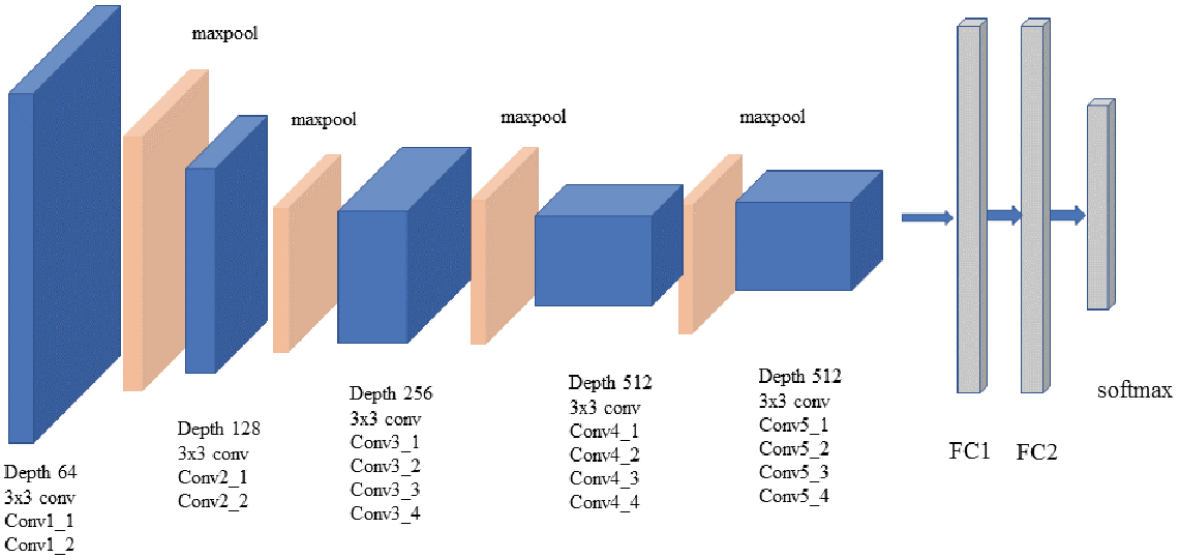

FIGURE 23: Aperçu de l'architecture VGG [54].

— **GoogleNet** : GoogleNet [55], vainqueur de l'ILSVRC 2014, est un réseau de 22 couches. Il introduit les couches d'incorporation qui sont la sortie concaténée de trois couches convolutionnelles parallèles avec différentes tailles de filtre. La couche d'incorporation utilise de petits filtres pour une résolution fine et de grands filtres pour un champ réceptif plus large.

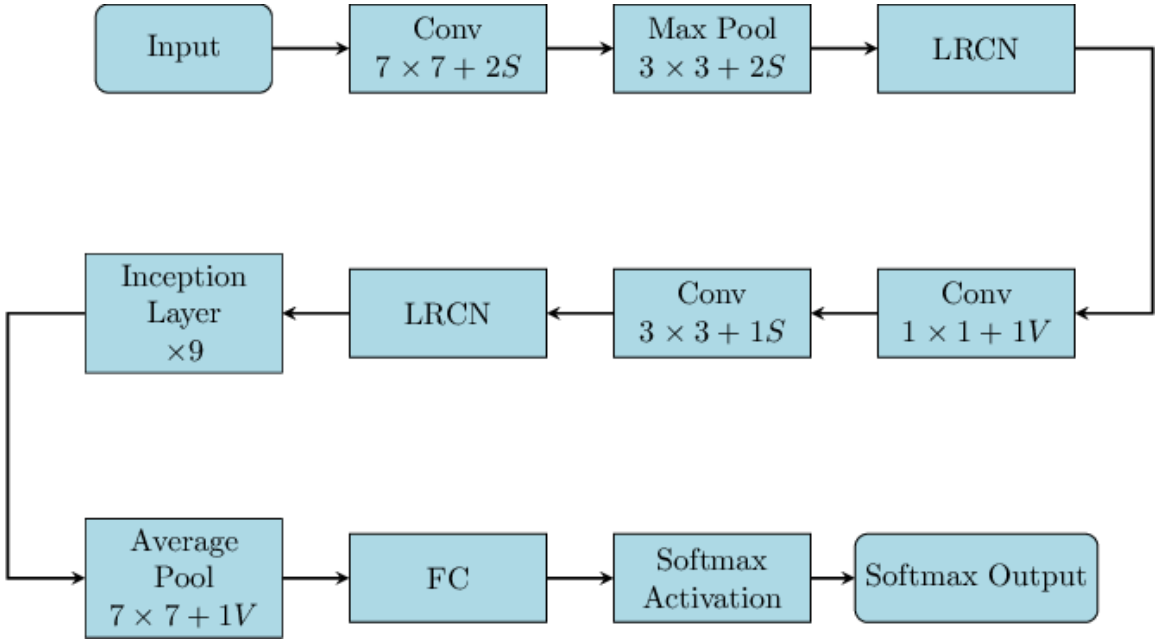

FIGURE 24: Un diagramme simplifié de l'architecture GoogleNet [56].

— **ResNet** : ResNet [114], ou *Residual Network*, est une architecture CNN qui utilise des connexions résiduelles pour éviter le problème de la disparition du gradient dans les réseaux profonds. ResNet a été le vainqueur de l'ILSVRC 2015 et est largement utilisé pour la classification d'images.

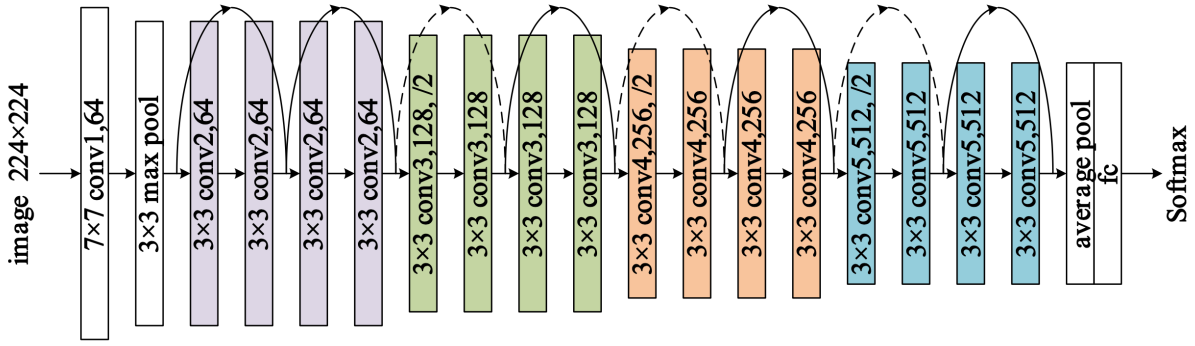

FIGURE 25: Aperçu de l'architecture ResNet, spécifiquement ResNet-18 [115].

— **DETR** : DETR [57], ou *DEtection TRansformer*, est une architecture récente qui utilise les Transformeurs pour la détection d'objets, une tâche généralement effectuée par des architectures basées sur les CNN.

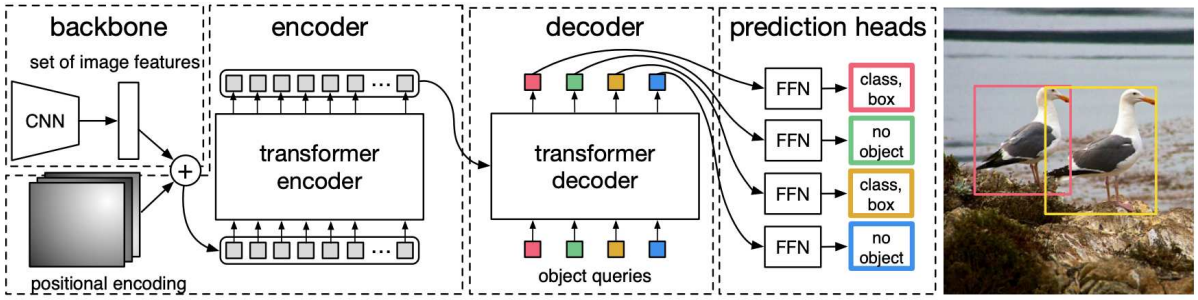

FIGURE 26: Aperçu de l'architecture DETR [57].

— **YOLO** : YOLO [58], ou *You Only Look Once*, est une architecture populaire pour la détection d'objets en temps réel. YOLO réalise la détection d'objets en une seule passe, ce qui le rend particulièrement rapide.

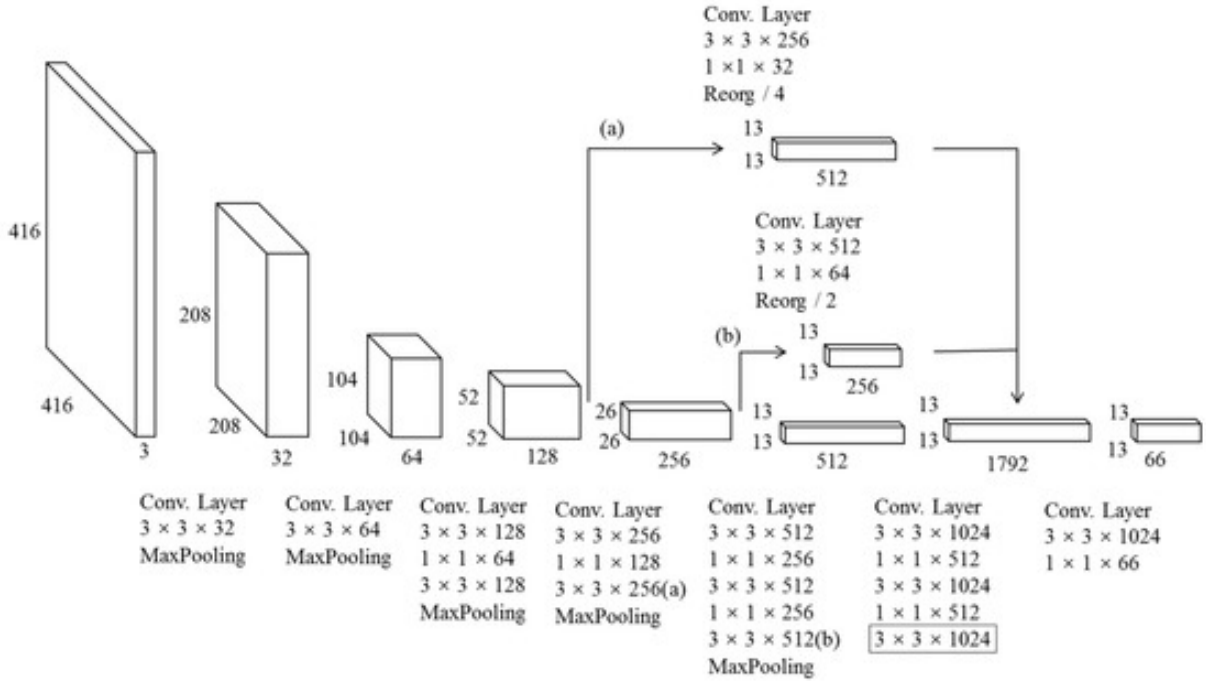

FIGURE 27: Aperçu de l'architecture YOLO (spécifiquement YOLOV2) [123].

— **U-Net** : U-Net [59] est une architecture CNN utilisée pour la segmentation d'images. U-Net est particulièrement utile pour la segmentation d'images médicales et biomédicales.

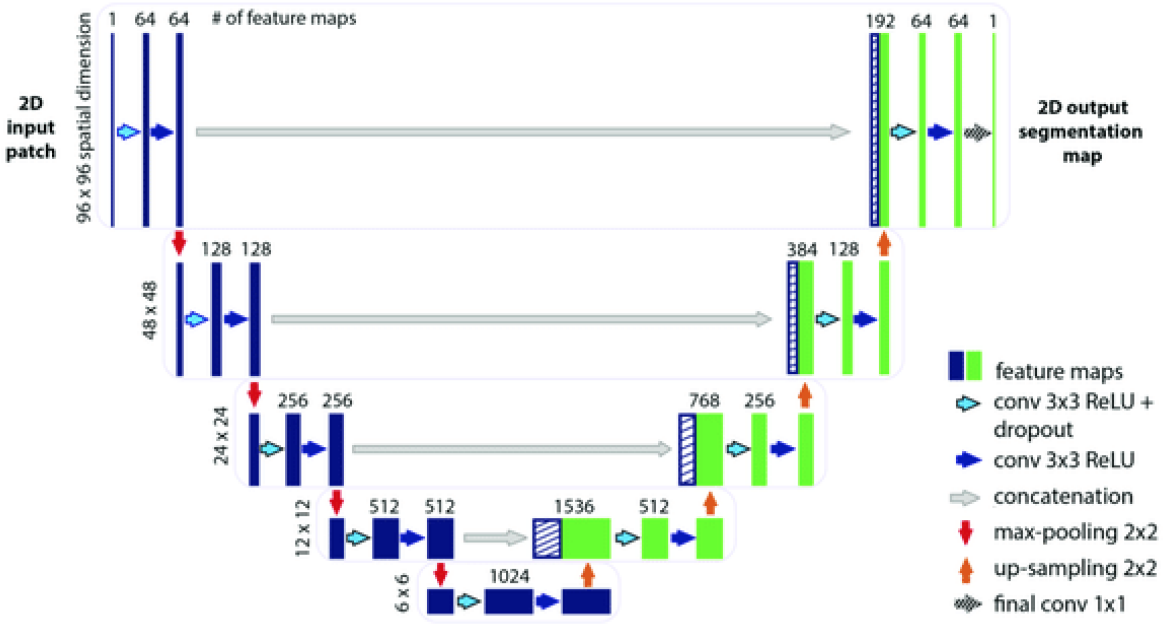

FIGURE 28: Aperçu de l'architecture U-Net [59].

2.4 Détection d'anomalies

2.4.1 Définition

La détection d'anomalies est un champ d'étude dans le domaine de l'apprentissage automatique qui se focalise sur l'identification des motifs dans les données qui ne se conforment pas aux motifs des données attendus. Ces motifs non-conformes sont souvent qualifiés d'anomalies, d'outliers, ou d'observations discordantes.

Un aspect important de la détection d'anomalies est la nature de l'anomalie recherchée. Les anomalies peuvent être classées en trois catégories [61] :

1. Anomalies ponctuelles : Si une instance de données individuelle peut être considérée comme anormale par rapport au reste des données, alors cette instance est qualifiée d'anomalie ponctuelle. C'est le type d'anomalie le plus simple et il fait l'objet de la majorité des recherches sur la détection d'anomalies. Considérons à titre d'exemple, la détection de la fraude par carte de crédit. Supposons que l'ensemble de données correspond aux transactions d'une carte de crédit individuelle. Pour simplifier, supposons que les données sont définies en utilisant une seule caractéristique : le montant dépensé. Une transaction pour laquelle le montant dépensé est très élevé par rapport à la plage normale de dépenses pour cette personne sera une anomalie ponctuelle. La figure 29 montre un exemple d'anomalies ponctuelles (N1, N2, N3, N4, N5).

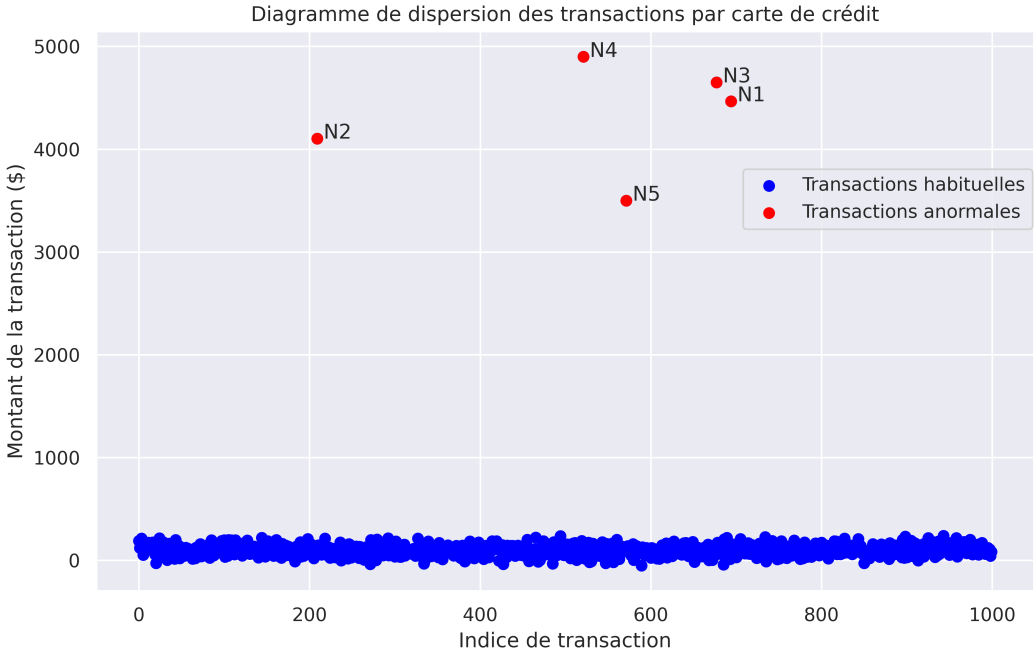

FIGURE 29: Exemple d'anomalies ponctuelles en relation avec les fraudes bancaires.

2. Anomalies contextuelles : Si une instance de données est anormale dans un contexte spécifique, alors elle est qualifiée d'anomalie contextuelle. La notion de contexte est induite par la structure de l'ensemble de données et doit être spécifiée dans le cadre de la formulation du problème. Chaque instance de données est définie à l'aide des deux ensembles d'attributs suivants :

   (a) Attributs contextuels : Les attributs contextuels sont utilisés pour déterminer le contexte (ou voisinage) pour cette instance. Par exemple, dans les ensembles de données spatiales, la longitude et la latitude d'un emplacement sont les attributs contextuels. Dans les données de séries temporelles, le temps est un attribut contextuel qui détermine la position d'une instance sur la séquence entière.

   (b) Attributs comportementaux : Les attributs comportementaux définissent les caractéristiques non contextuelles d'une instance. Par exemple, dans un ensemble de données spatiales décrivant les précipitations moyennes dans le monde entier, la quantité de précipitations à un endroit donné est un attribut comportemental.

   Le comportement anormal est déterminé en utilisant les valeurs des attributs comportementaux dans un contexte spécifique. Une instance de données peut être une anomalie contextuelle dans un contexte donné, mais une instance identique (en termes d'attributs comportementaux) pourrait être considérée comme normale dans un autre contexte. Cette propriété est essentielle pour identifier les attri-

buts contextuels et comportementaux dans une technique de détection d'anomalies contextuelles.

Les anomalies contextuelles ont été le plus souvent explorées dans les données de séries temporelles [62] et les données spatiales [63]. Un autre facteur clé est la disponibilité des attributs contextuels. Dans de nombreux cas, la définition d'un contexte est évidente, ce qui rend l'application d'une technique de détection d'anomalies contextuelles pertinente. Cependant, dans d'autres situations, définir un contexte n'est pas aisé. Ceci rend difficile l'application de ces techniques. La figure 30 montre un exemple d'anomalie contextuelle.

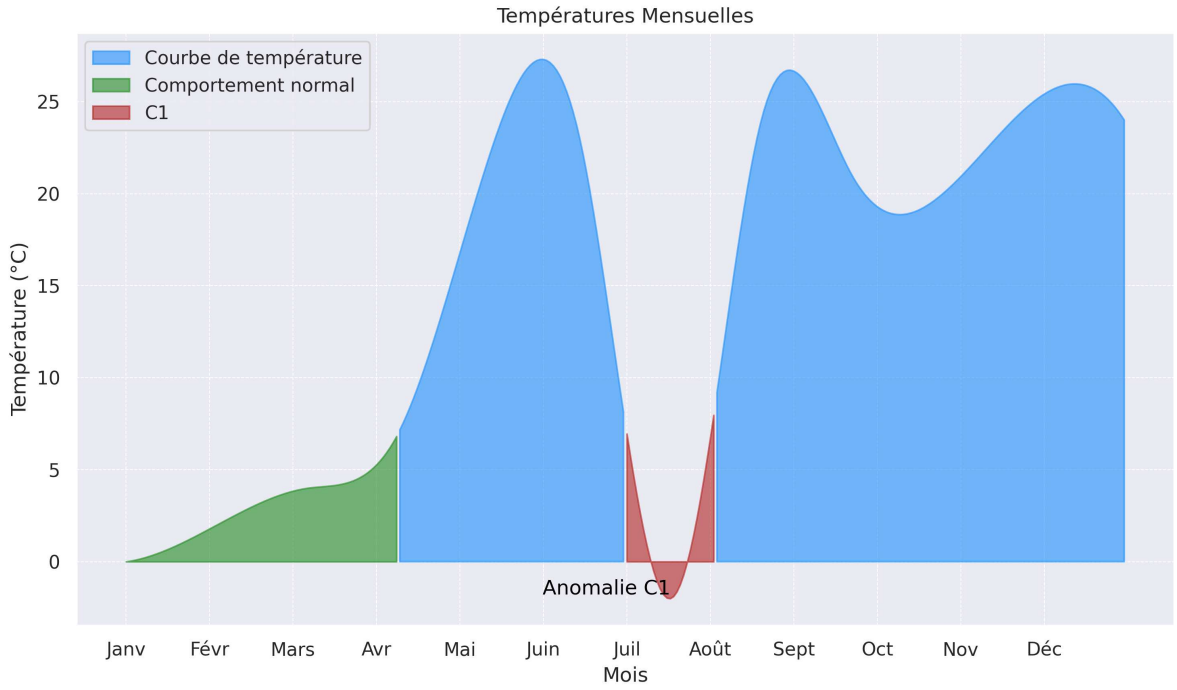

FIGURE 30: Exemple d'une anomalie contextuelle : La zone bleue indique des températures normales, la verte une baisse durant l'hiver (janvier à mars), mais une chute drastique des températures en juillet-aout (partie C1) suggère une anomalie contextuelle.

3. **Anomalies collectives :** Si un ensemble de données liées, c'est-à-dire des données interconnectées par une séquence ou une relation, est considéré comme anomal par rapport à l'ensemble des données, on parle d'une anomalie collective. Les instances de données individuelles dans une anomalie collective peuvent ne pas être des anomalies en elles-mêmes, mais leur occurrence commune en tant qu'ensemble est considérée comme anormale. La figure 31 illustre un exemple montrant le résultat d'un électrocardiogramme humain [64]. La zone mise en évidence dénote une anomalie car la même valeur faible persiste pendant une durée anormalement longue (correspondant à une Contraction Prématurée Auriculaire). Il faut noter que cette faible valeur en elle-même n'est pas une anomalie.

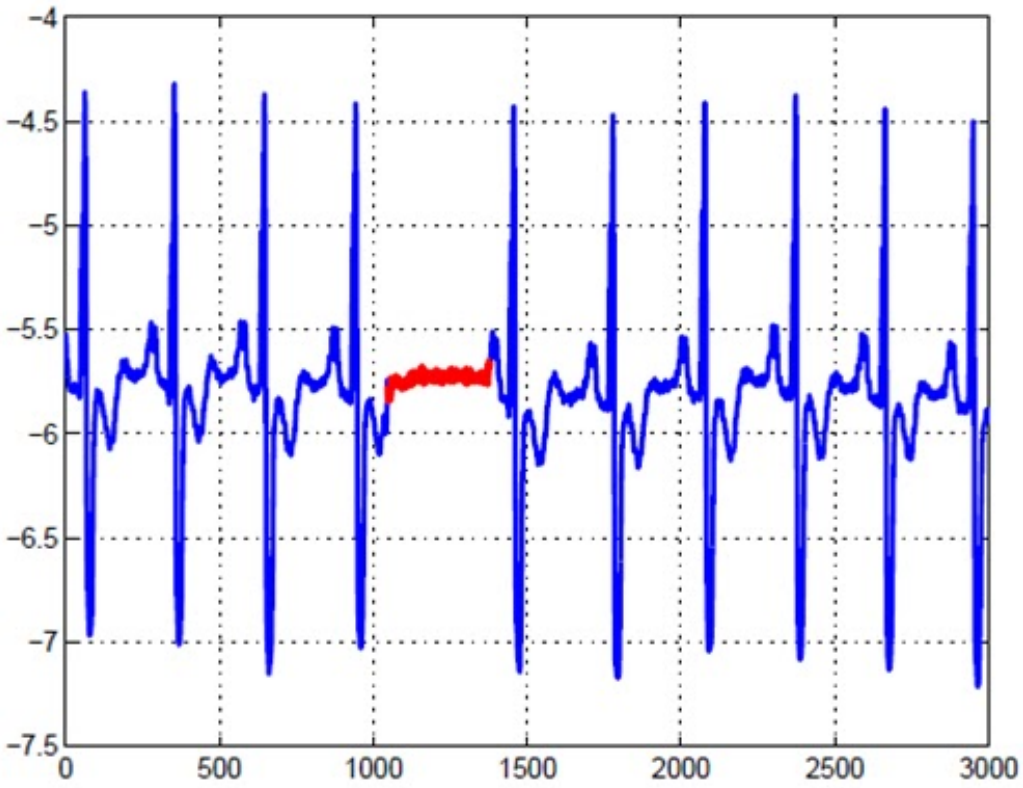

FIGURE 31: Exemple d'une anomalie collective [61].

Il convient de noter que, si des anomalies ponctuelles peuvent se produire dans n'importe quel ensemble de données, des anomalies collectives ne peuvent se produire que dans des ensembles de données où les instances sont liées. En revanche, l'apparition d'anomalies contextuelles dépend de la disponibilité d'attributs de contexte dans les données. Une anomalie ponctuelle ou une anomalie collective peut également être une anomalie contextuelle si elle est analysée par rapport à un contexte. Ainsi, un problème de détection d'anomalies ponctuelles ou de détection d'anomalies collectives peut être transformé en un problème de détection d'anomalies contextuelles en intégrant l'information de contexte.

Dans la partie suivante, nous allons voir les difficultés et défis de la détection d'anomalies dans la section 2.4.2, la section 2.4.3 sera dédiée aux différentes approches de la détection d'anomalies et finalement, la section 2.4.4 sera consacrée pour donner des exemples de modèles utilisés dans la détection d'anomalies.

2.4.2 Difficultés et défis

La détection d'anomalies, qui vise à repérer les motifs déviants du comportement normal attendu, présente plusieurs défis. Tout d'abord, établir une frontière nette entre ce qui est normal et anormal est complexe, car des observations peuvent chevaucher

cette limite. De plus, dans certains cas, comme les cyberattaques, les acteurs malveillants peuvent délibérément masquer des activités anormales pour qu'elles semblent normales. La définition de "normal" est aussi fluide, évoluant avec le temps et variant d'un domaine à l'autre. Ajoutez à cela la rareté des données étiquetées, essentielles pour former des modèles précis [61]. De plus, les méthodes non supervisées, bien que puissantes, ajoutent une couche supplémentaire de complexité. L'entraînement de ces modèles est difficile sans feedback clair, et l'interprétation des résultats peut être ambiguë, car il n'y a pas d'étiquettes de référence pour guider l'évaluation. En outre, le défi de distinguer le bruit, qui peut parfois ressembler à de véritables anomalies, persiste. Ainsi, chaque situation exige souvent une solution adaptée, influencée par le contexte spécifique d'application.

2.4.3 Approches utilisées et output

Les étiquettes associées à une instance de données dans un algorithme de détéction d'anomalies, indiquent si cette instance est normale ou anormale. Il est à noter que l'obtention de données étiquetées qui sont à la fois précises et représentatives de tous les types de comportements est souvent prohibitivement coûteuse, voire impossible dans certains cas. L'étiquetage est souvent effectué manuellement par un expert humain et nécessite donc un effort substantiel pour obtenir l'ensemble de données d'entraînement étiquetées. De manière générale, obtenir un ensemble d'instances de données anormales étiquetées qui couvre tous les types possibles de comportement anormal est plus difficile que d'obtenir des étiquettes pour le comportement normal. De plus, le comportement anormal est souvent dynamique. De nouveaux types d'anomalies peuvent apparaître, pour lesquelles il n'existe pas de données d'entraînement étiquetées. Dans certains cas, tels que la sécurité du trafic aérien, les instances anormales se traduiraient par des événements catastrophiques, et sont donc très rares.

Sur la base de la disponibilité des étiquettes, les techniques de détection d'anomalies peuvent fonctionner selon l'un des trois modes suivants :

1. **Détection d'anomalies supervisée.** Les techniques entraînées en mode supervisé supposent la disponibilité d'un ensemble de données d'entraînement qui contient des instances étiquetées pour la classe normale ainsi que pour la classe d'anomalie. L'approche typique dans ces cas est de construire un modèle prédictif pour les classes normales et anormales. Toute instance de données non observée est comparée au modèle pour déterminer à quelle classe elle appartient. Il y a deux problèmes majeurs qui se posent dans la détection d'anomalies supervisée. D'abord, les instances anormales sont beaucoup moins nombreuses que les instances normales dans les données d'entraînement. Les problèmes qui surviennent en raison de distributions de classes déséquilibrées ont été abordés dans la littérature sur l'exploitation des données et l'apprentissage automatique. Ensuite, obtenir

des étiquettes précises et représentatives, en particulier pour la classe d'anomalie, est généralement difficile. Un certain nombre de techniques ont été proposées pour injecter des anomalies artificiellement dans un ensemble de données normal afin d'obtenir un ensemble de données d'entraînement étiqueté [65].

2. **Détection d'anomalies semi-supervisée.** Les techniques qui fonctionnent en mode semi-supervisé supposent que les données d'entraînement comportent des instances étiquetées uniquement pour la classe normale. Comme elles ne nécessitent pas d'étiquettes pour la classe d'anomalie, elles sont plus largement applicables que les techniques supervisées. L'approche typique utilisée dans ces techniques consiste à construire un modèle pour la classe correspondant au comportement normal, et à utiliser le modèle pour identifier les anomalies dans les données de test [66].
3. **Détection d'anomalies non supervisée.** Tout comme les techniques semi-supervisées, les techniques qui fonctionnent en mode non supervisé ne nécessitent pas de données d'entraînement pour le cas des anomalies, et sont donc les plus largement applicables. Les techniques de cette catégorie font l'hypothèse implicite que les instances normales sont beaucoup plus fréquentes que les anomalies dans les données de test. Il est à noter que dans la littérature, l'utilisation des termes "non supervisé" et "semi-supervisé" pour la détection d'anomalies en apprentissage profond est souvent interchangeable par abus de langage. [66].

Un aspect important pour toute technique de détection d'anomalies est la manière dont les anomalies sont signalées. En général, les sorties produites par les techniques de détection d'anomalies sont de l'un des deux types suivants :

1. **Scores.** Les techniques de notation attribuent un score d'anomalie à chaque instance dans les données de test en fonction du degré auquel cette instance est considérée comme une anomalie. Ainsi, la sortie de ces techniques est une liste de scores d'anomalies.
2. **Étiquettes.** Les techniques de cette catégorie attribuent une étiquette (normale ou anormale) à chaque instance de test.

Les techniques de détection d'anomalies basées sur le score permettent à l'analyste d'utiliser un seuil spécifique au domaine pour sélectionner les anomalies les plus pertinentes. Les techniques qui fournissent des étiquettes binaires aux instances de test ne permettent pas directement aux analystes de faire un tel choix, bien que cela puisse être contrôlé indirectement par des choix de paramètres au sein de chaque technique [61].

2.4.4 Modèles globaux de détection d'anomalies

Cette section traite des modèles de détection d'anomalies, présentant une liste des modèles fréquemment utilisés avec leurs particularités et références.

— **Autoencodeurs (AE, VAE, CAE)** : Ces modèles basés sur les réseaux de neurones sont conçus pour comprimer des informations dans une représentation de faible dimensionnalité et ensuite les reconstruire. L'erreur de reconstruction est souvent utilisée pour identifier les anomalies [67].

— **Isolation Forest** : Cette méthode utilise des arbres de décision pour partitionner les données. Elle isole les anomalies en les divisant à travers des hyperplans aléatoires [68].

— **LOF (Local Outlier Factor)** : LOF compare la densité d'un point avec celle de ses voisins. Un score élevé indique que le point est loin de ses voisins [69].

— **LSTM (Long Short-Term Memory)** : Une forme spécialisée de réseaux de neurones récurrents, les LSTMs sont efficaces pour modéliser des dépendances temporelles longues dans les séries temporelles [70].

— **GANs (Generative Adversarial Networks)** : Un GAN est composé de deux réseaux de neurones, un générateur et un discriminateur. Ils peuvent apprendre à imiter n'importe quelle distribution de données, ce qui est utile pour détecter des anomalies [72].

— **ANNs (Artificial Neural Networks)** : Les ANNs sont capables de modéliser des frontières décisionnelles complexes et sont applicables à divers types de données, y compris pour la détection d'anomalies [34].

— **OC-SVMs (One-Class Support Vector Machines)** : Contrairement aux SVMs traditionnels, OC-SVM est formé uniquement avec des données normales et identifie les points qui se situent en dehors de la frontière normale comme des anomalies [172].

— **KNN (K-Nearest Neighbors)** : KNN fonctionne en identifiant les échantillons qui ont un nombre inhabituellement faible de voisins proches. Ces points sont souvent considérés comme des anomalies [73].

— **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** : C'est une méthode de clustering qui marque les régions de faible densité comme anomalies, séparant les données normales des anomalies basées sur la densité locale [74].

— **Elliptic Envelope** : Cette méthode estime la forme d'une enveloppe elliptique autour des données. Les points hors de cette enveloppe sont considérés comme des anomalies. Elle assume une distribution gaussienne des données [75].

— **CVAE (Conditional Variational Autoencoders)** : Similaire aux VAEs, mais ils sont conditionnés sur des variables supplémentaires, permettant une sortie plus spécifique et ciblée [76].

— **Réseaux Bayésiens** : Ces modèles graphiques probabilistes représentent un ensemble de variables et leurs dépendances conditionnelles. Ils sont utiles pour la détection d'anomalies en utilisant les techniques d'inférence bayésienne [77].

Un récapitulatif des modèles présentés pour la détection d'anomalies est fourni dans la table 2.2.

Modèle	**Type de Données**	**Type**	**Méthode d'apprentissage**
AE / VAEs / CAEs	Vision	DL	NS
Isolation Forest	Numérique	ML	NS
LOF	Numérique	ML	NS
LSTM	Séries Temporelles	DL	S, SS, NS
GANs	Vision	DL	SS, NS
ANNs	Multiple	DL	S, SS, NS
OC-SVMs	Multiple	ML	NS
KNN	Multiple	ML	S, NS
DBSCAN	Multiple	ML	NS
Elliptic envelope	Numérique	ML	NS
CVAE	Multiple	DL	SS
Réseaux Bayésiens	Multiple	ML	S, SS, NS

TABLE 2.2: Synthèse des modèles pour la détection d'anomalies.

2.4.5 Modèles pour divers domaines

La méthode décrite dans [157] est conçue pour la détection et la segmentation non supervisée d'anomalies dans l'imagerie cérébrale. Elle est entraînée à partir de données saines et identifie les régions anormales à travers des étapes de diffusion. Cette méthode offre des temps d'inférence rapides, les rendant plus adaptés pour des applications cliniques. Le DSR [160] détecte des anomalies de surface, en particulier dans des milieux industriels. PatchCore [158] segmente les images en utilisant un réseau pré-entraîné pour chaque segment. AnoVL [161] localise les anomalies dans des images sans étiquettes, combinant l'attention visuelle et l'interprétation textuelle. PyramidFlow [162] est spécialisée pour les défauts dans des images industrielles haute résolution. SimpleNet [159] est conçu pour la détection d'anomalies dans des images, utilisant des caractéristiques pré-entraînées et un mécanisme de bruit. La méthode DefGAN [163] automatise la détection des défauts sur les isolateurs de la caténaire du train à grande vitesse en utilisant les GANs [221]. Elle génère des échantillons défectueux à partir de représentations latentes érodées, améliorant ainsi la fiabilité du classificateur de détection des défauts. La méthode évalue les anomalies en se basant sur la probabilité d'anomalie du classificateur et l'erreur de reconstruction de l'autoencodeur de débruitage. La méthode mentionnée dans [164] détecte les anomalies dans les entrepôts avec des véhicules autonomes guidés en utilisant la "surprise bayésienne", une métrique mesurant les différences entre les attentes et les observations. Elle exploite des modèles d'apprentissage non supervisés pour repérer les événements inattendus sans nécessiter de données annotées pour les anomalies, ce qui la rend utile

dans la logistique industrielle avec des véhicules autonomes guidés. Dans le domaine de la santé, f-Anogan [175] est un modèle de détection d'anomalies basé sur les GANs qui a été largement utilisé. Il est spécialement conçu pour identifier les irrégularités dans les données médicales, ce qui en fait un outil essentiel pour la surveillance des patients et le diagnostic précoce de pathologies. D'autre part, dans le contexte industriel, Le modèle VT-ADL [178], une architecture autoencodeur hybride combinant les transformeurs de vision et les couches de convolution avec l'intégration d'un réseau de densité de mélange gaussien (*Gaussian mixture density network*), joue un rôle majeur dans l'analyse avancée des données visuelles. Il permet la détection rapide de défauts de production et contribue à l'amélioration de la qualité des produits. Ce mélange gaussien aide à localiser précisément les zones anormales à partir des informations spatiales préservées par les réseaux de transformeurs.

2.4.6 Domaines d'application

La détection d'anomalies offre un large ensemble d'applications dans de nombreux domaines. Voici quelques exemples :

- **Santé** : La détection d'anomalies en santé est essentielle pour repérer des conditions inattendues. Par exemple, dans l'imagerie médicale, elle peut identifier des tumeurs précoces non visibles à l'œil nu. Dans le suivi des signes vitaux, elle peut alerter sur des irrégularités cardiaques avant qu'elles ne deviennent critiques. De plus, en analysant les données épidémiologiques, elle peut détecter une propagation inhabituelle d'une maladie, indiquant potentiellement le début d'une épidémie. Ces détections précoces favorisent des interventions rapides et une meilleure gestion des soins.
- **Finance** : Dans le domaine financier, la détection d'anomalies est souvent utilisée pour identifier des transactions frauduleuses. Elle peut également être utilisée pour identifier des schémas de trading anormaux qui pourraient indiquer une manipulation du marché.
- **Fabrication** : Les entreprises de fabrication utilisent la détection d'anomalies pour identifier les défauts dans les produits et prédire les pannes de machines, ce qui peut contribuer à augmenter la qualité des produits et à réduire les temps d'arrêt.
- **Informatique** : La détection d'anomalies peut être utilisée pour identifier un trafic réseau inhabituel qui pourrait indiquer une cyberattaque en cours. Elle peut également aider à identifier des menaces potentielles pour la cybersécurité.
- **Transport** : Dans le domaine du transport, la détection d'anomalies peut être utilisée pour la détection d'obstacles pour les véhicules autonomes, améliorant ainsi leur sécurité et leur efficacité.
- **Vente au détail** : Enfin, dans le secteur de la vente au détail, la détection d'ano-

malies peut aider à identifier des schémas de vente anormaux ou des problèmes potentiels d'inventaire, ce qui peut permettre aux entreprises de gérer plus efficacement leurs stocks.

Domaine	**Application**
Santé	Détection de conditions de patients inhabituelles, Prédiction de potentielles épidémies
Finance	Détection de transactions frauduleuses, Identification de schémas de trading anormaux
Fabrication	Identification de défauts dans les produits, Prédiction de pannes de machines
Informatique	Détection d'un trafic réseau inhabituel, Identification de menaces potentielles pour la cybersécurité
Transport	Détection d'obstacles pour les véhicules autonomes
Vente au détail	Détection de schémas de vente anormaux, Identification de problèmes potentiels d'inventaire

TABLE 2.3: Domaines d'application de la détection d'anomalies

2.5 Conclusion

Ce chapitre a fourni une description du projet Train de Fret Autonome (TFA). Il a débuté par une clarification des niveaux d'automatisation. La description a ensuite porté sur les éléments clés du système autonome ferroviaire TFA, identifiant les contraintes et défis majeurs. De manière parallèle, une présentation détaillée de l'intelligence artificielle et plus spécifiquement de l'apprentissage automatique et profond a été abordée, mettant en lumière les fonctionnements des architectures neuronales comme les ANNs et les CNNs. Le chapitre a aussi souligné l'importance de divers datasets de vision et architectures CNN, illustrant leur utilité en vision par ordinateur. Enfin, la section sur la détection d'anomalies a couvert sa définition, les défis associés, les méthodes couramment utilisées, ainsi que les modèles dédiés et leurs domaines d'application. En résumé, ce chapitre constitue une base pour les discussions futures, englobant les aspects fondamentaux de l'autonomie ferroviaire et des technologies connexes.

Chapitre 3

Etat de l'art

3.1 Introduction

La sécurité et l'efficacité du transport ferroviaire dépendent fortement de la capacité à détecter les obstacles sur les voies et à surveiller l'environnement des trains. Les systèmes traditionnels, qui reposent principalement sur le traitement d'images et les capteurs physiques, rencontrent certaines limitations. Avec l'avènement de technologies modernes, l'application des techniques d'apprentissage profond, comme les réseaux de neurones [84], a montré des avancées prometteuses, notamment dans les domaines de la vision par ordinateur et du traitement du langage naturel. Cependant, alors que la voiture autonome a vu une explosion des recherches exploitant l'intelligence artificielle pour la détection d'obstacles routiers, le secteur ferroviaire semble avoir été quelque peu négligé. Cette différence d'attention est surprenante, étant donné les défis uniques du ferroviaire. Les voies ferrées, par exemple, peuvent être obstruées par divers obstacles tels que des piétons, des animaux comme les vaches ou les renards, des véhicules non autorisés, ou encore des anomalies potentielles telles que des rochers ou objets tombés. Plusieurs raisons pourraient expliquer ce déséquilibre ; La complexité intrinsèque du système ferroviaire, les défis associés à la collecte de données d'entraînement adéquates, ou peut-être un manque de prise de conscience sur l'importance de ces problématiques. Néanmoins, compte tenu des progrès récents de l'IA, il y a un potentiel immense à exploiter pour renforcer la sécurité et l'efficacité du transport ferroviaire grâce à une détection d'obstacles améliorée.

Il est à noter que le nombre de travaux dans le domaine de la détection d'obstacles sur les voies ferrées utilisant l'apprentissage profond est relativement restreint comparé à ceux dédiés aux véhicules autonomes. Plusieurs facteurs pourraient expliquer cette tendance :

— **Complexité et diversité de la détection d'obstacles dans les véhicules autonomes :** À la différence des systèmes ferroviaires, qui présentent une infrastructure principalement linéaire et homogène, les environnements routiers auxquels font face les véhicules autonomes sont caractérisés par une grande variabilité et imprévisibilité, ce qui stimule davantage de recherches. Par exemple, le travail présenté dans [90] met en exergue l'importance d'augmenter la sécurité et l'efficacité de la conduite autonome par la détection et la prédiction d'une diversité d'obstacles.

— **Ampleur du marché et potentiel économique :** Le marché des véhicules autonomes possède un potentiel bien supérieur à celui des trains autonomes. Cette différence incite à un investissement privé accru et stimule la recherche dans le domaine. Une analyse de McKinsey & Company [91] aborde les remarquables perspectives économiques liées aux véhicules autonomes.

— **Flexibilité accrue pour les essais et la mise en œuvre :** Les véhicules autonomes peuvent faire l'objet d'essais et d'intégrations de manière plus aisée et progressive par rapport aux trains autonomes. Ils peuvent être testés individuellement dans divers environnements et leur intégration peut s'effectuer de manière incrémentale, contrairement aux trains qui opèrent au sein d'un réseau fortement régulé, systématique et interconnecté.

— **Contraintes réglementaires et de sécurité :** Les systèmes ferroviaires sont en général soumis à une réglementation stricte en raison des conséquences désastreuses potentielles en cas de défaillance. Cela peut constituer un frein pour les chercheurs en raison de la complexité d'obtention des autorisations nécessaires pour des essais et des mises en œuvre grandeur nature. L'étude dans [92] traite des défis réglementaires significatifs liés à l'implémentation de la technologie autonome dans l'industrie ferroviaire.

— **Disponibilité de jeux de données :** L'accès à des jeux de données volumineux et diversifiés pour l'entraînement de modèles d'apprentissage profond est nettement plus important pour les véhicules autonomes, principalement en raison des facteurs énoncés précédemment (complexité accrue, marché plus vaste, etc.). Des jeux de données réputés tels que Waymo Open Dataset [93], nuScenes [90], KITTI dataset [129], et ApolloScape [94] soutiennent les chercheurs dans le développement de véhicules autonomes.

L'importance de la recherche en détection d'obstacles ferroviaires est fondamentale, en raison de la signification des chemins de fer dans le transport global. Des systèmes de détection d'obstacles efficaces sont nécessaires pour éviter les accidents. Une analyse exhaustive de ces méthodes permet une meilleure sécurité et une transition vers l'autonomisation des trains. Elle sert aussi à guider les futures recherches. Les figures 32, 33 et 34 montrent l'évolution de l'intérêt de la recherche sur ce sujet de 2004 à 2022.

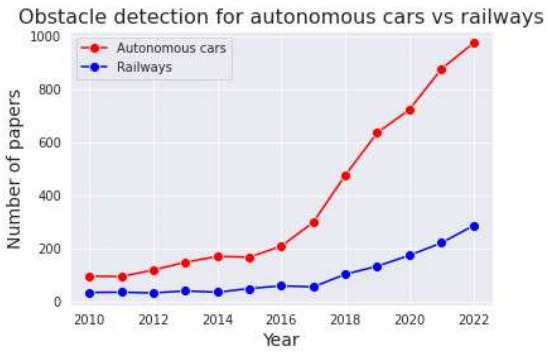

FIGURE 32: Évolution des recherches sur la détection d'obstacles ferroviaire et la voiture autonome de 2010 à 2022 - Scopus

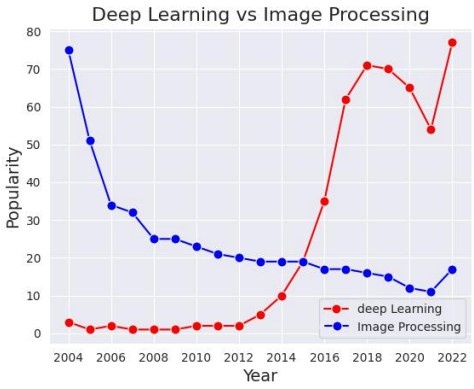

FIGURE 33: Évolution générale des recherches sur l'apprentissage profond et les techniques de traitement d'images 2004 à 2022 - Scopus

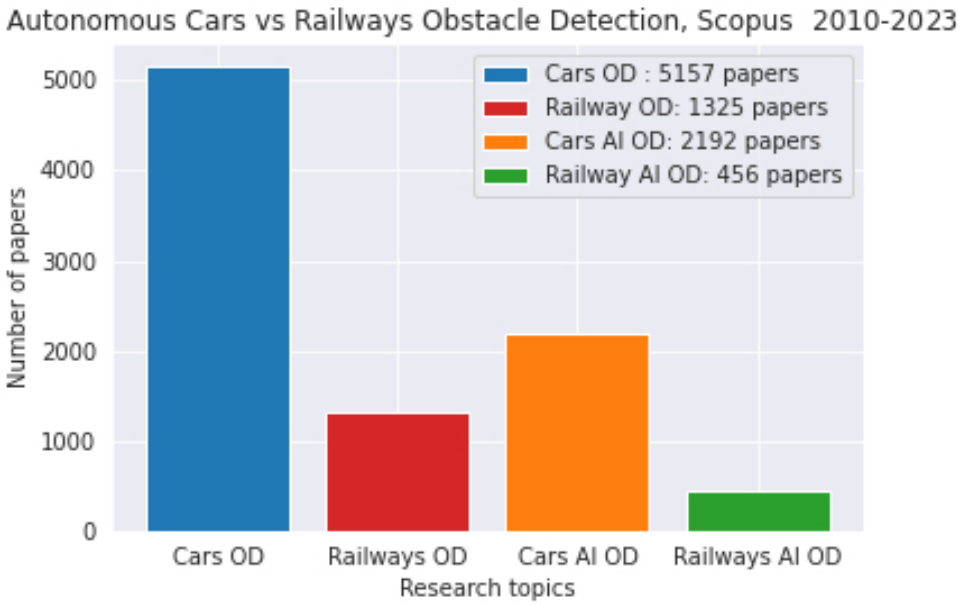

FIGURE 34: Évolution de la recherche de 2010 à 2023 : Analyse comparative des travaux sur la détection d'obstacles ferroviaires et de voiture autonome par approche IA, ainsi que de la détection globale des obstacles dans les deux domaines.

L'étude se focalise sur l'utilisation de l'apprentissage profond pour la détection des obstacles ferroviaires, avec un penchant vers les méthodes classiques telles que le traitement d'images et les méthodes qui reposent sur des capteurs physiques. Elle est divisée en trois parties : méthodes supervisées, non supervisées, et les méthodes classiques mentionnées précédemment. Chaque travail qui représente l'une de ces catégories sera détaillé et discuté.

3.2 Détection d'obstacles ferroviaires via l'apprentissage profond

L'utilisation des méthodes d'apprentissage profond dans le domaine ferroviaire offre de nombreuses possibilités pour améliorer la sécurité et l'efficacité du transport. En ce qui concerne la vision, les techniques d'apprentissage profond supervisées ont été largement appliquées pour la détection des obstacles ferroviaires. Ces méthodes exploitent des réseaux neuronaux pour identifier et classer précisément les obstacles présents sur les voies. Elles offrent une grande capacité de reconnaissance des formes et des caractéristiques, ce qui permet une détection fiable. Cependant, elles nécessitent des ensembles de données annotées et peuvent être sensibles à la variabilité des conditions environnementales, ce qui peut entraîner des performances moins robustes dans des situations imprévues. En ce qui concerne la détection d'anomalies dans l'environnement des trains, les techniques d'apprentissage profond non supervisées sont utilisées. Ces méthodes permettent d'explorer et de découvrir des motifs cachés dans les données sans annotations préalables. Elles offrent une grande flexibilité et sont capables de détecter des anomalies inattendues. Cependant, elles peuvent être plus sensibles au bruit et nécessitent une bonne maîtrise du jeu de données d'entraînement afin de bien définir le cas normal sans anomalies sur lequel le modèle sera entrainé pour obtenir des résultats précis.

En résumé, l'utilisation de l'apprentissage profond dans le domaine ferroviaire présente des avantages considérables en termes de détection d'obstacles et de surveillance de l'environnement des trains. Les techniques supervisées offrent une précision et une fiabilité élevées, tandis que les méthodes non supervisées permettent la détection d'anomalies inattendues. Cependant, il est important de noter que chaque méthode présente ses propres qualités et défauts, et leur choix dépend des exigences spécifiques du système ferroviaire et des contraintes opérationnelles.

3.2.1 Détection via apprentissage supervisé

Les auteurs dans [106] utilisent des véhicules aériens sans pilote (UAV) et des techniques d'apprentissage profond (DL) pour la détection de la végétation et des objets sur les voies ferrées. Les auteurs ont créé un ensemble de données dédié composé de trois sous-ensembles : DS1 qui contient des données publiques telles que des vidéos YouTube, DS2 qui contient leurs propres acquisitions vidéo et DS3 qui combine DS1 et DS2. L'ensemble de données est ensuite utilisé pour entraîner deux versions de Faster R-CNN [107] afin de détecter 12 classes telles que "*object on the tracks*", "*fence*", etc. Les auteurs prétendent atteindre une précision moyenne (mAP) [108] comprise entre 51 % et 61 %. Les auteurs prouvent en effet l'utilisation des véhicules aériens sans pilote et l'apprentissage profond pour la détection des objets sur les voies ferrées, mais leur travail est limité. Aucun détail

sur l'ensemble de données n'est fourni et il n'est pas disponible publiquement. L'utilisation uniquement de Faster-RCNN [107] et l'absence de comparaison avec d'autres méthodes limitent également l'impact du travail. La figure 35 montre les résultats de détection par cette approche.



FIGURE 35: Résultats de détection basée sur UAVs [106]

Dans l'étude présentée par [118], la question des accidents ferroviaires liés au déraillement a été traitée en utilisant des réseaux neuronaux profonds pour identifier les obstacles. Un ensemble de données de détection d'obstacles ferroviaires intitulé "*Railroad Obstacle Detection*" (RROD) a été créé en utilisant des images en temps réel de voies ferrées capturées par des UAVs. Cet ensemble de données a été recueilli à l'aide d'un appareil photo Sony DSC-RX1RM2, d'une résolution de 1920x1080 et d'une capacité de 20 M (mégapixels). Il contient plus de 2000 images de 5 classes considérées comme des obstacles potentiels : '*Barrel*', '*Boulder*', '*Branch*', '*IronRod*', '*Jerrycan*' et '*Person*'. Plusieurs modèles de réseaux neuronaux profonds avec différents backbones (réseau DNN de base) ont été entraînés et évalués à l'aide de cet ensemble de données. Les résultats expérimentaux montrent que les modèles SSD [112] avec le backbone MobileNetV2 [119], Faster RCNN [107] et SSD avec le backbone ResNet50 [112] [114] ont mieux performé que les autres modèles, avec des précisions moyennes de 96,75%, 84,75% et 83,75% mAP respectivement. Comme mentionné précédemment, limiter la détection à seulement quelques classes pose toujours problème pour l'apprentissage supervisé, et aucune information qualitative supplémentaire sur l'ensemble de données n'est donnée par rapport à ce travail. La figure 36 et 37 montrent les résultats de détection ainsi que des exemples des diférentes classes d'obstacles utilisées dans leur dataset.

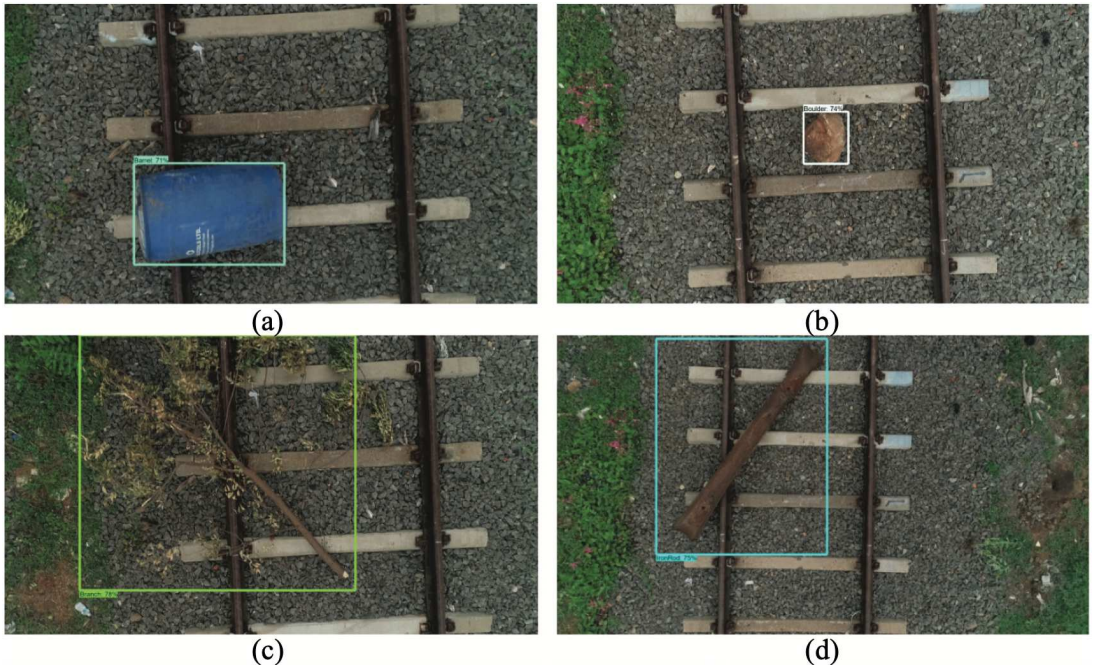

FIGURE 36: Résultats de la détection des obstacles par la méthode décrite dans [118]

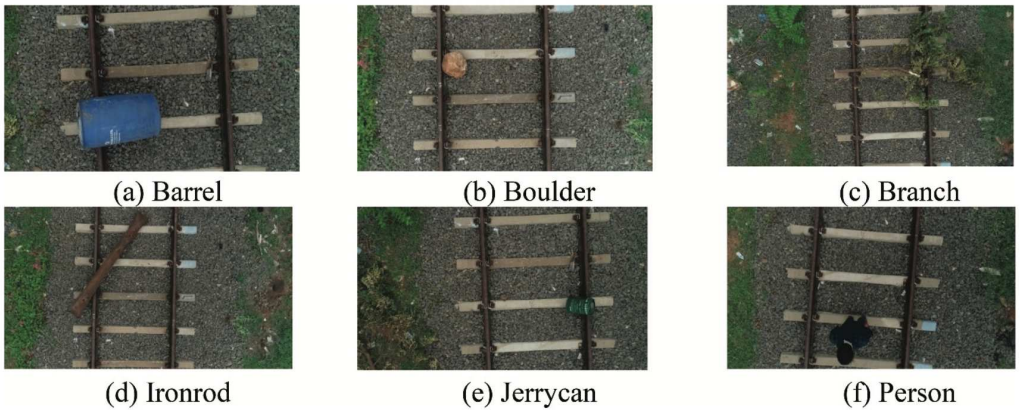

FIGURE 37: Aperçu des différentes classes dans le dataset utilisé dans [118]

Ls auteurs de l'étude [109] proposent un modèle basé sur les SSDs [112] intitulé FE-SSD. La méthode FE-SSD intègre un module de détection préalable basé sur la connexion inverse avec des réseaux de priorité d'objet [110], un bloc de transfert de caractéristiques du réseau plus rapide et optimal (FB-Net) [111], ainsi qu'un module novateur d'amélioration du champ récepteur afin d'optimiser la précision et la robustesse des caractéristiques. Cette méthode a été évaluée sur un ensemble de données de trafic ferroviaire élaboré par les auteurs, affichant un taux de détection moyen de 89,5% à 38 FPS, surpassant ainsi les performances d'autres modèles basés sur SSD [112]. FE-SSD s'avère efficace pour la détection d'objets ferroviaires dans des conditions réelles. La figure 38 illustre les résultats de détection obtenus grâce au modèle FE-SSD.

Dans le travail de [122], un modèle Improved-YOLOv4 [126] a été proposé pour la détection des obstacles dans les trains de transport ferroviaire afin de surmonter les li-

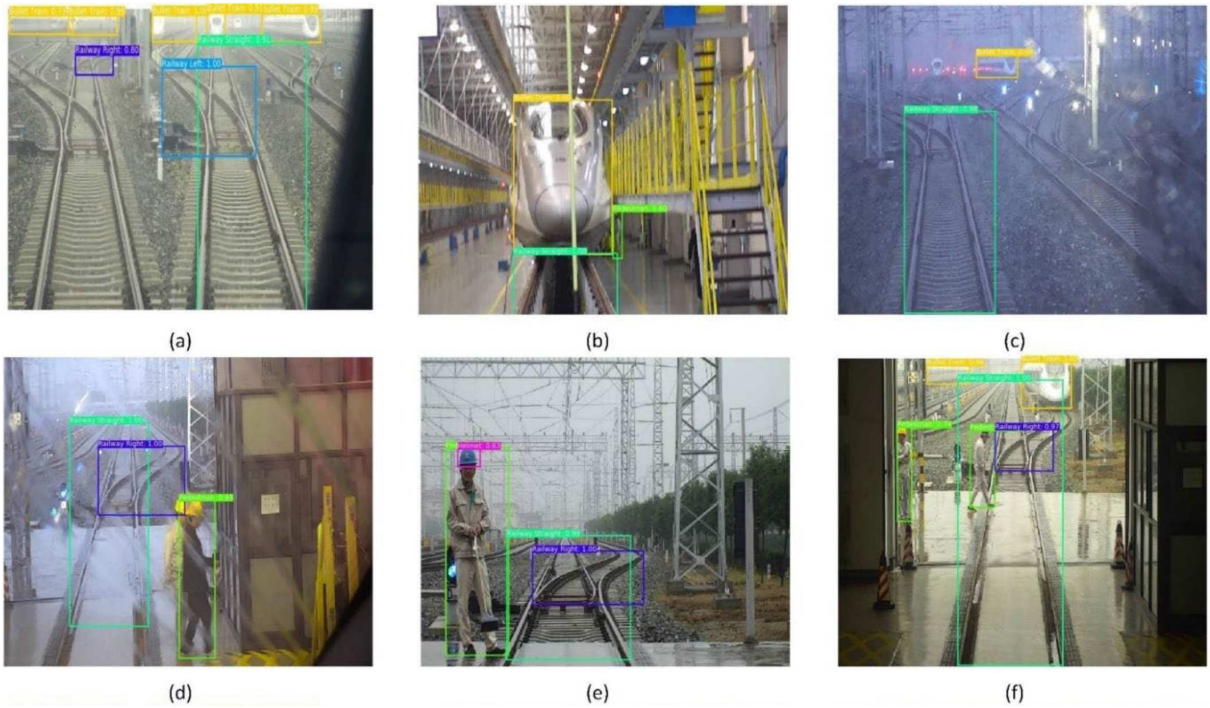

FIGURE 38: Performances du modèle FE-SSD [109]

mitations des méthodes de détection traditionnelles. Le nouveau réseau D-CSPDarknet a été conçu comme réseau de base pour l'extraction des caractéristiques. Les auteurs ont entraîné leur modèle sur KITTI [129], puis l'apprentissage par transfert a été utilisé pour reconnaître les classes considérées comme des obstacles dans le jeu de données MS-COCO [128]. Pour améliorer la précision et la vitesse de traitement, la Région d'Intérêt (RoI) a été créée à l'aide d'un masque. Les auteurs affirment que la division de la région d'intérêt lors du prétraitement améliore la précision. Improved-YOLOv4 atteint une précision de 93% avec l'utilisation de la RoI. Le modèle a été testé sur des données réelles collectées dans l'environnement d'exploitation des trains et exécuté sur le kit de développement NVIDIA Jetson AGX. Les résultats ont montré que l'utilisation d'une RoI améliorait la précision du modèle de 2,42% et atteignait une vitesse de traitement de 4ms. Le modèle Improved-YOLOv4 parvient à un bon équilibre entre vitesse de détection et précision. La figure 39 montre les résultats obtenus de la détection avec Improved YOLOV4 au niveau des RoIs et de l'image globalement. La figure 39 montre les résultats obtenus avec la méthode proposée [122]

Dans [117], un modèle de détection d'intrusions multitâche est proposé pour détecter les obstacles dans les scènes ferroviaires. Le modèle obtient des résultats compétitifs à la fois en détection d'objets et en détection des rails, tout en utilisant un algorithme d'optimisation multi-objectifs avec un temps d'inférence rapide de 50 images par seconde (FPS). Le jeu de données utilisé a été créé en combinant des données réelles d'exploitation ferroviaire collectées sur un site d'essai, des données simulées à l'aide du moteur Unreal Engine et des données collectées sur le web. Le jeu de données comprend une variété

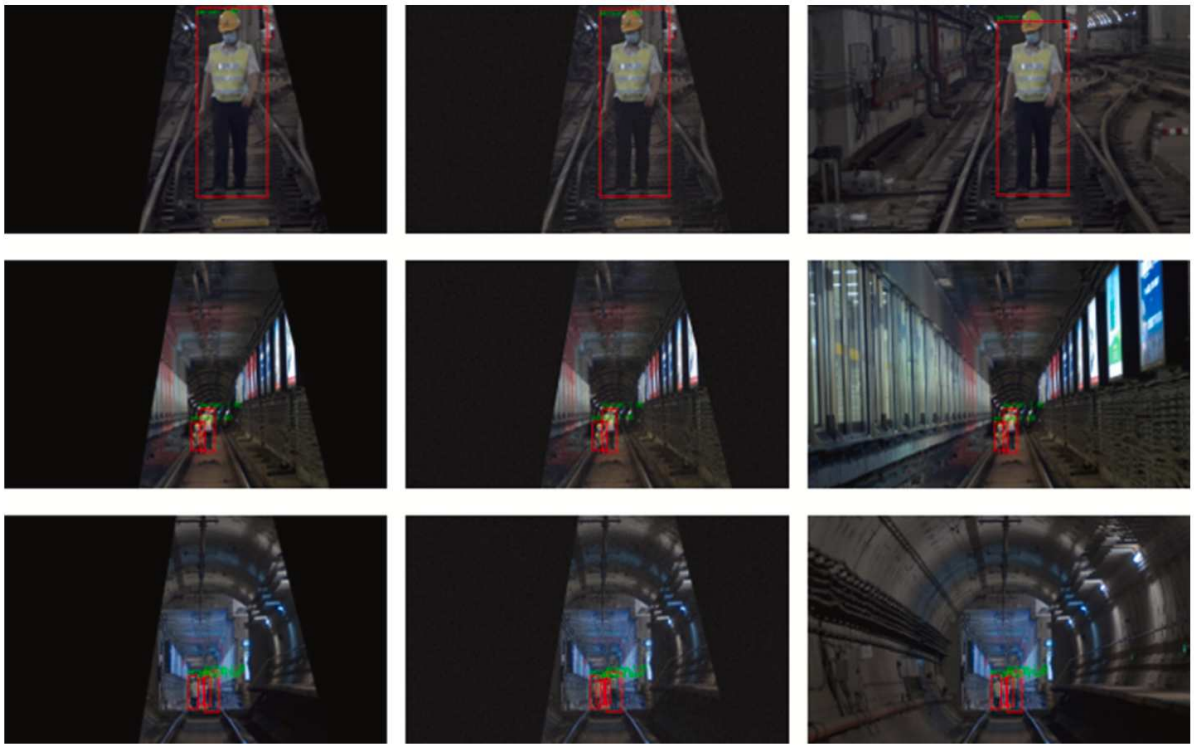

FIGURE 39: Performances d'Improved-YOLOv4 dans les RoIs [122]

de types et de conditions d'obstacles, tels que des personnes, des animaux ainsi que des conditions météorologiques variées. Le jeu de données a été prétraité pour éliminer les images de mauvaise qualité ou répétitives, ce qui a donné un total de 2400 images de jour et nuit, réparties en ensembles d'entraînement, de validation et de test. L'approche multitâche peut effectuer à la fois la reconnaissance des rails et la détection d'obstacles en utilisant le réseau RepVGG [113] et a été testée sur une variété de backbones telles que ResNet50 [114] et DarkNet53 [116] avec des précisions allant de 35% à 69,6%. Dans leurs expériences, RepVGG obtient les meilleurs résultats. Bien que l'article présente des résultats comparatifs et des contributions authentiques, la méthode manque d'informations importantes concernant les données d'entraînement et de test, ce qui rend difficile l'évaluation de la validité des résultats. De plus, l'utilisation de données purement synthétiques pour la détection d'obstacles peut introduire des biais et des limitations. De plus, l'absence d'informations qualitatives et quantitatives sur la détection des obstacles et ses classes soulève des questions sur l'exactitude et la fiabilité des résultats. La figure 40 montre les résultats de détection sur les scènes 3D synthétiques avec la méthode multitâche.

L'un des premiers travaux qui ont tenté d'adopter un réseau neuronal convolutif (CNN) [121] pour détecter les obstacles est décrit dans [120]. La méthode est simple et propose l'utilisation d'un modèle de réseau neuronal basé sur le modèle Faster R-CNN [107] pour détecter un nombre précis d'objets jouant le rôle d'obstacles. Le modèle est entraîné et testé sur un dataset rassemblé par les auteurs et donne une précision de 94,85%. Bien que ce soit l'un des premiers travaux utilisant des CNNs pour détecter les obstacles, l'absence

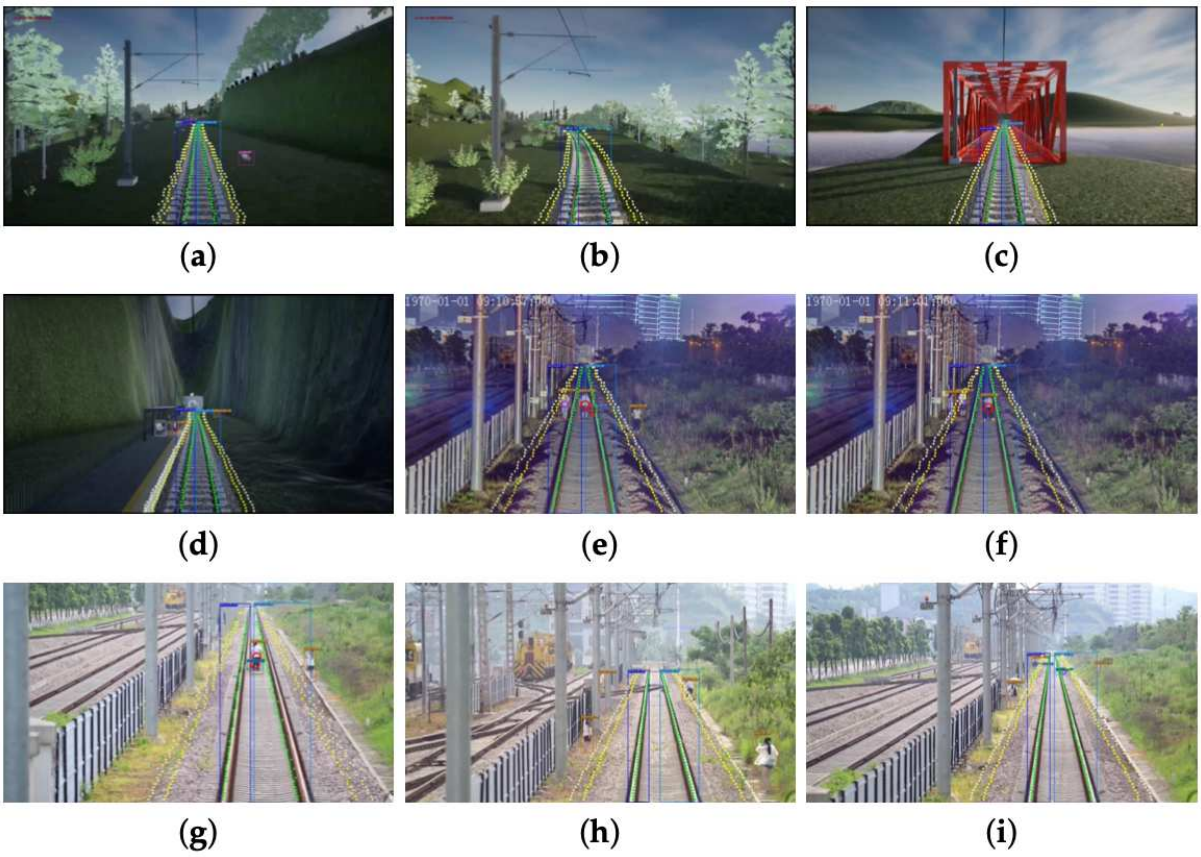

FIGURE 40: Résultats de détection sur des données synthétique avec la méthode multitâche [117]

d'informations sur les données utilisées soulève des questions quant à la validité et à la fiabilité des résultats. De plus, aucun détail n'est donné sur le type d'obstacles détectés ou sur le nombre d'obstacles détectés dans le jeu de données privé. Le manque de résultats qualitatifs ou de description des obstacles détectés suscite des préoccupations quant à la spécificité et à la praticité du modèle. Cela soulève la question de la transparence et de la généralisabilité des résultats. La figure 41 montre quelques résultats de détection avec le modèle entraîné Faster-RCNN.



FIGURE 41: Résultat de détection du travail proposé [120]

Les auteurs de l'article [131] ont conçu un système de détection d'objets dans les

systèmes de transport ferroviaire qui utilise l'edge computing. La détection d'objets est une tâche cruciale dans les transports ferroviaires, mais les ressources informatiques limitées des équipements embarqués dans les trains rendent difficile l'exécution de ces tâches. La solution proposée utilise l'edge computing pour alléger la charge sur les nœuds cloud, permettant ainsi une détection d'objets en temps réel avec des ressources embarquées limitées. Les auteurs présentent deux méthodes basées sur la segmentation pour l'inférence collaborative : une pour le traitement sériel et une pour le traitement parallèle. L'objectif de cette dernière méthode est de minimiser les coûts en ressources et le temps d'inférence. La méthode proposée a été testée et s'est révélée efficace lors d'expériences réelles. Les auteurs suggèrent la possibilité d'améliorer davantage la méthode en la combinant avec l'apprentissage par renforcement. Le modèle YOLOV3 [124] a été utilisé lors de la phase d'entraînement sur les ensembles de données Pascal VOC [130] et MS-COCO [128]. Seules deux classes d'objets ont été utilisées : les trains et les personnes, sans plus de détails sur l'ensemble de données de test. Aucune information supplémentaire n'a été partagée sur les performances du modèle entraîné par rapport à cette méthode. Les résultats de cette méthode sont visibles au niveau de la figure 42



FIGURE 42: Résultat de détection du travail proposé [131]

Le travail réalisé par les auteurs dans [132] consiste en la mise en œuvre d'un framework en deux étapes pour la détection en temps réel d'obstacles qui pénètrent sur les voies ferrées. La première étape est un réseau léger de classification d'images ferroviaires pour classifier l'image d'entrée en normale ou anormale. Une version améliorée de l'unité d'inversion résiduelle (IR) [133] est utilisée avec une convolution sélective de noyau

et un mécanisme d'attention léger pour obtenir une classification efficace et précise. La deuxième étape est un réseau de détection d'objets étrangers pour détecter l'emplacement et la classe des objets dans l'image anormale. Un ensemble de données d'objets étrangers ferroviaires a été constitué avec 3145 images : 1523 normales et 1622 anormales. Les images anormales ont été créées en plaçant des objets tels que des bouteilles, des pneus et des parapluies sur la voie ferrée. Les images ont été étiquetées manuellement avec des boîtes englobantes. Jusqu'à 1885 images ont été utilisées pour entraîner le réseau de classification d'images et les 1260 restantes ont été utilisées pour évaluer les performances. Les 1622 images d'anomalies ont été utilisées pour évaluer la détection d'objets étrangers de la deuxième étape, avec 487 images de test contenant 176 bouteilles, 204 pneus et 107 parapluies. Le classifieur de la première étape a été entraîné et évalué en comparaison avec trois modèles de classification d'images basés sur l'apprentissage profond : le transformeur de vision (ViT) [134], ResNet-50 [114] et MobileNetV2 [119]. Les performances du modèle proposé se sont révélées comparables aux autres modèles, avec un mAP [108] de 96,88%, inférieur à celui de ViT avec 97,51%. Cependant, le modèle proposé atteint le plus haut nombre d'images par seconde (FPS), atteignant 72,18, ce qui satisfait l'exigence de temps réel de 60 FPS pour le système de détection d'objets étrangers. Le modèle proposé est également léger, avec seulement 1% des paramètres par rapport à ViT [134] et 40% par rapport à MobileNetV2 [119]. La figure 43 montre les résultats obtenus par la nouvelle approche en deux étapes.



FIGURE 43: Résultat de détection du travail proposé [132]

Concernant le monitoring de l'environnement des trains autonomes, l'étude présentée

dans [136] introduit une stratégie combinée visant à localiser et classifier les individus dans le cadre d'une application de train autonome, avec l'objectif de distinguer les travailleurs des passagers. Cette stratégie s'articule autour d'une démarche en deux phases : premièrement, le recours au modèle YOLOV3 [124] préalablement entraîné sur le jeu de données MS-COCO [128] permet de détecter les individus. Deuxièmement, un SVM [137] binaire, entraînée à la fois sur le jeu de données PETA [135] pour les passagers et sur un jeu de données spécifique de travailleurs ferroviaires vêtus de combinaisons oranges, est employée. Ce classifieur SVM [137] sert à distinguer les passagers des travailleurs en se basant sur les détections issues de YOLOV3 [124]. L'évaluation de cette méthode s'effectue sur un jeu de données spécifique comprenant 30 000 images de passagers et de travailleurs ferroviaires. Après une phase d'augmentation des données, le jeu de données s'élève à 60 000 images. Les expérimentations montrent que cette approche offre un taux de précision de 95% pour la détection à une vitesse de 40 images par seconde (FPS) et un taux de précision de 98,5% pour la classification. Les figures 44 et 45 montrent les résultats de détection avec la méthode proposée sur des images de scènes ferroviaires comportants des voyageurs et des travailleurs des chemins de fer.

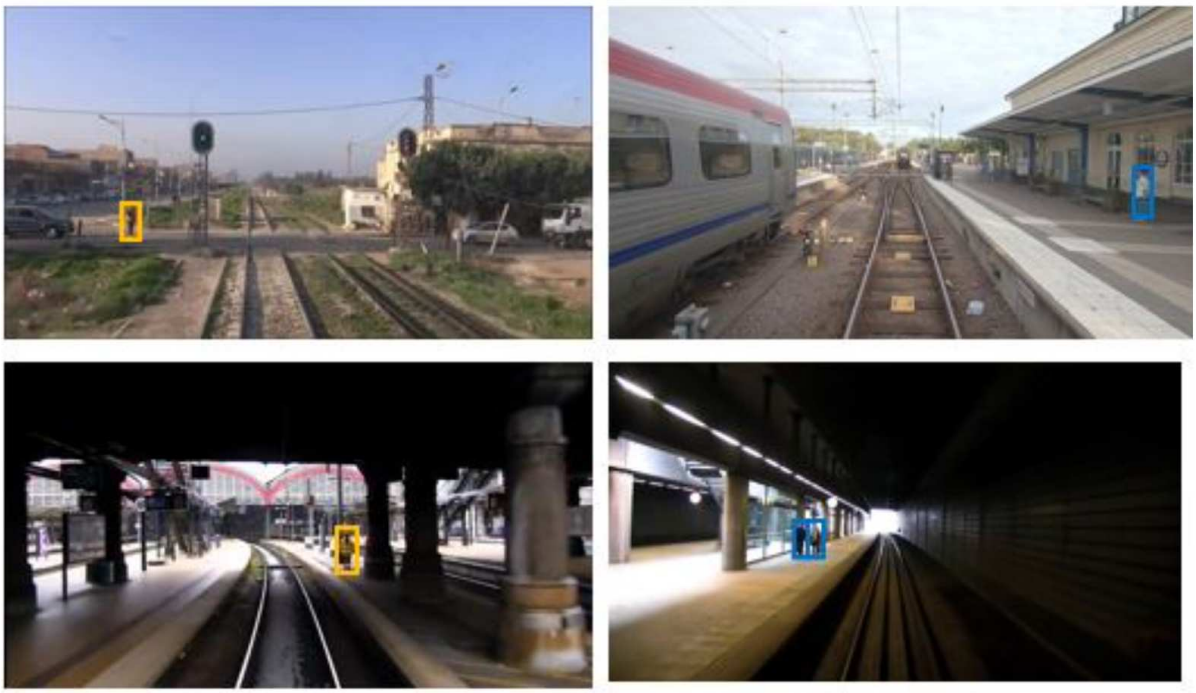

FIGURE 44: Résultat de détection de la méthode proposée sur des scènes ferroviaires avec des voyageurs [136]



FIGURE 45: Résultat de détection de la méthode proposée sur des scènes ferroviaires avec des voyageurs et des travailleurs [136]

Les auteurs de [209] décrivent une stratégie hybride pour détecter les voies ferrées en combinant une technique de traitement d'image et le modèle YOLOV4 [126] pré-entraîné sur MS-COCO [128] pour les obstacles. Cette méthode mise sur la détection de voies en temps réel grâce à la vision par ordinateur, en s'appuyant sur l'algorithme de Canny [210] pour la détection des contours et la transformation de Hough probabiliste [211]. Une technique de fenêtre glissante, spécialement adaptée pour traquer les inclinaisons des voies, utilise des fenêtres rectangulaires couplées à une courbe de Bézier [212]. L'approche affiche une exactitude de 92,08% sur 8 vidéos issues de YouTube, atteignant 43 FPS pour une résolution de 1920x1080. La figure 46 montre les résultats obtenus.



FIGURE 46: Figure montrant les résultats de l'approche utilisée [209]

3.2.2 Détection par apprentissage non-supervisé

L'étude présentée dans [142] introduit un framework pour la détection d'obstacles sur les voies ferrées, en combinant des techniques de vision par apprentissage supervisé et non supervisé avec des images RGB et thermiques. Ce système repose sur un drone équipé d'un autoencodeur convolutionnel (CAE) [147] ainsi que d'un classifieur binaire pour la détection. Les expérimentations, réalisées sur un jeu de données nocturnes, attestent de la précision de cette méthode. Dans le cadre de leurs perspectives de recherche, les auteurs envisagent d'intégrer des données stéréo, d'utiliser une carte embarquée sur GPU, et de considérer les conditions climatiques défavorables. Il convient de noter que la méthode est conçue principalement pour l'inspection lorsque le train est à l'arrêt. Pour mener à bien leurs recherches, un jeu de données spécifique a été élaboré en se focalisant sur les anomalies nocturnes dans le contexte ferroviaire grâce à un drone spécialisé. Des capteurs haut de gamme, y compris la caméra Flir Boson 640 [143], la caméra stéréo Zed [144], et la caméra Basler acA800-510uc [145], ont été mobilisés. Sur le plan technique, le CAE [147] est utilisé pour extraire les caractéristiques des images standards (sans obstacles). Une reconstruction des images initiales est ensuite réalisée, suivie du calcul des différences absolues et de gradient, qui alimentent un réseau de neurones convolutifs (CNN) [121] dédié à la classification. Ce dernier est en mesure de discerner 10 types d'anomalies. Les performances atteintes sont de 96,6% de précision pour les données thermiques et de 81,1% pour les données RGB. Une extension approfondie de cette étude est disponible dans [146], fournissant des informations complémentaires sur le jeu de données baptisé "Vesuvio". La figure 47 montre un aperçu des différents objets à détecter. La figure 48 montre les résultats de détection par la composante non-supervisée ainsi que l'utilisation de la composante supervisée pour quantifier les anomalies détectées.

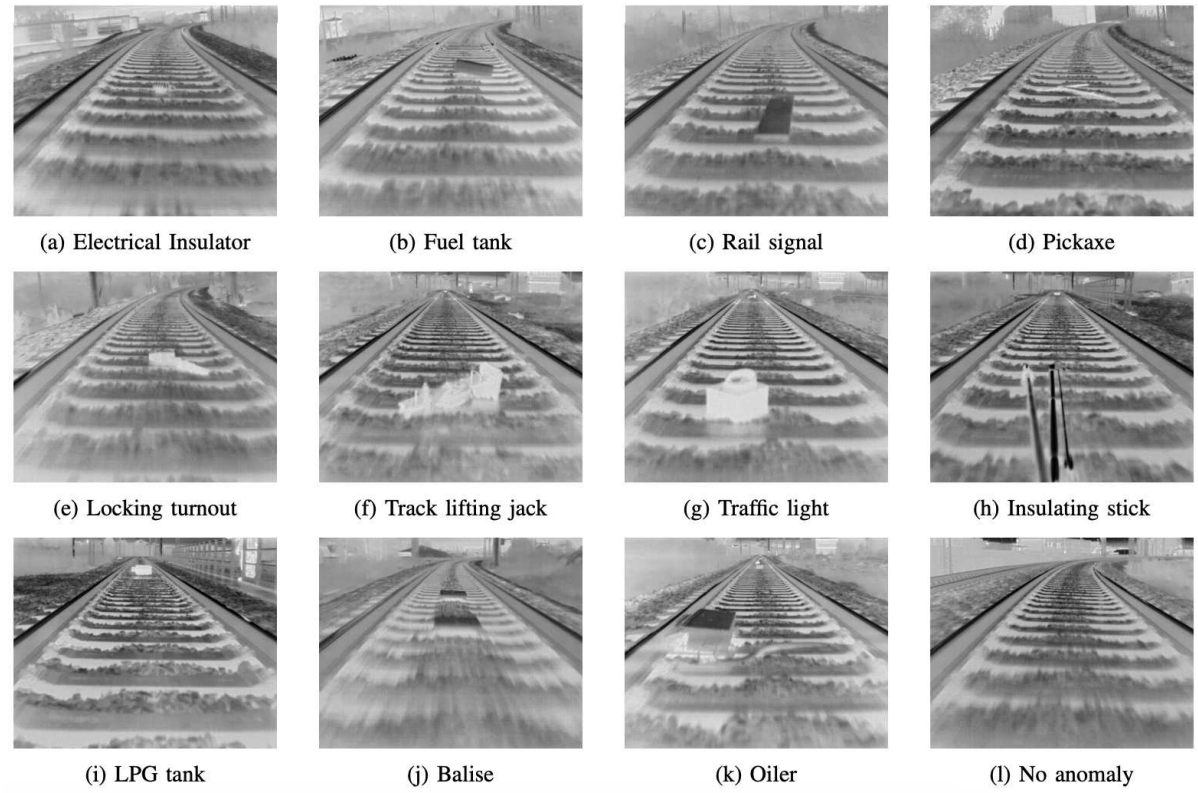

FIGURE 47: Aperçu des différentes classes à détecter par la composante supervisée du travail proposé [142]

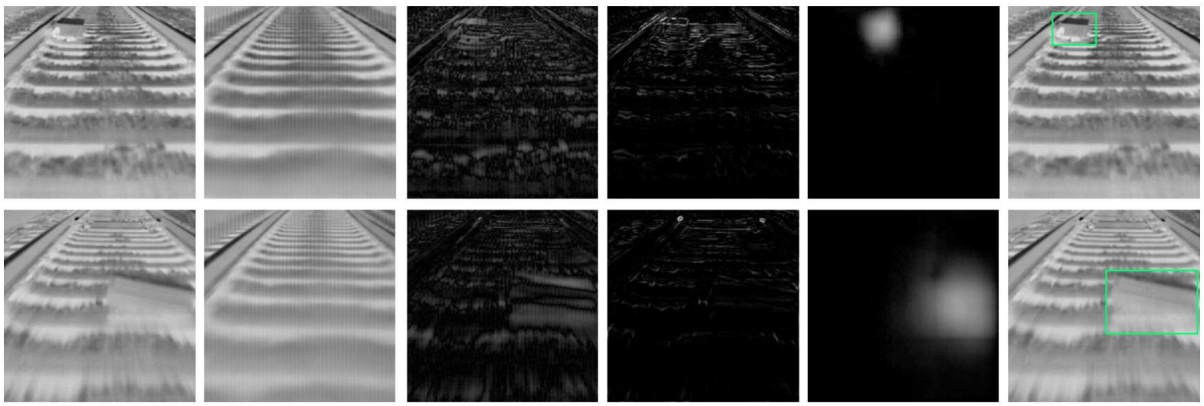

FIGURE 48: Résultat de détection du travail proposé [142]

La méthode décrite dans [151] introduit une approche pour la détection d'anomalies sur les voies ferrées. En utilisant 7,500 images provenant du jeu de données RailSem19 [218], cette étude se concentre sur des fragments d'images plutôt que sur une segmentation globale traditionnelle. Par le biais d'un réseau neuronal peu profond, cette méthode segmente minutieusement les régions ferroviaires en mettant l'accent sur leurs motifs répétitifs, ce qui minimise les prédictions inexactes. Une caractéristique distinctive de cette approche est sa capacité à simuler des images sans obstacles, permettant de détecter des anomalies en identifiant les écarts par rapport à l'environnement ferroviaire standard. En complé-

ment des images RailSem19 [218] qui constituent 7500 images ferroviaire, 1200k images d'ImageNet ont été incorporées, et l'évaluation a été réalisée sur un ensemble de RailSem19 [218] enrichi par PascalVOC [130], comprenant 7,142 images. Comparée à des méthodes de référence comme DeeplabV3 [176] et deux autoencodeurs (avec deux fonctions de pertes MSE et SSIM), la méthode présentée a prouvé sa valeur avec un AUROC dépassant 0,926 et un score F1 de 0,863. L'étude a également mis en évidence des défis associés à des méthodes comme DeeplabV3 [176], en particulier pour détecter des obstacles de petite taille ou de couleurs semblables. La figure 49 montre un aperçu de la méthode proposée tandis que la figure 50 montre les résultats de la méthode en comparaison avec d'autres méthodes.

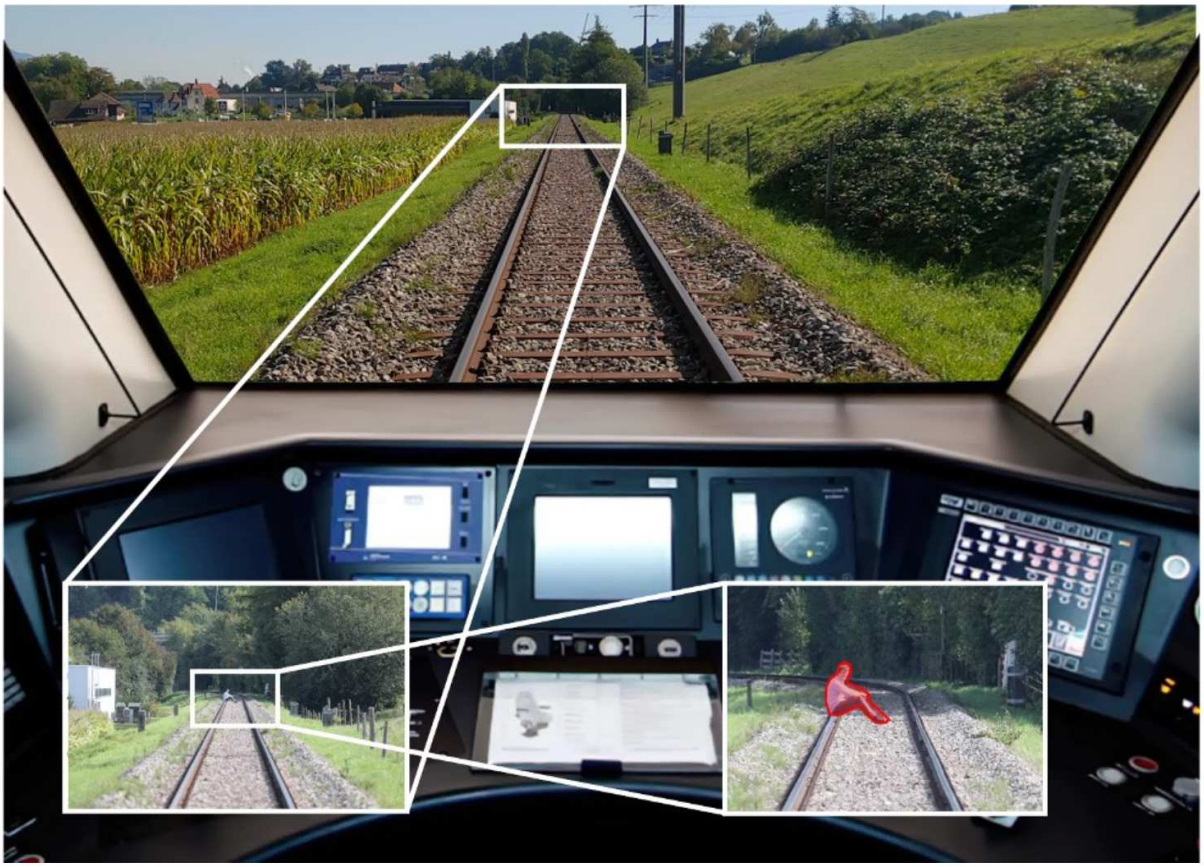

FIGURE 49: Aperçu de la méthode décrite dans [151]

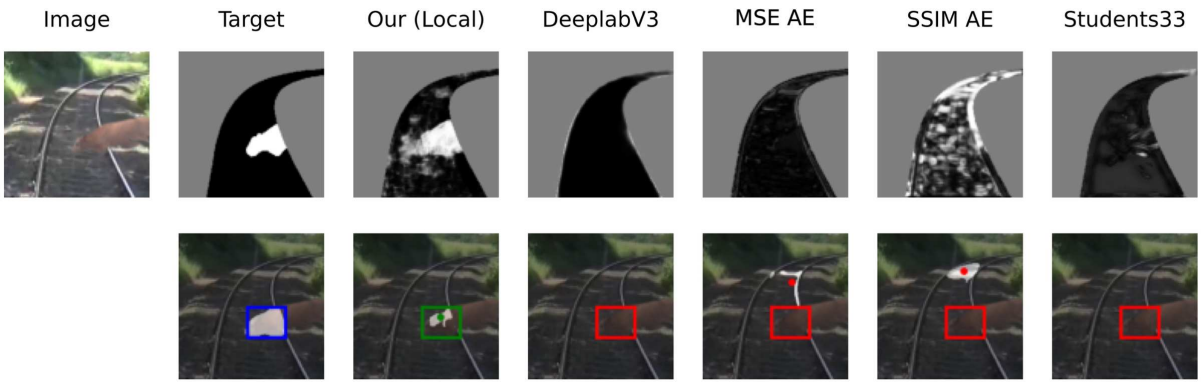

FIGURE 50: Image exemple où l'approche décrite dans [151] détecte l'obstacle correctement contrairement aux méthodes de base. La première ligne montre les cartes de segmentation prédites : le gris pour les zones externes, le noir pour la voie et le blanc pour la détection d'anomalies. La seconde ligne indique la localisation de l'anomalie

L'article de [150] introduit une stratégie non supervisée de détection des intrusions pour les chemins de fer à grande vitesse. Cette méthode repose sur des réseaux génératifs profonds couplés à des modèles d'auto-régression, afin de pallier le manque d'échantillons d'intrusion. Le modèle présenté, AnoVQVAE, fusionne un réseau VQ-VAE [153] avec un réseau PixelCNN [154], permettant l'exploitation de techniques de détection d'anomalies axées sur la reconstruction et la vraisemblance. L'entraînement du VQ-VAE [153] se fait exclusivement avec des images normales, tandis que le PixelCNN [154] se forme sur l'espace latent du VQ-VAE [153]. Durant la phase de détection, l'erreur de reconstruction associée à la vraisemblance de la variable latente est employée pour qualifier une image d'entrée comme normale ou anormale. L'enjeu majeur réside dans la capacité à appréhender adéquatement la distribution probabiliste des échantillons normaux tout en écartant les échantillons anormaux. Les auteurs dévoilent également RailAnomaly, un ensemble de données inédit, englobant huit scènes distinctes de chemins de fer à grande vitesse, et regroupant entre 8 400 à 17 600 échantillons normaux et de 3 200 à 4 000 échantillons anormaux. Pour attester de la pertinence de leur approche, ils utilisent cet ensemble, qui est construit à partir d'images réelles de surveillance ferroviaire destinées aux techniques de détection d'objets. La distinction et la localisation des anomalies au sein de cet ensemble sont réalisées en mesurant l'écart entre la reconstruction et l'image d'entrée originale. Les résultats de cette détection, issus de l'approche générative non supervisée, sont illustrés dans la figure 51

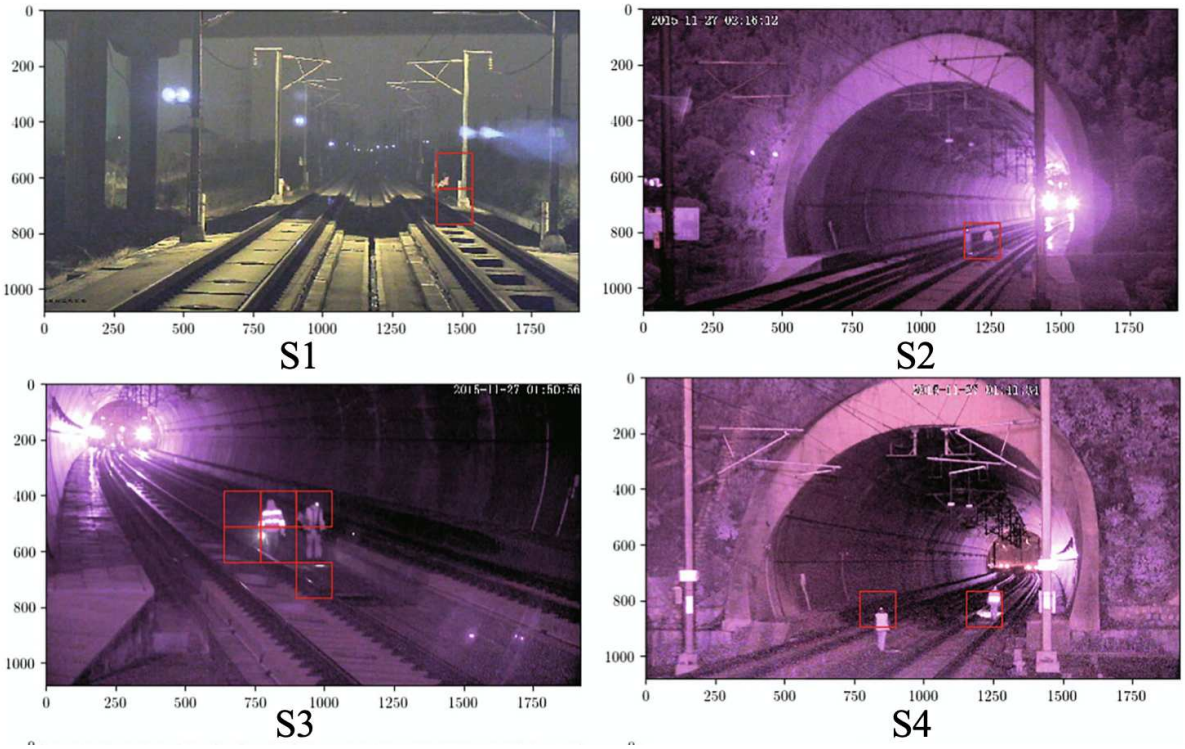

FIGURE 51: Résultat de détection du travail proposé [150]

Dans [156], une méthode basée sur une approche générative profonde est décrite pour détecter les objets sur les voies ferrées sans marquage préalable. La démarche repose principalement sur l'utilisation du modèle Autoencodeur (AE) [165] et d'un réseau discriminateur. Cette stratégie semi-supervisée se révèle performante pour localiser les anomalies. En comparaison avec d'autres architectures, notamment DFF-Net [167], Cascade R-CNN [168], RefineDet [169], OC-SVM [172], Isolation Forest [173] et AE [165], les performances obtenues sont de 85,66% en AUROC, 87,72% en AUPRC, 87,08% en Rappel, et 78,68% en F1-Score. La méthode affiche également un temps d'inférence rapide de 1,67 ms. Les résultats détaillés de cette approche sont présentés dans la figure 52. La figure 52 illustre les résultats obtenus avec leur approche générative non supervisée.

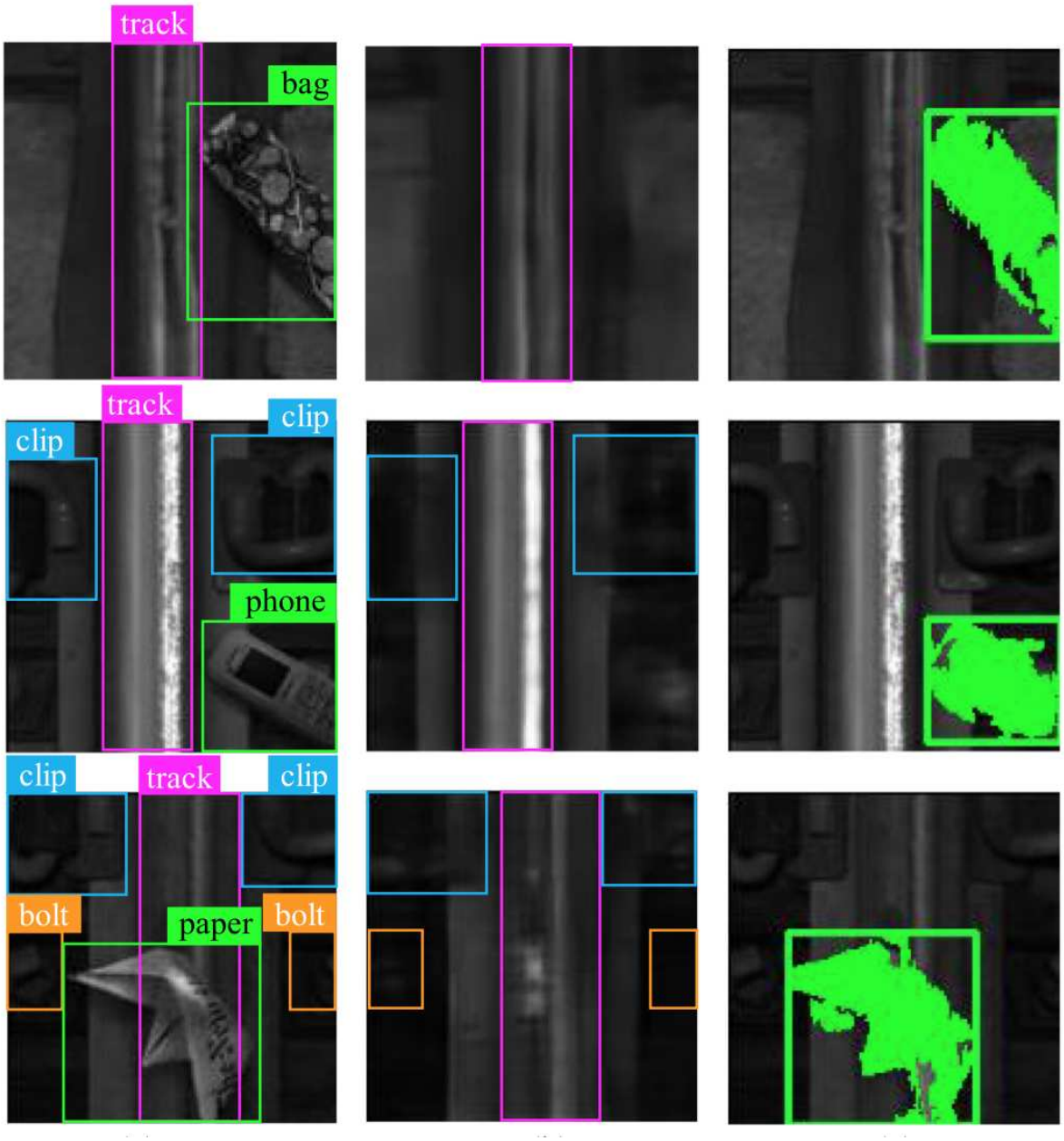

FIGURE 52: Résultat de détection du travail proposé [156]

3.3 Détection par d'autres approches

Les "autres approches", faisant principalement référence aux algorithmes traditionnels tels que le traitement d'images ou à l'utilisation extensive de capteurs physiques, ont été largement utilisées pour la détection d'obstacles sur les vouies ferrées. Ces approches intègrent des technologies comme les caméras RGB et infrarouges, le LiDAR, les radars, et d'autres capteurs. Bien que l'apprentissage profond offre souvent une précision et une fiabilité accrues, ces méthodes traditionnelles ont leur place. Contrairement à l'apprentissage profond qui s'adapte dynamiquement, ces approches s'appuient sur des règles et algorithmes prédéfinis, qui peuvent être limités face aux variations environnementales ou aux scénarios complexes [195]. Cependant, pour des applications spécifiques comme la détection d'objets stationnaires ou la surveillance de l'usure de l'infrastructure ferroviaire, ces méthodes traditionnelles peuvent s'avérer efficaces. Les capteurs peuvent détecter des propriétés physiques d'un obstacle, tandis que le traitement d'image peut reconnaître des caractéristiques visuelles, tels que la couleur ou la texture. La fusion de ces approches peut donc renforcer la précision et la fiabilité des systèmes de détection ferroviaire.

L'étude de [196] explore l'utilisation des capteurs, en particulier une caméra et un LiDAR, pour la détection des obstacles. La technique vise à repérer principalement les individus à une distance cible de 300 m afin de prévenir les accidents. La méthode proposée a montré qu'elle pouvait détecter une personne à 200 m la nuit avec 45% de précision. Pour évaluer cette méthode, des expérimentations ont été réalisées avec une caméra USB de haute résolution montée dans une cabine de locomotive. Les tests, basés sur 56 vidéos, ont montré un taux de détection dépassant 98% jusqu'à une distance de 300 m. Cependant, des capteurs supplémentaires, comme le radar à ondes millimétriques et le LiDAR, sont nécessaires en faible luminosité. En combinant le LiDAR et la caméra, l'étude a généré des données de nuage de points 3D. Malgré les variations d'éclairage, la détection reste efficace jusqu'à 300 m grâce à cette combinaison. La figure 53 montre les résultats de détection ainsi que la méthode proposée.

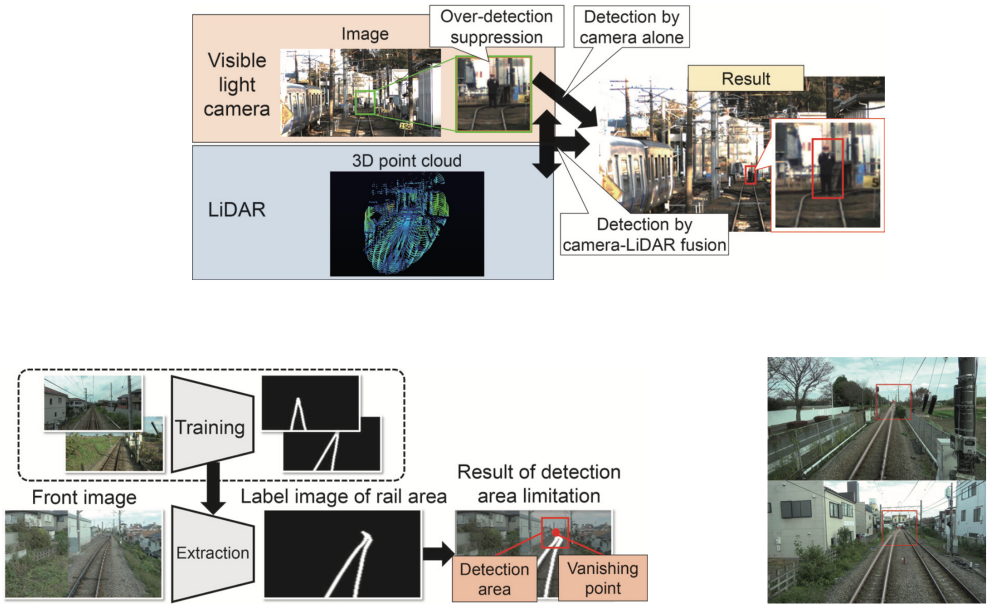

FIGURE 53: Description de la méthode utilisée et résultats de la détection du travail proposé [196]

Dans la poursuite de la conception d'un système innovant de détection d'obstacles, l'étude mentionnée dans [204] introduit LROD, une solution destinée à la détection d'obstacles à longue distance pour les dispositifs d'aide à la conduite ferroviaire. La démarche adoptée se base sur une caméra de surveillance fixe dédiée à la reconnaissance des voies ferrées et à la délimitation des zones d'intérêt potentiel. Un ensemble de capteurs dynamiques, constitué d'une caméra à focale étendue et d'un LiDAR 1D à haute portée, balaye la zone préalablement identifiée dans le but de détecter et de positionner les éventuels obstacles. La phase finale de cette procédure consiste en une technique de détection des anomalies, en l'occurrence les obstacles, appuyée sur le processus de reconstruction. Sur le plan matériel, le système se distingue par un mécanisme de pointage sur mesure doté d'une précision de 6 mdeg, permettant de repérer des objets jusqu'à une distance de plus de 1,5 km. La figure 54 montre le capteur de détection proposé.

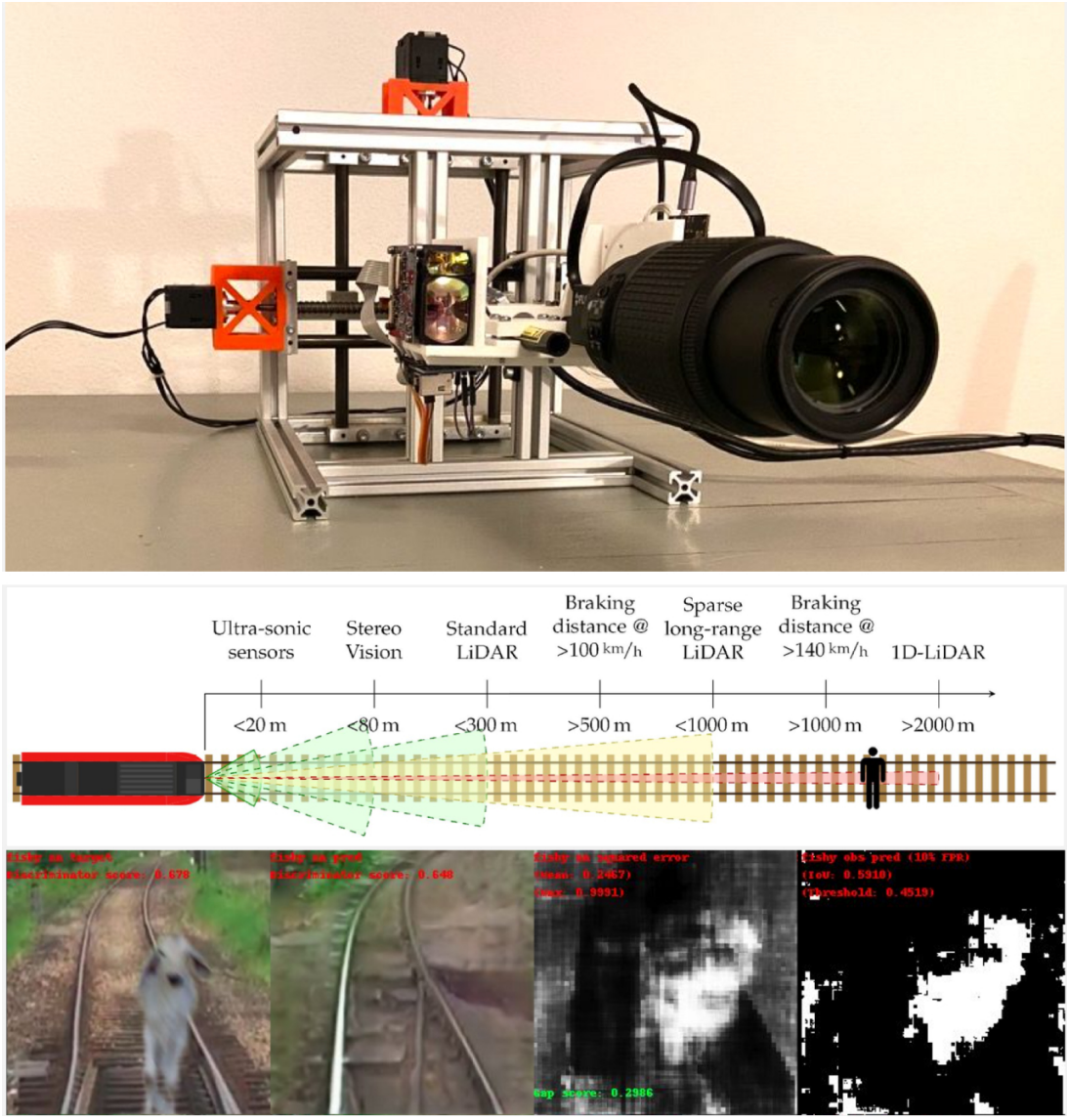

FIGURE 54: Figure du capteur de détection LROD [204]

L'article de [208] présente une méthode innovante pour la détection d'obstacles sur les voies ferrées en utilisant des capteurs IoT basiques. Cette initiative, en collaboration avec la compagnie ferroviaire malaisienne, propose un dispositif d'inspection visuelle pour la surveillance des voies et une solution IoT supplémentaire avec un affichage radar pour monitorer les obstacles. Trois capteurs ultrasoniques sont placés stratégiquement sur la locomotive et transmettent des données à un serveur cloud via Wifi. Les tests, bien que limités à des objets petits et simples, ont montré des résultats encourageants. Les informations sont affichées sous forme de radar, visualisant les obstacles dans une gamme de 0 à 40 cm. L'objectif est d'améliorer la sécurité des voies ferrées. L'affichage radar distingue trois scénarios : pas d'obstacles (vert), présence d'un obstacle (rouge avec position et distance) et plusieurs obstacles superposés (rouge). Malgré un taux d'erreur de 2,9% à 6,7%, l'étude n'élucide pas entièrement la nature des obstacles ou l'extension de cette méthode à des situations réelles plus complexes.

La figure 55 montre les résultats de détection par l'approche proposée dans le travail.

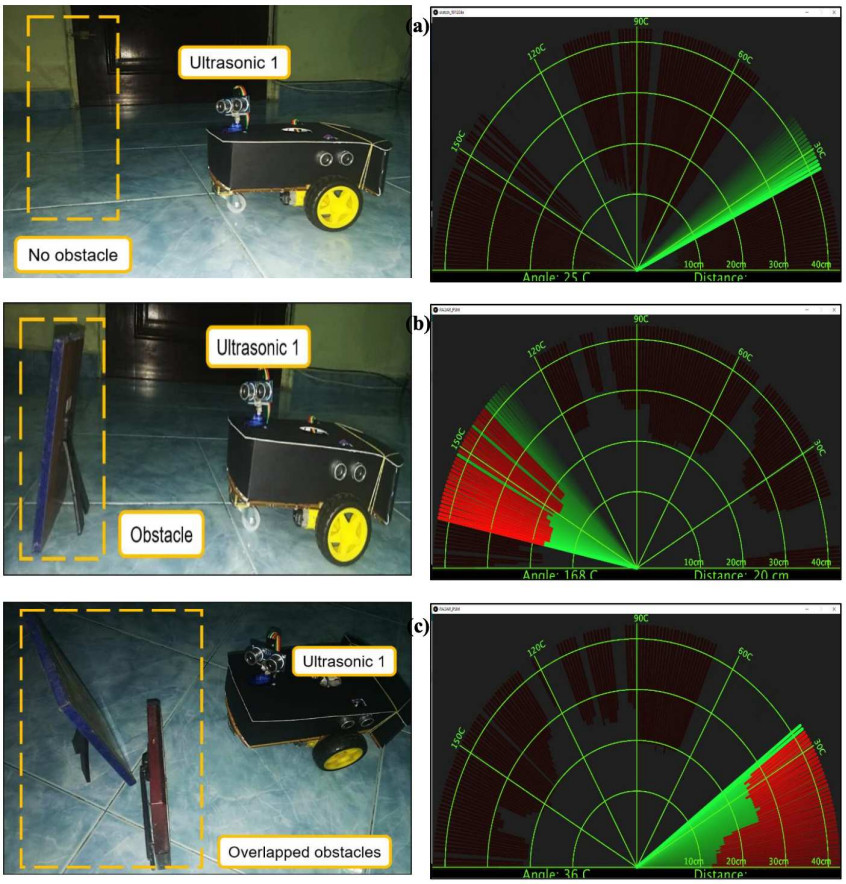

FIGURE 55: Description des résultats de la détection du travail proposé [208]

L'article présenté dans [213] détaille la conception et la mise en œuvre de TRINETRA, un système avancé pour la détection d'obstacles sur les voies ferrées, spécifiquement dans des scénarios de visibilité réduite. Le dispositif associe une caméra, un radar et un laser infrarouge pour identifier les obstacles et offrir une perspective élargie de la voie au conducteur. Les essais du prototype, effectués dans un environnement ferroviaire réel, ont confirmé son aptitude à détecter efficacement des obstacles en présence de brouillard. Envisagé comme une solution pour limiter les incidents majeurs sur les réseaux ferroviaires indiens durant la période hivernale, TRINETRA intègre des fonctionnalités telles que la signalisation automatique, des dispositifs anti-collision, des mécanismes de protection des trains et des systèmes d'alerte. Techniquement, le système incorpore une caméra bullet équipée d'un filtre infrarouge, un capteur CMOS de 1,2 mégapixels, et des lentilles à focale fixe allant de 3,5 mm à 100 mm, permettant des captures de qualité à une résolution jusqu'à 1920x1080. Il est également doté d'une caméra thermique, d'un radar pour la détection et d'un illuminateur laser infrarouge. Les résultats, partagés par les auteurs, mettent en évidence la capacité du dispositif à identifier des obstacles sur une plage allant de 1 km à 12,6 km, exploitant la synergie des composants intégrés à TRINETRA. La figure 56 montre le capteur proposé.

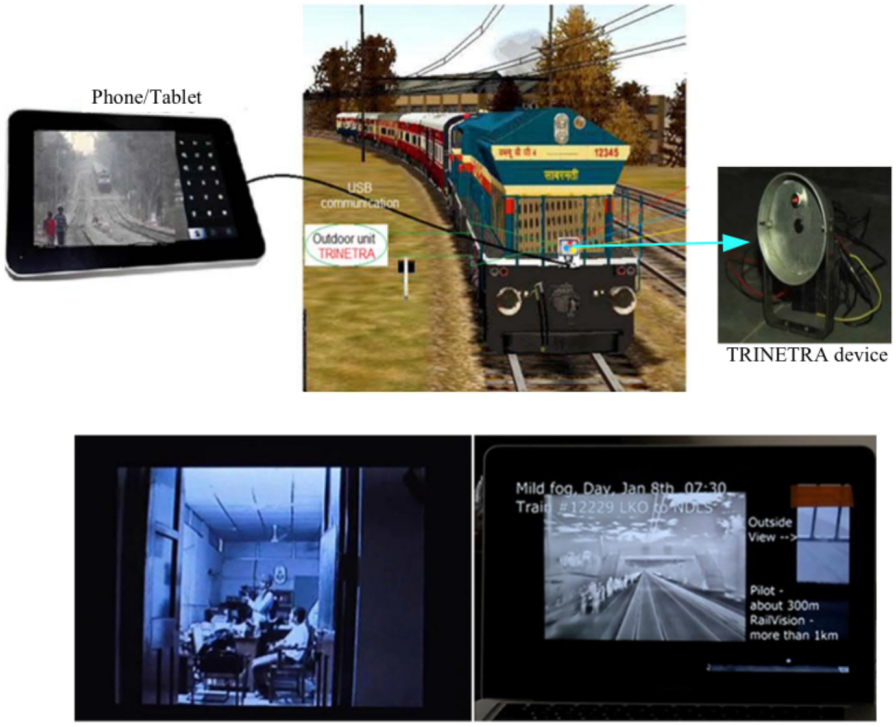

FIGURE 56: Figure de l'approche basée sur des capteurs physiques TRINETRA [213]

Dans l'étude de [197], les auteurs proposent une technique de détection d'obstacles basée sur la comparaison d'une image capturée par une caméra frontale d'un train avec une image de référence, en utilisant une soustraction d'arrière-plan adaptée aux caméras en mouvement. L'originalité réside dans une nouvelle métrique pour aligner des séquences d'images avec une base réduite, et une stratégie de détection à distance basée sur l'enregistrement point par point et divers mécanismes de soustraction. L'approche implique un alignement double : temporel (comparant images actuelles et de référence) et spatial (via un enregistrement des images bien alignées). La déformation entre les images est calculée avec DeepFlow [198]. Le processus principal se concentre sur deux métriques déduites des images : la distance vectorielle normalisée (NVD) [199] et le filtre de portée radiale (RRF) [200]. Après seuillage, ces métriques isolent des obstacles, et des techniques morphologiques éliminent les bruits. Les régions d'intérêt sont identifiées, évaluées selon les normes d'écartement des rails, et les zones non pertinentes sont éliminées. La détection combine ensuite les données de la NVD [199] et du RRF [200]. Les auteurs ont validé leur méthode en la comparant à d'autres techniques reposant uniquement sur la NVD [199] ou le RRF [200]. Leur approche a atteint une précision de 85,8%, mais présente des limites, en particulier dans des conditions défavorables pour la NVD [199] ou le RRF [200], influençant les résultats globaux. La figure 57 montre les résultats ainsi que les étapes suivies.

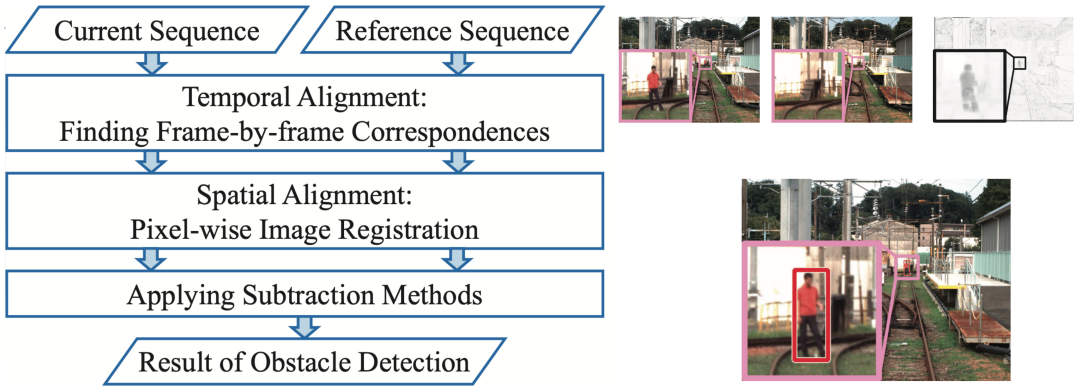

FIGURE 57: Figure montrant les résultats de l'approche utilisée [197]

Dans [205], une méthode de détection d'obstacles sur les voies ferrées est présentée, s'inspirant d'une technique d'identification des ballasts. Elle se base sur la cartographie de perspective inverse adaptative (IPM) pour atténuer les effets de perspective, favorisant ainsi la détection des voies et obstacles. Les contributions majeures sont : une technique de détection des bords des rails et des traverses, et un modèle de cartographie adapté aux voies ferrées. Ce modèle est capable d'estimer les paramètres de la caméra à partir de l'angle de lacet. L'algorithme se distingue par la détection différentielle des contours, et l'identification d'une région trapézoïdale pour la calibration, en se focalisant sur les traverses via un détecteur directionnel et un seuillage OTSU. La détection des rails est gérée distinctement. Les auteurs ont évalué la méthode sur des voies ferrées ballastées. Les taux d'erreur relevés sont de 10,4% (vision verticale) et 7,2% (vision horizontale). Les résultats sont illustrés dans la figure 58.

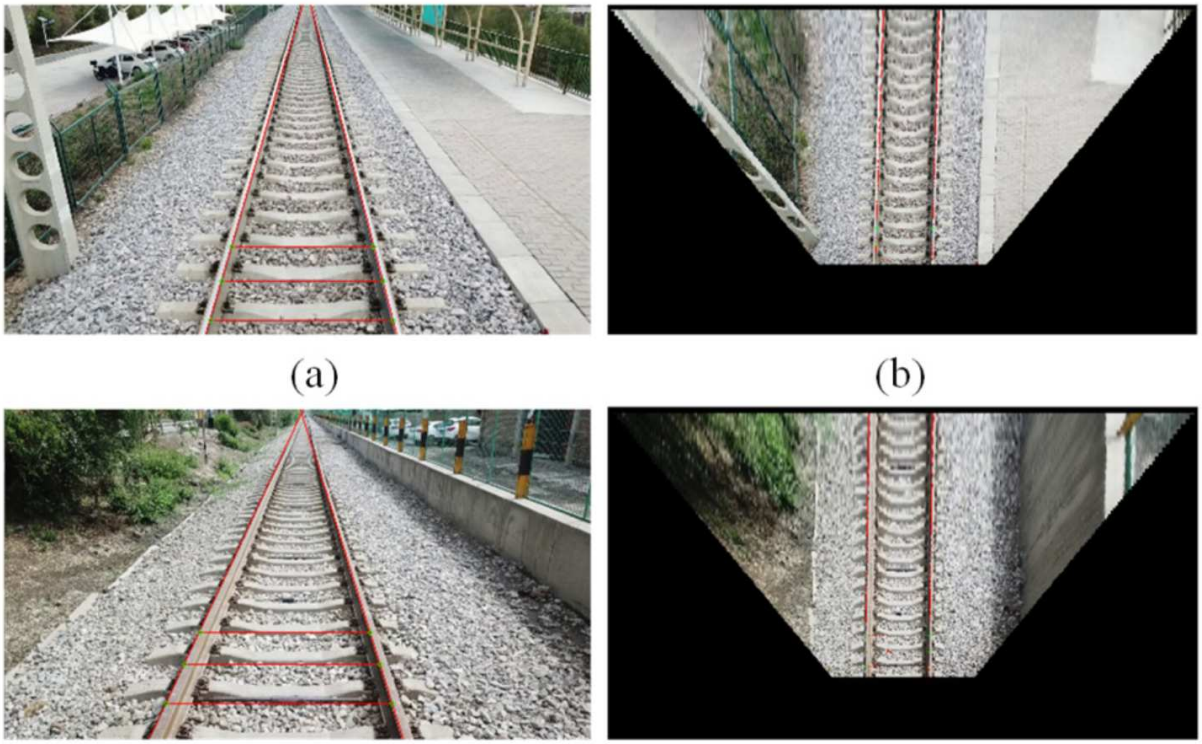

FIGURE 58: Figure montrant les résultats de l'approche utilisée [205]

Le travail décrit dans [201] se concentre sur la sécurité du transport ferroviaire urbain, qui est confronté à des obstacles de plus en plus fréquents en raison de l'augmentation du trafic de passagers. Pour résoudre ce problème, les auteurs ont utilisé la technologie de vision par ordinateur (traitement d'image) et le LiDAR pour surveiller en temps réel l'environnement des trains. Ils ont développé un dispositif qui collecte des images vidéo et des données de télémétrie de la voie, détecte les obstacles et évalue leur dangerosité. Les informations sont ensuite transmises au train pour contrôler son fonctionnement en fonction du niveau de danger. La méthode proposée dans ce travail se divise en deux étapes principales : La première étape se focalise sur la détection des rails de la voie à l'aide de l'algorithme de transformation de Hough [228] basé sur la détection de contours de Sobel. Cette étape permet de délimiter les régions d'intérêt sur la voie. La deuxième étape est axée sur la détection des obstacles au niveau de l'emprise ferroviaire. Elle utilise une méthode améliorée de différence d'images pour détecter les obstacles dans les régions d'intérêt définies précédemment. Cette méthode nécessite un arrière-plan initial pour mettre à jour l'arrière-plan en temps réel. L'utilisation de la vision par ordinateur et du LiDAR permet de détecter la position, la forme et la hauteur des obstacles, ainsi que d'évaluer leur impact sur le fonctionnement des trains. En fusionnant ces deux méthodes, la précision de détection des obstacles est améliorée. Les informations de détection sont ensuite transmises au système de signalisation du train, qui peut ajuster automatiquement le fonctionnement du train en fonction du niveau de danger des obstacles détectés. Cette approche contribue ainsi à garantir la sécurité du système de transport ferroviaire

urbain. La figure Figure 59 montre les résultats de détection par l'approche proposée.

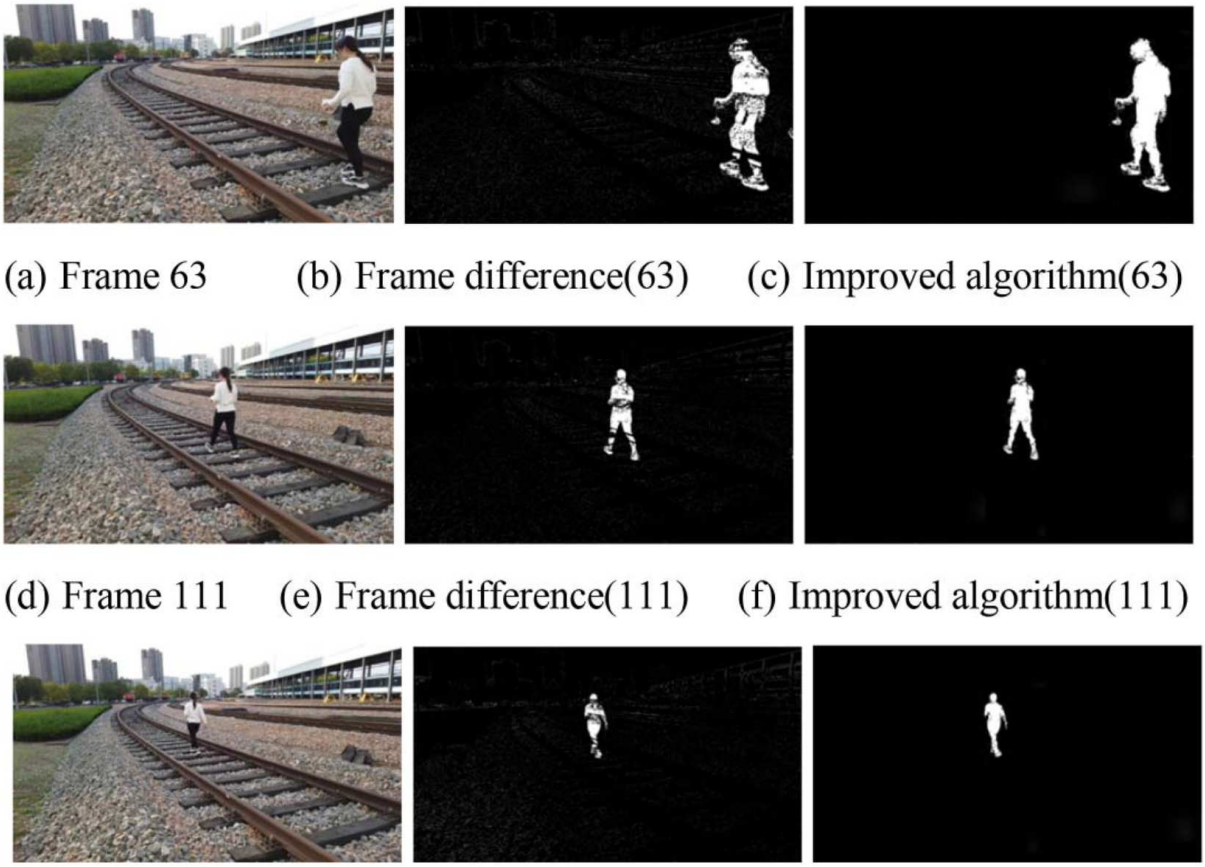

FIGURE 59: Figure montrant les résultats de l'approche utilisée [201]

3.4 Conclusion

Dans ce chapitre, une étude des travaux récents sur la détection des obstacles ferroviaires à l'aide de l'apprentissage profond et des méthodes classiques comme le traitement d'image et l'utilisation des capteurs a été réalisée. Cette étude a montré que malgré les avancées significatives dans le domaine de l'IA et des technologies connexes, des opportunités d'amélioration demeurent en matière de détection des obstacles ferroviaires. L'accent est mis sur l'utilisation d'algorithmes d'apprentissage profond, notamment les approches supervisées et non supervisées, pour la détection des obstacles ferroviaires. Il ressort de cette analyse que la détection des obstacles ferroviaires a été moins étudiée que celle des obstacles pour les voitures autonomes. Toutefois, cette étude apporte un éclairage sur les progrès récents en matière de détection d'obstacles ferroviaires, offrant des perspectives aux acteurs du domaine de la sécurité et de l'automatisation ferroviaire. Les perspectives obtenues pourraient guider le développement de systèmes de détection plus efficaces, renforçant ainsi la sécurité des systèmes ferroviaires.

3.4.1 Synthétisation des travaux

Dans cette section, une synthèse structurée des recherches antérieures portant sur la détection des situations anormales dans l'environnement des trains de fret autonomes est proposée. Présentée sous forme de tableaux, cette synthèse met en lumière trois approches principales : supervisées, non-supervisées et classiques. Suite à cette revue, il ressort que les approches non-supervisées sont moins couramment abordées. Tout en reconnaissant les contributions des travaux existants, cette recherche vise à explorer ce segment moins étudié. En considérant la détection d'obstacles et le monitoring de l'environnement du train de fret autonome comme un enjeu de détection d'anomalies, l'ambition est de compléter les connaissances en apportant une perspective additionnelle à la littérature existante.

Les tableaux de synthèse, inclus dans cette partie, sont organisés comme suit : le tableau 3.1 liste les travaux relatifs aux approches supervisées, le tableau 3.2 se focalise sur les approches non-supervisées, et le tableau 3.3 aborde les approches classiques, en particulier celles s'appuyant sur le traitement d'images ou l'emploi de capteurs.

TABLE 3.1: Synthèse des travaux mettant en jeu des approches supervisées.

Article	**Ensemble de données**	**Capteur**	**Modèle**	**Contribution**	**Évaluation**	**Commentaire**
Kafetzis et al. 2020 [106]	4K images de 12 classes	Caméra UAV	Faster RCNN, Inception v2, ResNet Atrous	Deux versions de Faster RCNN	51%-61% mAP	Dataset peu documenté
Tao et al. 2020 [109]	9k images de 7 classes	Caméra d'entraînement	FE-SSD (SSD amélioré)	Amélioration de la détection des petits objets	89.5% à 38fps (taille d'entrée 320x320)	Manque de données qualitatives sur le dataset
Haixia et al. 2022 [117]	Dataset synthétique de 2.4k images	Vidéo de test en 3D	RepVGG, ResNet50 et DarkNet53	Approche de détection multitâche avec RepVGG	69.6% mAP à 52 FPS	N/A
Mingyang et al. 2018 [120]	20k images pour reconnaître 3 classes	N/A	Faster-RCNN	Faster-RCNN utilisant des unités résiduelles	89.06% mAP	Manque de données, méthodologie et résultats insuffisants.
Deqiang et al. 2021 [122]	Données personnalisées, KITTI , MS-COCO	Caméras binoculaires, caméra RGB, d'un radar et d'un LiDAR	YOLOV4 amélioré	Nouveau backbone pour YOLOV4 "D-CSPDarknet"	93% mAP toutes les 0.139s	N/A
Song et al. 2021 [131]	MS-COCO pour l'entraînement	N/A	YOLOV3	Utilisation de l'edge computing pour la détection d'obstacles	N/A	Axé sur l'edge computing, manque d'informations sur la partie détection
Rampriya et al. 2022 [118]	RROD : Rail Obstacle Detection, 2k images, données aériennes de 6 classes	Caméra embarquée sur UAV : Sony DSC-RX1RM2 1920x1080	Center Hourglass, SSD, Faster RCNN, YOLOV3	Utilisation de UAV pour détecter les obstacles et plusieurs détecteurs d'objets pour la comparaison	96.75%, 84.75% et 83.75% mAP respectivement avec SSD MobileNet V2, Faster RCNN et SSD	N/A
Weixun et al. 2022 [132]	Dataset personnalisé de 3k images pour la classification et la détection de 3 classes	N/A	Classificateur personnalisé + YOLOV3	Nouvelle détection en deux étapes : classificateur pour intrusions, YOLOV3 pour obstacles	96.88% à 72.18 FPS	Le premier travail sur les obstacles ferroviaires à inclure des détecteurs ViT dans ses expériences
Ankur et al. 2020 [136]	MS-COCO, PETA, dataset personnalisé : 60k images de voyageurs et de travailleurs ferroviaires	N/A	YOLOV3, SVM	Nouvelle détection en deux étapes utilisant YOLOV3 + SVM	95% à 40 FPS pour le détecteur, 98.5% pour le classificateur	Contribution axée sur le monitoring de l'environnement des trains autonomes
Shreyas et al. 2022 [209]	MS-COCO et 8 vidéos YouTube	N/A	YOLOV4 [126]	Approche hybride basée sur les détecteurs d'objets et le traitement d'images	Précision 92,08%	Application du modèle YOLOV4 comme boîte noire [46]

TABLE 3.2: Synthèse des travaux mettant en jeu des approches non-supervisées.

Article	**Ensemble de données**	**Capteur**	**Modèle**	**Contribution**	**Évaluation**	**Commentaire**
Gasparini et al. 2021 [142] [146]	Vesuvio : 10k images RGB et thermiques	Drone ferroviaire équipé de différents capteurs : Flir Boson 6401, caméra stéréo Zed, Basler acA800-510uc3	CAE + CNN	Nouveau framework hybride pour l'apprentissage non supervisé utilisant à la fois CAE et CNN. Nouveau dataset appelé Vesuvio	Précision de 96,6%, 90,3% et 97% à 100 FPS pour la détection, la localisation et la classification respectivement	N/A
Wang et al. 2022 [150]	RailAnomaly : 8k-17k échantillons normaux et 3k-4k échantillons anormaux	N/A	VAE, VQ-VAE, PixelCNN	AnoVQVAE : un nouveau framework hybride utilisant à la fois des modèles génératifs non supervisés et des CNNs, nouveau dataset : RailAnomaly	Précision de 89,56% pour la classe des piétons, temps d'inférence 30 ms	N/A
Tiange et al. 2022 [156]	Ensemble de données : 6k images normales et 579 images anormales et normales pour la validation	N/A	AE	AE antagoniste génératif	85,66%, 87,72%, 87,08% et 78,68% en AUROC, AUPRC, rappel et score F1 respectivement, temps d'inférence 1,67 ms	Images de basse résolution. Les images près des voies sont plus adaptées à l'inspection
Brucker et al. 2023 [151]	RailSem19 : 7.5k images, PascalVOC : 7.1k images, ImageNet : 120k images	N/A	Réseau neuronal superficiel	Approche basée sur des patches d'images locaux pour détecter les anomalies. Comparaison avec d'autres méthodes courantes.	AUROC de 0,926 et score F1 de 0,863	Pas d'informations complémentaires sur le réseau utilisé. Pas d'informations supplémentaires sur la nature des images complémentaires utilisées

TABLE 3.3: Synthèse des travaux mettant en jeu des approches basées sur le traitement d'images et l'utilisation de capteurs physiques.

Article	**Basé sur les capteurs**	**Basé sur IP**	**Contribution**	**Évaluation**	**Commentaire**
Kageyama et al. 2022 [196]	✓	X	Fusion LiDAR et caméra pour la détection	Taux de détection de 45% la nuit, jusqu'à 90% en basse luminosité	YOLOV4 pour la détection. Manque d'informations sur les résultats expérimentaux
Mukojima et al. 2016 [197]	X	✓	Soustraction du background avec une caméra en mouvement	Taux de détection de 85,8%	Testé uniquement sur un piéton et une boîte
Tienwen et al. 2021 [201]	✓	✓	Détection de voie ferrée pour la détection d'obstacle utilisant plusieurs capteurs physiques et des techniques de traitement d'images	N/A	Manque d'informations sur les résultats et les obstacles
Cornelius et al. 2022 [204]	✓	X	Système de détection basé sur LiDAR	N/A	Le travail est un poster. Manque d'informations pour évaluer la qualité du matériel proposé
Junting et al. 2023 [205]	X	✓	Méthode basée sur l'IPM pour la détection de voies ferrées ballastées	Les taux d'erreur varient entre 2,6% et 6,7%	Aucune autre information n'a été partagée sur l'extension de cette méthode pour détecter les obstacles
Aziz et al. 2020 [208]	✓	X	Détection d'obstacles ferroviaires basée sur les capteurs	Les taux d'erreur varient entre 2,9% et 6,7% à différents angles et distances	Aucune autre information n'a été partagée sur la méthode de détection et les obstacles. Utilisé seulement dans un cas d'utilisation miniature
Kumar et al. 2022 [213]	✓	X	TRINETRA : Un système de vision hardware pour la détection d'obstacles ferroviaires	Distance de détection entre 1 km et 12,6 km	Manque d'informations sur l'utilisation du système avec d'autres méthodes pour évaluer ses performances

Chapitre 4

Détection d'obstacles ferroviaires sur la voie principale

Avec l'évolution constante de la technologie autonome, une détection d'obstacles fiable et efficace s'impose, notamment pour les trains autonomes. Dans le cadre de ce chapitre, nous explorons une contribution méthodologique à la détection d'obstacles sur la voie de circulation ferroviaire en utilisant une approche non-supervisée. Contrairement aux systèmes de conduite autonome habituels qui s'appuient sur des modèles supervisés ou des méthodes de traitement d'images classiques pour identifier des obstacles spécifiques, la détection d'obstacles ferroviaires doit être fonctionnelle en présence de dangers imprévisible. Nous proposons une étude exploratoire basée sur des autoencodeurs convolutionnels (CAEs) [147] pour améliorer la robustesse des systèmes de détection d'obstacles ferroviaires. Nous générons 240 modèles dans notre étude suivants trois axes de composants intrinsèques : Les fonctions d'activations, les fonctions de pertes ainsi que les optimiseurs utilisés. Nous abordons la limitation de l'utilisation de seuils fixes pour évaluer les performances du modèle, et proposons une méthodologie basée sur la prise de décision multicritère (MCDM). De plus, nous introduisons une nouvelle mesure, le score d'écart (*gap-score*) pour évaluer l'efficacité des modèles. Nous utilisons aussi les réseaux antagonistes génératifs (GANs) [221] pour créer des images synthétiques d'obstacles afin de valider les performances des modèles. Nos résultats montrent que nos modèles peuvent produire un score d'écart moyen allant jusqu'à 68%, illustrant le potentiel de notre méthodologie pour renforcer la sécurité des trains autonomes.

4.1 Introduction

La détection d'obstacles reste l'un des défis les plus complexes et critiques dans le domaine de la vision par ordinateur. Des voitures autonomes [216] [215] aux trains autonomes [228] [227] [229], la détection d'obstacles joue un rôle important pour améliorer la perception du système autonome. Cette tâche est de la plus haute priorité, surtout pour le monitoring de l'environnement dans les trains autonomes. Grâce à l'évolution constante des méthodes d'apprentissage profond et à la puissance de calcul des systèmes embarqués, le monitoring de l'environnement dans ces systèmes est devenue relativement

facile à gérer.

Dans ce chapitre, nous exploitons les approches d'apprentissage profond afin de détecter les obstacles pour les applications ferroviaires. Nous nous concentrons uniquement sur le niveau de la voie en utilisant des caméras RGB montées sur le train. Étant donné le constat que les recherches sur la détection d'obstacles ferroviaires à l'aide de méthodes d'apprentissage profond sont encore peu nombreuses, et que la majorité des études existantes privilégient des approches supervisées telles que les détecteurs d'objets [107, 112, 126] ou les techniques de segmentation [217], notre démarche scientifique s'oriente vers l'exploration de méthodes d'apprentissage non supervisé. Plus précisément, nous envisageons l'emploi d'autoencodeurs convolutionnels comme une alternative potentielle pour aborder ce défi. Ce choix est motivé par trois facteurs : Premièrement, une rareté de travaux exploitant des méthodes non supervisées dans la détection d'obstacles ferroviaires est remarquable. Deuxièmement, l'utilisation des méthodes supervisées nécessite une connaissance préalable des obstacles, ce qui implique d'énumérer tous les obstacles possibles que nous pourrions rencontrer et de les étiqueter pour créer des classes, ce qui est presque impossible dans un scénario réel. Troisièmement, les ensembles de données du domaine ferroviaire décrivant des obstacles réels sont rares voire inexistants, en revanche, les données normales sans obstacles sont disponibles.

Nous proposons dans ce chapitre une étude exploratoire utilisant un autoencodeur convolutionnel pour discriminer, au niveau des images, entre les images normales et anormales en utilisant un score d'erreur. Dans notre étude, nous nous concentrons principalement sur les trois composantes suivantes : les fonctions d'activation, les fonctions de perte et les optimiseurs afin de trouver le meilleur modèle capable de détecter des anomalies au niveau des voies ferrées. Dans notre travail, nous générons 240 modèles suivant la combinaison de 5 fonctions de perte, 8 fonctions d'activation et 6 configurations de 3 optimiseurs différents. Nous avons utilisé l'architecture décrite dans la figure 60 pour mesurer leur influence sur l'amélioration des résultats et aussi pour explorer quelle configuration donne les meilleurs résultats. Nous proposons également une méthode pour classer tous les modèles du meilleur au pire en introduisant la notion de *gap-score*. Le gap-score est une métrique basée sur le pourcentage de la différence entre les reconstructions d'images sans obstacles et les reconstructions des mêmes images avec obstacles. Le processus d'apprentissage a été réalisé à l'aide d'un sous-ensemble de railsem19 [218] constitué de Régions d'Intérêt (RoI) au niveau de la voie de circulation principale. Afin de classer tous les modèles suivant le gap-score, nous utilisons un ensemble de validation constitué d'images normales et des mêmes images avec des obstacles incrustés. Enfin, le processus d'évaluation des 240 modèles générés a été réalisé en utilisant la Technique de l'Ordre de Préférence par Similarité à la Solution Idéale (TOPSIS) [148], un algorithme d'aide à la décision multi-critères. Les critères utilisés sont le gap-score moyen sur toutes les images normales et le nombre de gap-score positifs pour chaque image normale. De

cette façon, nous évitons de nous reposer sur des seuils arbitraires pour juger si une entrée est anormale ou non.

Le modèle le plus performant a été testé sur un scénario du monde réel impliquant un obstacle concret. Les résultats montrent un gap-score élevé pour les images avec l'obstacle et un gap-score faible pour les images normales sans obstacles. Une analyse complète des résultats de ce contexte est présentée dans la section 4.4.4.2.

4.2 Architecture proposée

4.2.1 Autoencodeur convolutionnel

Nous utilisons un autoencodeur convolutif où seules des données non étiquetées normales sont nécessaires pour extraire des connaissances sans aucune étiquette. En termes de détection d'anomalies, le modèle est entraîné uniquement sur des entrées saines. Étant donné une entrée x, le modèle tente de l'encoder en la comprimant pour en extraire ses données latentes puis la décode en la comparant à l'entrée originale, générant ainsi sa reconstruction x'. L'autoencodeur est divisé en trois composantes :

Encodeur : L'encodeur mappe un échantillon d'entrée x vers la couche goulot z.

Couche goulot : La couche goulot z stocke les représentations latentes de faible dimension pour chaque échantillon x.

Décodeur : Le décodeur mappe les données de z pour générer une reconstruction x' de l'entrée x.

Nous utilisons l'algorithme de rétropropagation pour minimiser la différence entre les données d'entrée x et sa reconstruction x' conjointement pour l'encodeur et le décodeur.

$$\arg\min_{\omega,\theta} \mathcal{L}(x, x') \tag{4.1}$$

Où ω et θ sont respectivement les paramètres à optimiser pour l'encodeur et le décodeur. $\mathcal{L}$ est la fonction de perte utilisée.

La figure ci-dessous 60 illustre l'architecture de l'autoencodeur convolutionnel employé pour cette étude. Cette structure est composée de 6 couches de convolution formant l'encodeur et 6 couches de déconvolution constituant le décodeur. Elle comporte également des opérations de stride et d'upsampling. Au cœur de cette architecture, un espace latent de dimensions 4x4 est prévu pour résumer efficacement les informations essentielles de l'image d'entrée de 256x256 pixels. L'organisation de cette architecture a été conçue pour trouver un compromis entre une réduction dimensionnelle efficace et la conservation des informations cruciales pour la reconstitution de l'image d'origine.

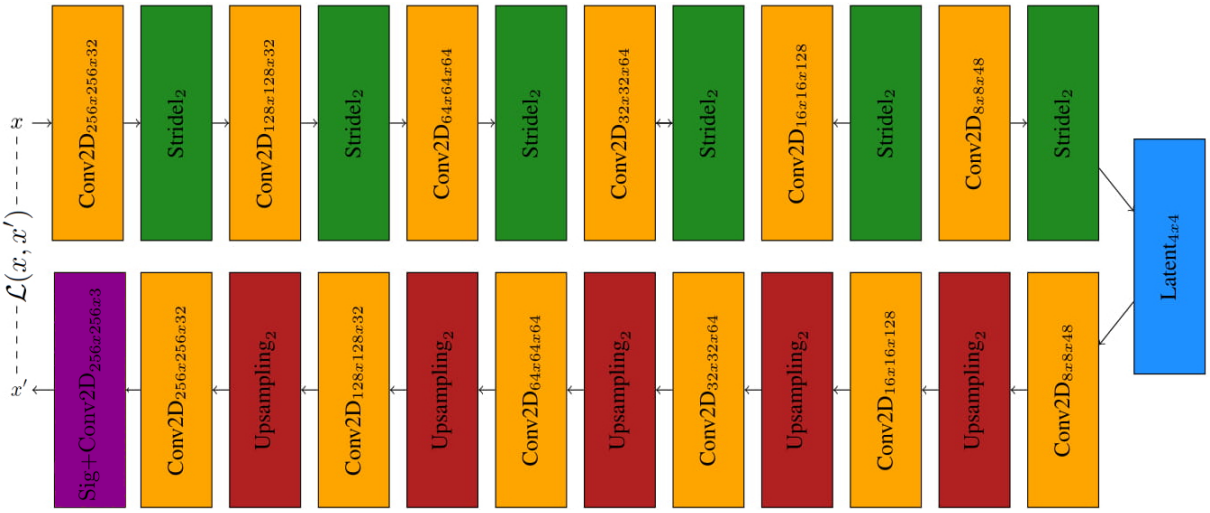

FIGURE 60: Description de l'architecture de l'autoencodeur convolutionnel utilisée dans cette étude exploratoire

4.2.2 Etude exploratoire

4.2.2.1 Exploration des fonctions de pertes

Dans cette sous-section, nous présentons les différentes fonctions de perte que nous considérons dans notre comparaison : l'erreur quadratique moyenne (MSE) [231] est une fonction de perte pixel par pixel mesurant la différence au carré entre la vérité terrain et les étiquettes prédites. L'erreur absolue moyenne (MAE) [231] est une fonction de perte pixel par pixel mesurant la différence absolue entre la vérité terrain et les étiquettes prédites. Le rapport signal/bruit de crête (PSNR) [232] est une mesure de la qualité d'image utilisée pour comparer la qualité de deux images. Souvent, la vérité terrain est appelée signal et la prédiction est appelée bruit. La similarité structurale (SSIM) [232, 233] est une mesure de qualité basée sur les patchs utilisée pour mesurer la similarité entre deux images. Au lieu d'utiliser des méthodes traditionnelles de sommation d'erreurs, cette mesure est conçue en modélisant toute distorsion d'image comme une combinaison de trois facteurs que sont la structure, la luminance et le contraste. La similarité structurale multi-échelle (MS-SSIM) [233] est une variante de SSIM. La MS-SSIM prend en compte de nombreux niveaux de résolution et de distorsion et peut être plus robuste par rapport aux variations des conditions de visualisation. On résume les fonctions de pertes utilisées dans cette étude exploratoire comme suit :

Erreur quadratique moyenne (MSE) : L'erreur quadratique moyenne entre deux images x et y avec N pixels peut être définie comme suit :

$$\text{MSE(x, y)} = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2 \tag{4.2}$$

Indice de similitude structurelle (SSIM) : L'indice de similitude structurelle entre deux

images x et y peut être défini comme suit :

$$\text{SSIM(x,y)} = \frac{(2\mu_x\mu_y + c_1) + (2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{4.3}$$

Dans l'Indice de Similitude Structurelle (SSIM), plusieurs paramètres sont utilisés pour calculer la similarité entre deux images. Ces paramètres incluent :

μ_x et μ_y : Ce sont les valeurs moyennes des pixels des deux images x et y, respectivement. Ils capturent les informations de luminance.

σ_x^2 et σ_y^2 : Ce sont les variances des valeurs des pixels des deux images x et y, respectivement. Ils capturent les informations de contraste.

σ_{xy} : C'est la covariance des valeurs des pixels des deux images x et y. Elle capture les informations structurelles (corrélation).

c_1 et c_2 : Ce sont deux variables pour stabiliser la division avec un faible dénominateur ; elles sont données par $c_1 = (k_1 \cdot L)^2$ et $c_2 = (k_2 \cdot L)^2$, où L est la plage dynamique des valeurs des pixels (généralement c'est $2^{bits\ par\ pixel} - 1$), et $k_1 << 1$ et $k_2 << 1$ sont de petites constantes. L'article original sur SSIM suggère d'utiliser $k_1 = 0,01$ et $k_2 = 0,03$.

Dans le cadre de l'utilisation de l'indice de similarité structurelle (SSIM) comme fonction de perte, nous cherchons à minimiser cette fonction de perte. Cela signifie que nous devons prendre "1 - SSIM" pour que les valeurs basses indiquent une bonne similarité et que les valeurs élevées indiquent une mauvaise similarité. La nouvelle formule sera donc :

$$\text{L(x,y)} = 1 - \text{SSIM(x, y)} \tag{4.4}$$

où $L(x, y)$ est la fonction de perte entre deux images x et y.

L'indice de similarité structurelle multi-échelle (MS-SSIM) est une méthode permettant de comparer la similarité de deux images. Il est une extension de l'indice de similarité structurelle (SSIM) et est conçu pour améliorer les performances en considérant plusieurs échelles. Le SSIM original est utilisé sur une seule échelle, mais le système visuel humain fonctionne sur plusieurs échelles.

La formule MS-SSIM est donnée par :

$$\text{MS-SSIM(x, y)} = l_M(x, y)^\alpha \prod_{i=1}^{M} c_i(x, y)^{\beta_i} s_i(x, y)^{\gamma_i} \tag{4.5}$$

Où :

— x et y sont les deux images à comparer.

— l_M, c_i, et s_i sont les fonctions de comparaison de luminance, de contraste et de structure, respectivement, chacune à l'échelle i. Ce sont les mêmes que dans la formule SSIM, mais sont calculées à différentes échelles.

— M est le nombre d'échelles utilisées.
— α, β_i, et γ_i sont des paramètres utilisés pour ajuster l'importance relative des différentes échelles. Généralement, ceux-ci sont définis de manière empirique.

Rapport signal sur bruit de crête (PSNR) : Le rapport signal sur bruit de crête entre deux images x et y peut être défini comme suit :

$$\text{PSNR(x, y)} = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{\text{MSE(x, y)}} \right) \tag{4.6}$$

où MAX_I est la valeur maximale possible du pixel de l'image.

Erreur moyenne absolue (MAE) : L'erreur moyenne absolue entre deux images x et y avec N pixels peut être définie comme suit :

$$\text{MAE(x, y)} = \frac{1}{N} \sum_{i=1}^{N} |x_i - y_i| \tag{4.7}$$

4.2.2.2 Exploration des fonctions d'activations

Pour le second composant de notre étude exploratoire, nous explorons l'influence des fonctions d'activation utilisées. Nous comparons 8 différentes fonctions d'activation : l'unité de rectification linéaire (relu) [238], l'unité linéaire exponentielle (elu) [239], l'unité linéaire exponentielle échelonnée (selu) [239], Mish [241], Swish [242], la tangente hyperbolique (tanh) [237], Mila [243] et Sharkfin [244].

Unité de Rectification Linéaire (ReLU) [238] :

$$\text{ReLU}(x) = \max(0, x) \tag{4.8}$$

Unité Linéaire Exponentielle (ELU) [239] :

$$\text{ELU}(x) = \begin{cases} x, & \text{si } x > 0 \\ \alpha(e^x - 1), & \text{autrement} \end{cases} \tag{4.9}$$

où α est un hyperparamètre à choisir.

Unité Linéaire Exponentielle Échelonnée (SELU) [240] :

$$\text{SELU}(x) = \lambda \begin{cases} x, & \text{si } x > 0 \\ \alpha(e^x - 1), & \text{autrement} \end{cases} \tag{4.10}$$

où α et λ sont des constantes prédéfinies. Dans le papier [240] $\alpha \approx 1.6733$ et $\lambda \approx 1.0507$.

Mish [241] :

$$\text{Mish}(x) = x \cdot \tanh(\ln(1 + e^x)) \tag{4.11}$$

Swish [242] :

$$\text{Swish}(x) = x \cdot \frac{1}{1 + e^{-\beta x}} \tag{4.12}$$

où β est un hyperparamètre à choisir.

Tangente Hyperbolique (tanh) [237] :

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{4.13}$$

Mila : Contrôler la concavité minimale dans la fonction d'activation [243] :

$$\text{Mila}(x) = x \cdot \tanh(\text{softplus}(1 + \beta)) = x \cdot \tanh(\ln(1 + \exp(x + \beta))) \tag{4.14}$$

Avec softplus(x) = $\log(1 + e^{x})$

Sharkfin [244] :

$$\text{Sharkfin(x)} = \tanh(e^{x}) \cdot \max(-1, x) \tag{4.15}$$

Ici, $\tanh(e^{x})$ représente la tangente hyperbolique de l'exponentielle de x, et $\max(-1, x)$ est la fonction ReLU avec un seuil de -1 ; elle renverra x si x est supérieur à -1, sinon elle renverra -1. Le produit de ces deux résultats donne le résultat final de la fonction.

Les fonctions d'activation influencent fortement les autoencodeurs. ReLU [238] est efficace et rapide, mais présente un risque d'unités inactives. ELU [239], permettant des valeurs négatives, évite ce problème mais est plus lent en raison de sa composante exponentielle. SELU [240] s'auto-normalise, mais ses paramètres prédéfinis peuvent ne pas convenir à tous les contextes. Mish [241] est reconnu pour ses bonnes performances, mais est computationnellement exigeant. Swish [242] offre une flexibilité via l'hyperparamètre β, qui demande cependant une calibration. Tanh [237], centré sur zéro, peut parfois souffrir de gradients qui disparaissent. Mila [243] propose une concavité ajustable, au prix d'une complexité computationnelle plus élevée. Enfin, Sharkfin [244], bien qu'innovant, est encore peu étudié et pourrait avoir des comportements inattendus.

Une représentation graphique des fonctions d'activations utilisées dans cette étude est visible au niveau de la figure 61

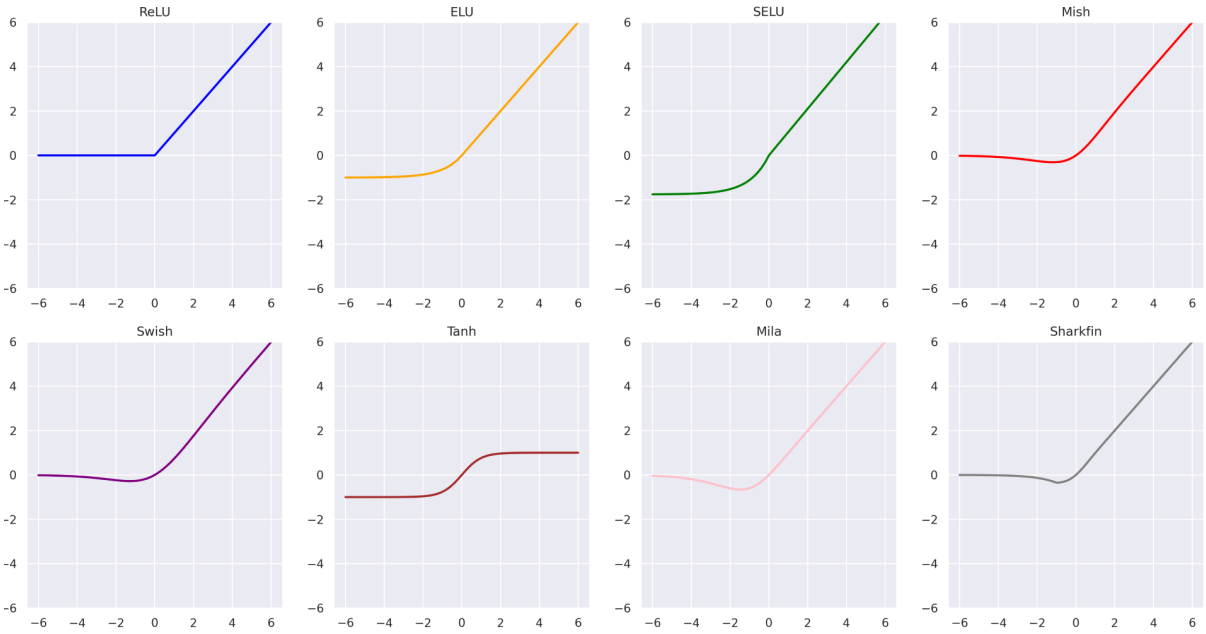

FIGURE 61: Figure montrant les différentes formes graphiques des fonctions d'activations : ReLU, ELU, SELU, Mish, Swish, Tanh, Mila et Sharkfin

4.2.2.3 Exploration des optimiseurs

Pour le dernier composant de notre étude exploratoire, nous étudions l'importance des optimiseurs utilisés et leur impact sur les résultats finaux du modèle. Dans cette étude, nous comparons trois optimiseurs différents :

— **Adam [234] :** L'optimiseur Adam est une méthode de descente de gradient stochastique basée sur l'adaptation des taux d'apprentissage pour chaque paramètre. Il est particulièrement efficace pour les problèmes d'apprentissage profond.
Adam est une combinaison de deux autres extensions de la descente de gradient stochastique [184] : RMSProp [185] (Root Mean Square Propagation) et Momentum [186]. RMSProp ajuste le taux d'apprentissage en divisant par une moyenne mobile (avec décroissance) du carré des gradients. Momentum accélère la convergence du gradient en prenant une moyenne mobile (avec décroissance) du gradient. L'algorithme Adam utilise à la fois les moyennes mobiles du gradient et du carré du gradient pour ajuster le taux d'apprentissage pour chaque poids dans le réseau de neurones.
Le pseudo-code de l'algorithme Adam est décrit ci-dessous dans la figure 62 :

Algorithm 1: Pseudo-code de l'algorithme Adam

Data: Entrée requise: Pas d'apprentissage (α); $\beta_1, \beta_2 \in [0, 1)$; ϵ; W_0: Poids initiaux pour l'entraînement

$m_0 \leftarrow 0$
$v_0 \leftarrow 0$
$t \leftarrow 0$
while W_t *n'a pas convergé* **do**
  $t \leftarrow t + 1$
  $g_t \leftarrow \nabla_W L(W_{t-1})_t$ `// Obtenir les gradients à l'étape t`
  $m_t \leftarrow \beta_1 \times m_{t-1} + (1 - \beta_1) \times g_t$
  $v_t \leftarrow \beta_2 \times v_{t-1} + (1 - \beta_2) \times g_t^2$
  $\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^t}$
  $\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^t}$
  $W_t \leftarrow W_{t-1} - \alpha \times \frac{\hat{m}_t}{\sqrt{\hat{v}_t}} + \epsilon$ `// Mettre à jour les poids`

Result: Retourner W_t (Poids apprenables résultants)

FIGURE 62: Description du fonctionnement de l'algorithme Adam à l'aide d'un pseudo-code. Les paramètres du pseudo-code sont : α est le pas utilisée pour ajuster la matrice de poids W, et ∇_W représente la dérivée directionnelle de W [234]

— **Rectified Adam [235] :**
L'optimiseur Rectified Adam (RAdam) [235] est une variante de l'Adam optimizer, introduit par Liu et al. en 2019 [234]. L'idée principale derrière RAdam est de rectifier la variance du taux d'apprentissage en utilisant une estimation de la borne inférieure de la variance non centrée du gradient. Contrairement à Adam, qui ajuste le taux d'apprentissage en fonction de la moyenne mobile du gradient et de son carré, RAdam utilise une longueur d'étape adaptative.
Le pseudo-code de l'algorithme RAdam est décrit ci-dessous dans la figure 63 :

Algorithm 2: Pseudo-code de l'algorithme Rectified Adam (RAdam)

Data: Entrée: $\{\alpha_t\}_{t=1}^{T}$: pas, $\{\beta_1, \beta_2\}$: taux de décroissance, θ_0: paramètre initial, $f_t(\theta)$: fonction objective stochastique.
Result: Sortie: θ_t: paramètres résultants

$m_0, v_0 \leftarrow 0, 0$
$\rho_\infty \leftarrow \frac{2}{(1-\beta_2)} - 1$
while t *dans* $\{1, \ldots, T\}$ **do**
  $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ `// Calculer les gradients par rapport à` $f_t(\theta)$ `à l'étape t`
  $v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2$
  $m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$
  $\hat{m}_t \leftarrow \frac{m_t}{(1-\beta_1^t)}$
  $\rho_t \leftarrow \rho_\infty - \frac{2t\beta_2^t}{(1-\beta_2^t)}$
  if *la variance est traçable, i.e.,* $\rho_t > 4$ **then**
    $l_t \leftarrow \sqrt{\frac{(1-\beta_2^t)}{v_t}}$ `// Taux d'apprentissage adaptatif`
    $r_t \leftarrow \sqrt{\frac{(\rho_t-4)(\rho_t-2)\rho_\infty}{(\rho_\infty-4)(\rho_\infty-2)\rho_t}}$ `// Terme de rectification de la variance`
    $\theta_t \leftarrow \theta_{t-1} - \alpha_t r_t \hat{m}_t l_t$ `// Mise à jour des paramètres`
  else
    $\theta_t \leftarrow \theta_{t-1} - \alpha_t \hat{m}_t$ `// Mise à jour des paramètres`
return θ_T

FIGURE 63: Description du fonctionnement de l'algorithme RAdam à l'aide d'un pseudo-code [235]

— **NovoGrad [236] :**
L'optimiseur NovoGrad est une variante de l'Adam optimizer qui combine des idées de descente de gradient stochastique avec normalisation des gradients. NovoGrad calcule d'abord la variance du gradient, puis normalise le gradient avant de mettre à jour les paramètres. Cette approche vise à accélérer la convergence tout en conservant une mémoire réduite. Le pseudo-code de novograd est décrit ci-dessous dans la figure 64 :

Algorithm 3: Pseudo-code de l'algorithme Novograd

Data: Entrée: Taux d'apprentissage initial λ_0, moments β_1, β_2, facteur de décroissance des poids d, nombre d'étapes T, fonctions g_l^t pour chaque couche l et gradient $\nabla_l L(w^t)$.

Result: Sortie: Paramètres de poids w_l^{T+1} pour chaque couche l

$t \leftarrow 0$ `// Initialisation`

$w_0 \leftarrow Init()$ `// Initialisation des poids`

$t \leftarrow 1$

`// Initialisation des moments pour chaque couche`

for *chaque couche* l **do**

  $v_l^1 \leftarrow \left\|g_l^1\right\|^2$

  $m_l^1 \leftarrow \frac{g_l^{p_1}}{v_l^1} + d \cdot w_l^1$

while $t \leq T$ **do**

  $\lambda_t \leftarrow LR(\lambda_0, t, T)$ `// Calcul du taux d'apprentissage global`

  for *chaque couche* l **do**

    $g_l^t \leftarrow \nabla_l L(w^t)$

    $v_l^t \leftarrow \beta_2 \cdot v_l^{t-1} + (1 - \beta_2) \cdot \|g_l^t\|^2$

    $m_l^t \leftarrow \beta_1 \cdot m_l^{t-1} + \left(\frac{g_l^{p_t}}{v_l^t} + d \cdot w_l^t\right)$

    $w_l^{t+1} \leftarrow w_l^t - \lambda_t \cdot m_l^t$ `// Mise à jour des paramètres`

  $t \leftarrow t + 1$

return w_l^{T+1} pour chaque couche l

FIGURE 64: Description du fonctionnement de l'algorithme NovoGrad à l'aide d'un pseudo-code. Les paramètres du pseudo-code sont : Le taux d'apprentissage λ_0, les moments β_1, β_2, le facteur de décroissance des poids d, et le nombre d'étapes T [236]

4.3 Dataset

Dans le domaine de l'intelligence artificielle appliquée à la reconnaissance d'images ferroviaires, le jeu de données railsem19 joue un rôle crucial. Il s'agit d'une collection d'images spécifiquement destinée à la segmentation sémantique des scènes de chemins de fer. La segmentation sémantique consiste à attribuer à chaque pixel d'une image une étiquette correspondant à la catégorie de l'objet auquel il appartient, ce qui est essentiel pour comprendre et analyser les scènes ferroviaires. Il est important de noter que la plupart des travaux dans ce domaine s'appuient sur des ensembles de données privés, ce qui limite l'accessibilité et la reproductibilité des recherches. C'est pourquoi, dans cette étude exploratoire, nous avons choisi de nous appuyer sur un ensemble de données public, à savoir railsem19. Pour adapter cet ensemble de données à nos besoins, nous l'avons préparé de manière à ce qu'il soit adapté à l'entraînement des autoencodeurs convolutionnels,

Le jeu de données railsem19 provient de diverses sources, comprenant des images capturées par des caméras embarquées sur des trains ainsi que des images aériennes prises par des drones. Ces images sont de haute qualité, avec des résolutions allant jusqu'à 1080p (1920 x 1080). Le dataset contient plusieurs milliers d'images, ce qui offre une diversité

suffisante pour entraîner des modèles de deep learning efficacement. En ce qui concerne les types de scènes ferroviaires, railsem19 est assez riche et varié, incluant des images de voies ferrées, de gares, de ponts ferroviaires, de matériel roulant, et de divers éléments d'infrastructure. De plus, pour faciliter la tâche de segmentation sémantique, les images sont accompagnées de masques d'annotation qui indiquent les classes d'objets présents. Ces masques sont extrêmement précis, ayant été annotés manuellement par des experts en reconnaissance d'images et en infrastructure ferroviaire. Cela fait de railsem19 un outil précieux pour les chercheurs et les ingénieurs travaillant sur des applications d'intelligence artificielle dans le domaine ferroviaire [218].

Ci-dessous un aperçu des images du dataset railsem19 au niveau de la figure 65 avec les résultats de ségmentation d'instances présentes dans les images du dataset au niveau des figures 66 67 ainsi qu'un tableau récapitulatif des caractéristiques du dataset 4.1.



FIGURE 65: Aperçu des images de scènes ferroviaires du dataset railsem19 [218]

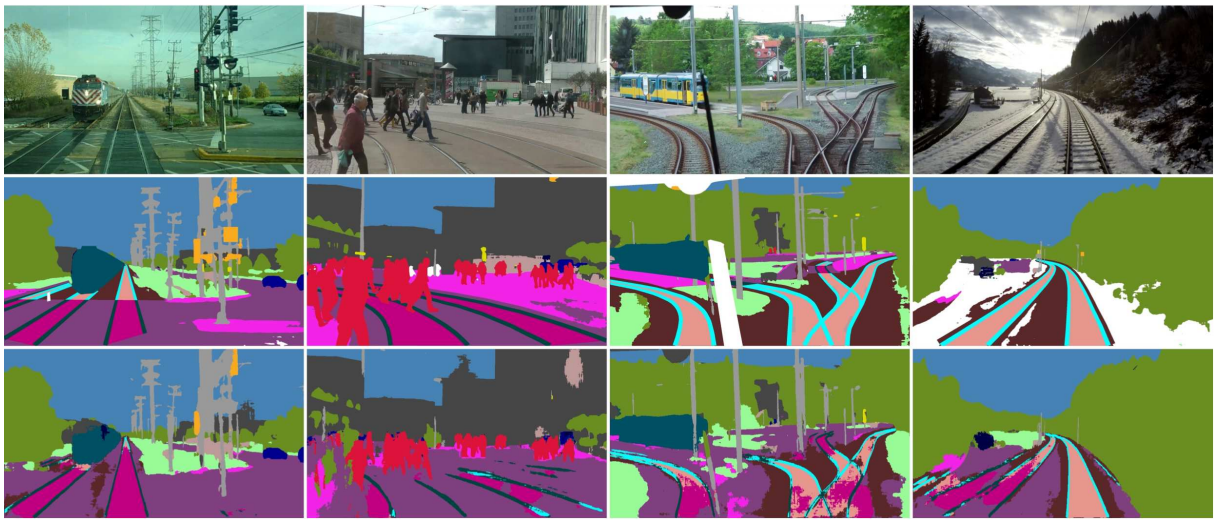

FIGURE 66: Description des différentes classes (instances) présentes dans le dataset railsem19 [218]

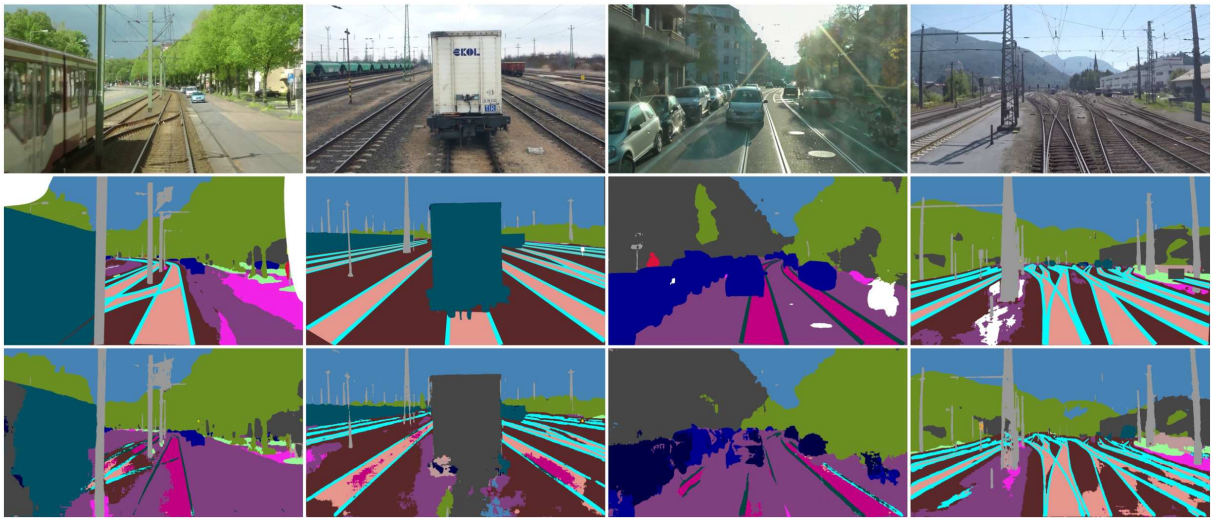

FIGURE 67: Description des différentes classes (instances) présentes dans le dataset railsem19 [218]

TABLE 4.1: Caractéristiques du jeu de données railsem19

Caractéristique	Valeur
Nom du jeu de données	railsem19
Nombre d'images	8500
Résolution	1920 x 1080 pixels
Scènes de tramway	Oui (≈ 1200 images)
Classes	19 (Entraîné sur modèle)
Format d'annotation	Segmentation au niveau des pixels

4.3.1 Manipulation du dataset

Dans le cadre de la mise au point de modèles de détection d'obstacles ferroviaires, la qualité et la pertinence du jeu de données utilisé sont fondamentales. railsem19, à notre connaissance, représente le seul dataset ferroviaire dédié à la segmentation de la scène ferroviaire, rendant son choix judicieux pour cette tâche. Toutefois, l'adaptation de ce dataset aux besoins spécifiques de la détection d'anomalies nécessite une phase préalable de prétraitement rigoureux. La phase de prétraitement du dataset railsem19 se décompose en plusieurs étapes méthodiques pour assurer la pertinence et la fiabilité de l'ensemble des données retenues. Tout d'abord, un filtrage manuel a été effectué pour conserver uniquement les images pertinentes à notre contexte. Parmi les images écartées, on note la présence d'images inutilisables, telles que celles entièrement noires typiques de séquences en tunnel, les images pixelisées ou celles issues de contextes de tramway. Ont également été éliminées les images présentant de potentiels obstacles sur les voies, ainsi que celles comportant des artefacts ou susceptibles d'introduire un biais, comme les images sur lesquelles du texte a été superposé ou celles prises depuis la cabine de la locomotive plutôt qu'à partir de capteurs fixés sur cette dernière (voir la section 4.3.2).

Suite à ce filtrage, une étape de normalisation a été mise en œuvre afin d'harmoniser l'ensemble de données. Cela a impliqué l'ajustement de la taille des images pour qu'elles soient uniformes et la conversion de leur format, par exemple du BGR au RGB. Enfin, pour cibler les zones d'intérêt au sein de chaque image, nous avons employé un modèle de détection d'objets basé sur YOLO-V3 [124]. Ce modèle, de par sa légèreté et son efficacité, nous a permis d'extraire en temps réel les Régions d'Intérêt (RoIs) représentées par des "bounding boxes". Ces boîtes englobantes ont été spécifiquement utilisées pour délimiter la RoI afin d'obtenir en sortie un masque précisément représentatif de la voie de circulation principale. Ce masque est essentiel pour concentrer l'attention des futurs modèles sur la portion centrale des voies, facilitant ainsi la distinction entre des situations normales et anormales. Cette méthodologie, dont les étapes principales sont illustrées à la figure 68, établit un cadre solide pour l'entraînement et l'évaluation des modèles basés sur le dataset railsem19.

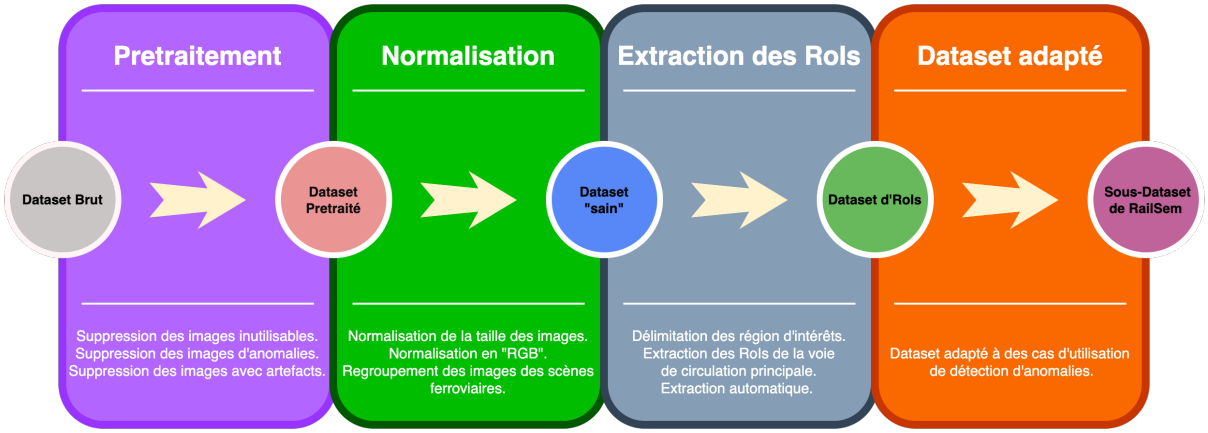

FIGURE 68: Résumé des étapes suivies pour élaborer le pipeline de prétraitement du dataset railsem19

4.3.2 Pretraitement du dataset

Le prétraitement du dataset railsem19 vise à élaborer un ensemble de données "sain" [250], spécialisé pour la détection d'anomalies, telles que les obstacles sur les voies ferrées. Cette étape initiale implique l'exclusion des images inutilisables ou celles présentant des anomalies et des artefacts, comme les textes superposés à la scène ferroviaire. En se concentrant uniquement sur les images représentant un cas "normal", sans obstacles ni perturbations, nous renforçons la capacité du modèle à identifier les caractéristiques essentielles pour la détection d'obstacles. L'étape suivante du processus est la normalisation des images sélectionnées, celle-ci permet d'homogénéiser l'ensemble au niveau de la résolution et du format, en convertissant par exemple de BGR à RGB et en rendant uniforme la taille de la résolution des d'images. Cette standardisation facilite les manipulations ultérieures des données.

La figure 69 représente des exemples d'images qui ont été supprimées lors de la phase de prétraitement de l'ensemble de données railsem19. La figure 70 montre les exemples d'images retenues, noramlisées, qui représentent le cas dit "normal".

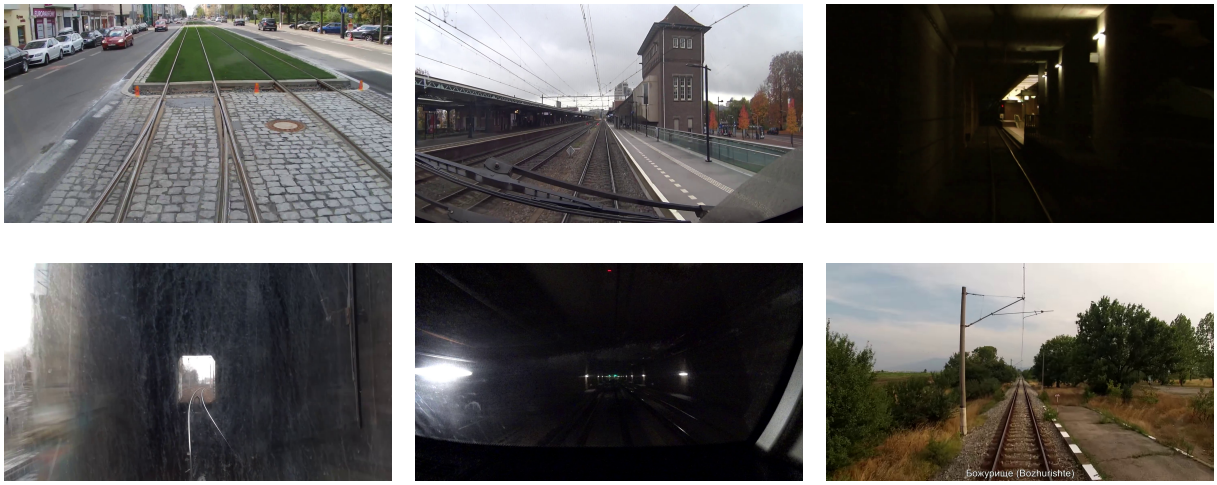

FIGURE 69: Exemples d'images supprimées lors de la creation du sous-dataset de railsem19

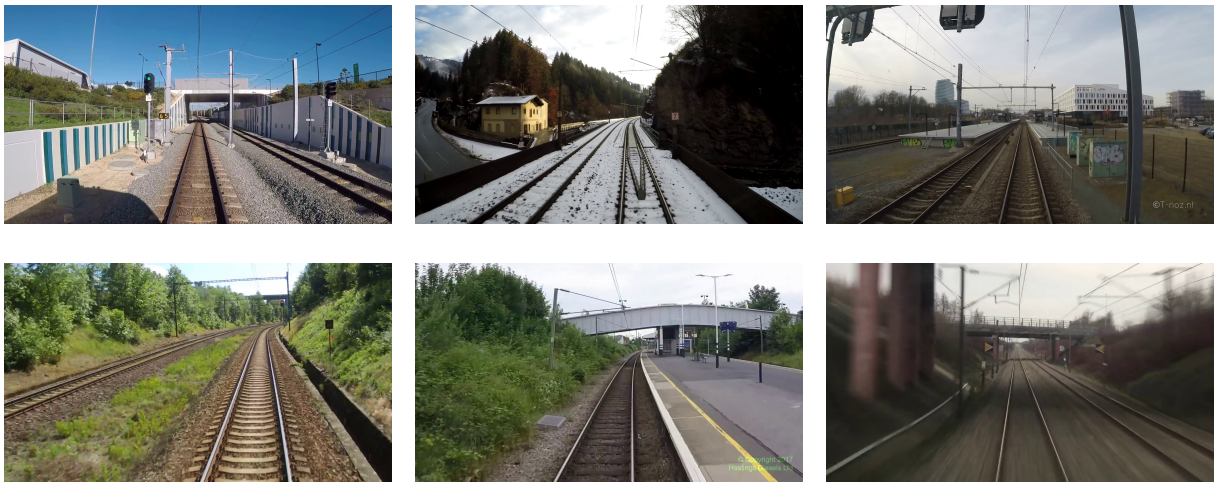

FIGURE 70: Exemples d'images normalisées illustrant le cas "normal"

4.3.3 Extraction des Régions d'intérêts

Suite au prétraitement et à la normalisation du dataset railsem19, une identification des régions d'intérêt (RoI) dans les images a été effectuée. Cette étape permet d'isoler les pixels représentatifs du "cas normal", optimisant ainsi les données pour l'entraînement des modèles. L'accent est mis sur la détection d'obstacles sur la voie principale, rendant superflu le traitement global de l'image et assurant une meilleure allocation des ressources de calcul.

Le choix s'est porté sur le modèle YOLOv3-tiny, apprécié pour son efficacité et sa compacité, tout en démontrant des performances compétitives, comme référencé dans

le tableau (4.3). Les boîtes englobantes ont été générées en se basant sur les labels de segmentation du railsem19.

L'algorithme de clustering hiérarchique, HCA, a été utilisé pour filtrer les boîtes englobantes correspondant à la voie principale. Une fois ces boîtes identifiées, les sommets distinctifs ont été utilisés pour délimiter avec précision la RoI, aboutissant à un masque final adapté à la détection d'obstacles.

4.3.3.1 Génération des boîtes englobantes

La génération des boîtes englobantes au niveau des voies de circulations porte sur l'utilisation de YOLOV3-Tiny, un modèle dédié pour la détection d'objets faisant partie de la famille des modèles YOLO. Les modèles YOLO, qui signifient "You Only Look Once", sont une famille de modèles de détection d'objets en temps réel. Le principal avantage de YOLO réside dans sa capacité à prédire les classes et les emplacements d'objets en une seule passe, ce qui le rend extrêmement rapide par rapport à d'autres méthodes de détection qui effectuent ces tâches en deux étapes distinctes. Il existe plusieurs variantes du modèle YOLO, notamment YOLOv1 [58], YOLOv2 (également connu sous le nom de YOLO 9000) [123], YOLOv3 [124], YOLOv4 [126] et YOLOv5 [127]. Chaque version successive de YOLO a introduit diverses améliorations en termes de précision de détection et de vitesse. Par exemple, YOLOv3 [124] a ajouté des boîtes englobantes de différentes tailles et une meilleure gestion des objets de petite taille. YOLOv4, quant à lui, a introduit plusieurs optimisations pour améliorer encore les performances et la précision.

YOLOv3-tiny est une variante de YOLOv3 [124] optimisée pour la vitesse d'exécution, au détriment de la précision. Il est particulièrement utile pour les applications embarquées où les ressources de calcul sont limitées. YOLOv3-tiny utilise une architecture de réseau de neurones convolutifs simplifiée, avec moins de couches et de paramètres que la version complète de YOLOv3.

Concernant la notion de "backbone" dans les modèles de détection d'objets, il s'agit de la partie du réseau de neurones qui est responsable de l'extraction des caractéristiques d'une image. Cette partie est généralement composée de couches convolutives. Dans l'état de l'art, plusieurs backbones sont couramment utilisés. Par exemple, VGG16 [190] et VGG19 [190], qui sont des architectures de réseau de neurones profondes, sont connus pour leur capacité à extraire efficacement des caractéristiques de bas niveau. ResNet [114], avec ses différentes variantes (ResNet-50 [187], ResNet-101 [188], etc.), est réputé pour sa capacité à entraîner des réseaux très profonds grâce à l'utilisation de connexions résiduelles.

YOLOv3-tiny utilise Darknet19 [123] comme backbone. Darknet est une architecture de réseau de neurones conçue pour être légère et rapide, et est souvent utilisée dans les variantes YOLO pour une détection d'objets efficace et en temps réel. Darknet19 [123] est une version spécifique de Darknet [116] avec 19 couches, optimisée pour un équilibre

entre performance et précision.

La figure 71 représente l'architecture du modèle YOLOV3-Tiny. La table 4.2 décrit les paramètres du modèle YOLOV3-Tiny utilisant le backbone Darknet19. La table 4.3 montre une comparaison entre les différents modèles YOLO de détection d'objets ainsi que quelques modèles de ségmentation d'instances.

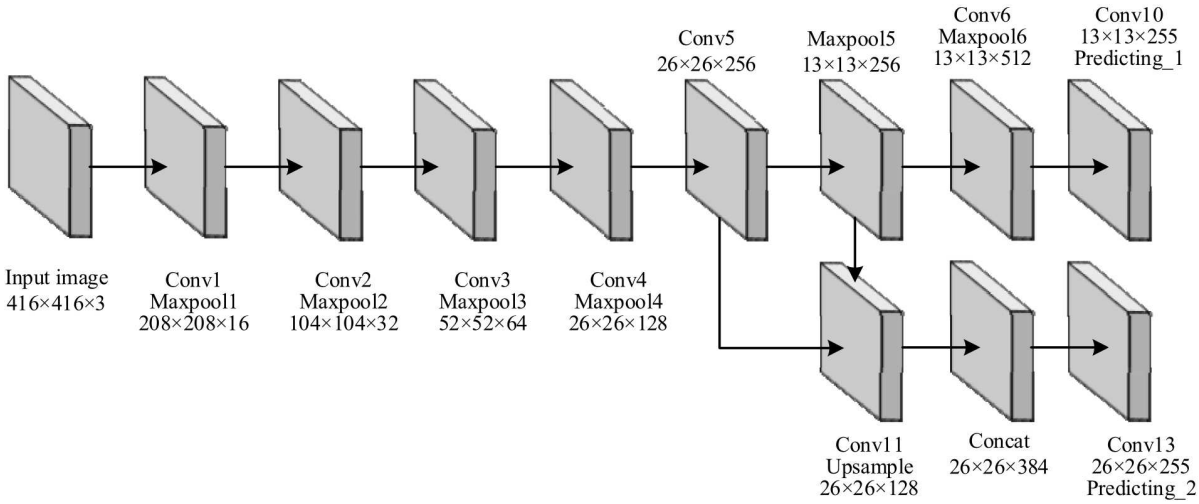

FIGURE 71: Description graphique de l'architecture du modèle YOLOV3-tiny [125]

Couche	Type de couche	Filtres	Taille/Pas	Entrée	Sortie
0	Conv	16	$3 \times 3/1$	$416 \times 416 \times 3$	$416 \times 416 \times 16$
1	Maxpool		$2 \times 2/2$	$416 \times 416 \times 16$	$208 \times 208 \times 16$
2	Conv	32	$3 \times 3/1$	$208 \times 208 \times 16$	$208 \times 208 \times 32$
3	Maxpool		$2 \times 2/2$	$208 \times 208 \times 32$	$104 \times 104 \times 32$
4	Conv	64	$3 \times 3/1$	$104 \times 104 \times 32$	$104 \times 104 \times 64$
5	Maxpool		$2 \times 2/2$	$104 \times 104 \times 64$	$52 \times 52 \times 64$
6	Conv	128	$3 \times 3/1$	$52 \times 52 \times 64$	$52 \times 52 \times 128$
7	Maxpool		$2 \times 2/2$	$52 \times 52 \times 128$	$26 \times 26 \times 128$
8	Conv	256	$3 \times 3/1$	$26 \times 26 \times 128$	$26 \times 26 \times 256$
9	Maxpool		$2 \times 2/2$	$26 \times 26 \times 256$	$13 \times 13 \times 256$
10	Conv	512	$3 \times 3/1$	$13 \times 13 \times 256$	$13 \times 13 \times 512$
11	Maxpool		$2 \times 2/1$	$13 \times 13 \times 512$	$13 \times 13 \times 512$
12	Conv	1024	$3 \times 3/1$	$13 \times 13 \times 512$	$13 \times 13 \times 1024$
13	Conv	256	$1 \times 1/1$	$13 \times 13 \times 1024$	$13 \times 13 \times 256$
14	Conv	512	$3 \times 3/1$	$13 \times 13 \times 256$	$13 \times 13 \times 512$
15	Conv	255	$1 \times 1/1$	$13 \times 13 \times 512$	$13 \times 13 \times 255$
16	YOLO				
17	Route 13				
18	Conv	128	$1 \times 1/1$	$13 \times 13 \times 256$	$13 \times 13 \times 128$
19	Upsampling		$2 \times 2/1$	$13 \times 13 \times 128$	$26 \times 26 \times 128$
20	Route 19,8				
21	Conv	256	$3 \times 3/1$	$13 \times 13 \times 384$	$13 \times 13 \times 256$
22	Conv	255	$1 \times 1/1$	$13 \times 13 \times 256$	$13 \times 13 \times 256$
23	YOLO				

TABLE 4.2: Structure et paramètres du réseau YOLOv3-tiny [125]

Modèle	Set d'entraînement	Set de test	mAP (%) ↑	FLOPS ↓	Vitesse (FPS) ↑
Tiny YOLO	MS-COCO	MS-COCO	23.7	5.41	244
YOLOv3-tiny	MS-COCO	MS-COCO	33.1	5.56	220
YOLOv3-416	MS-COCO	MS-COCO	55.3	65.86	35
DB-Swin-L*	MS-COCO	MS-COCO	51.0	35	1.9
SOTR*	MS-COCO	MS-COCO	41.5	35	8.1
Mask R-CNN*	MS-COCO	MS-COCO	38.6	36	23.2

TABLE 4.3: Comparaison des performances de différentes variantes de YOLO (YOLO V1, YOLOV3-Tiny, YOLOV3), ainsi que les modèles de ségmentation d'instances (DB-Swin-L, SoTR, Mask R-CNN). (↑) : Plus élevé est meilleur, (↓) : Plus bas est meilleur. Les modèles de détection d'objets (YOLO) sont entraînés et utilisé par le GPU Pascal Titan X, tandis que les modèles de ségmentation d'instances (∗) utilisent le GPU RTX 3090 [191, 192]

Au sein du processus d'exploitation de l'ensemble de données railsem19 en vue d'une analyse approfondie des attributs relatifs aux voies ferrées, l'intégration des masques mis

à disposition par railsem19 est une étape fondamentale. Cette intégration sert de première étape pour générer automatiquement des annotations de boîtes englobantes. Cela est impératif pour la préparation et l'entraînement du modèle YOLOv3-Tiny, en vue de détecter de manière efficace les boîtes englobantes entourant l'ensemble des voies ferrées. Les masques, en tant qu'éléments d'ancrage, constituent des repères inestimables qui permettent d'identifier avec précision les zones contenant des rails. À cet effet, une méthode algorithmique élaborée est mise en œuvre pour extraire les contours basés sur les masques, et une technique de détection de rectangle englobant minimal est ensuite appliquée, de manière à déterminer les boîtes englobantes les plus adaptées. Compte tenu de l'objectif spécifique de prédire une unique classe, celle des voies ferrées, une attention particulière est accordée à la génération d'un large éventail de boîtes englobantes en vue de délimiter efficacement les régions d'intérêt (RoI). Cette approche vise à maximiser la détection des RoI, même en présence de détections erronées (faux positifs), garantissant ainsi une probabilité accrue de capturer de manière fiable les voies ferrées. Cette phase, en tant que prélude essentiel, est capitale pour affiner et optimiser le modèle YOLOv3-Tiny pour détecter, par des boîtes englobantes, l'enssemble des voies de circulations présentes dans la scène ferroviaire.

La figure 72 montre les différentes étapes suivies lors de la génération des boîtes englobantes à partir des masques railsem19.

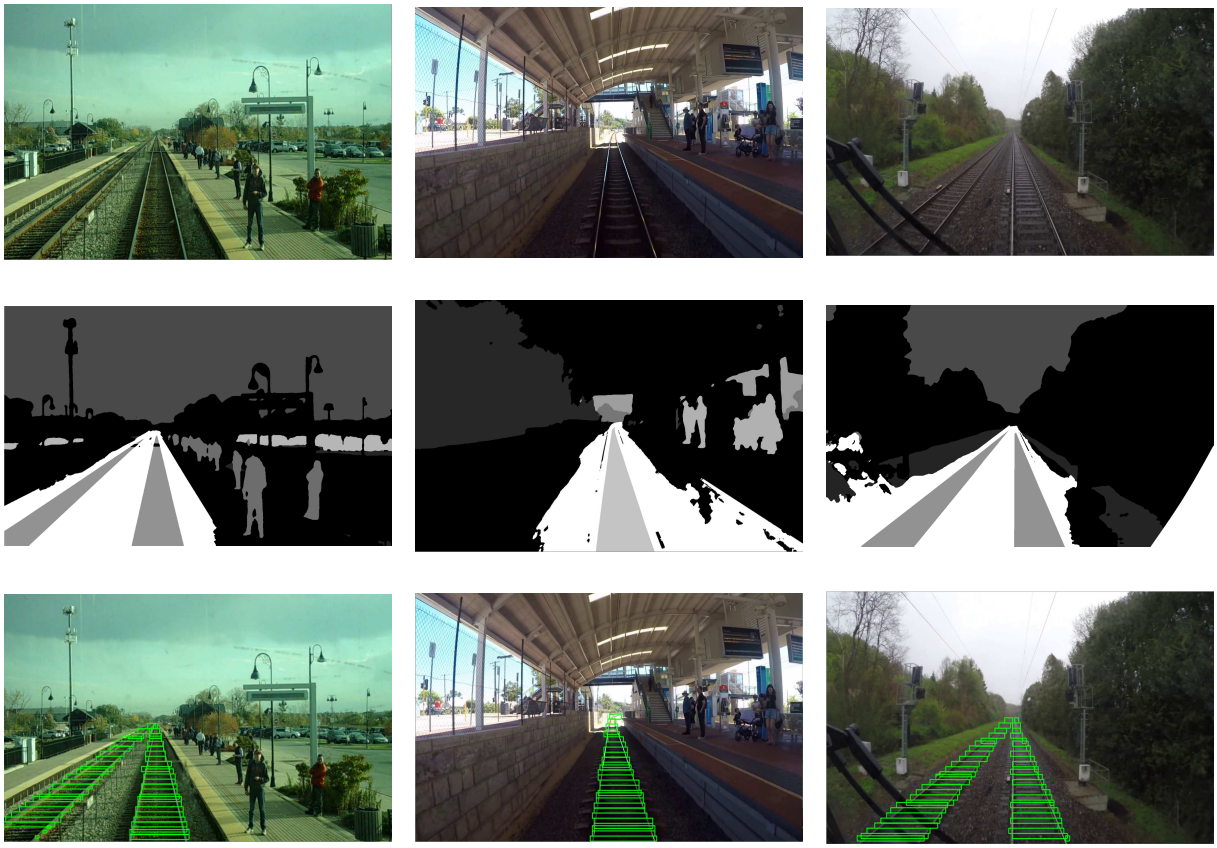

FIGURE 72: Etapes suivies lors de la création des boîtes englobantes à partir des masques railsem19, la première ligne montre des images de scènes ferroviaires railsem19, la deuxième ligne montre des images des masques correspondants, la troisième ligne montre la génération d'annotation de boîtes englobantes à partir des masques des voies de circulation.

Par la suite, l'étape suivante consistera à entraîner le modèle YOLOv3-Tiny sur les images annotées avec des boîtes englobantes. Il est important de noter que la version standard de YOLOv3-Tiny a été utilisée sans apporter de modifications aux hyperparamètres. De plus, les classes ont été adaptées pour que le modèle se concentre uniquement sur la détection d'une seule classe, celle des voies ferrées. La figure 73 montre la détection de YOLOV3-Tiny sur un ségment d'une vidéo de 4h tout le long du trajet Toulouse Matabiau - Paris Montparnasse.



FIGURE 73: Résultats de détection du modèle YOLOV3-Tiny sur une vidéo tout le long du trajet Toulouse Matabiau - Paris Montparnasse.

4.3.3.2 Extraction des clusters de la voie de circulation principale

Afin de pouvoir filtrer les boîtes englobantes qui représentent la voie de circulation principale, un algorithme de clustering est utilisé notamment le HCA [183] (*Hierarchical Clustering Analysis*). L'algorithme HCA est une méthode utilisée pour la classification hiérarchique de données. Il est couramment utilisé en analyse de données, bio-informatique et autres domaines scientifiques. L'algorithme regroupe progressivement les données en les fusionnant en fonction de la similarité, souvent mesurée en utilisant des distances, comme la distance euclidienne ou la distance de Manhattan [189]. L'algorithme construit une hiérarchie de clusters en commençant par considérer chaque point de données comme un cluster, puis en fusionnant les clusters les plus proches à chaque étape, jusqu'à ce que tous les points de données soient regroupés en un seul cluster.

Considérons un ensemble de données $X = \{x_1, x_2, \ldots, x_n\}$ où chaque x_i est un point dans un espace euclidien.

Soit X_i et X_j deux clusters. Les notations suivantes sont utilisées :

— n_i : le nombre de points dans le cluster X_i.

— n_j : le nombre de points dans le cluster X_j.

— $c(X)$: le centroïde d'un ensemble X, défini comme la moyenne de tous les points dans X.

— $d(X_i, X_j)$: une mesure de distance entre deux clusters X_i et X_j.

— $\Delta(X_i, X_j)$: la distance de Ward entre deux clusters X_i et X_j.

Avec ces notations, nous avons :

— Lien unique :

$$d(X_i, X_j) = \min_{x \in X_i, y \in X_j} \|x - y\|$$

— Lien complet :

$$d(X_i, X_j) = \max_{x \in X_i, y \in X_j} \|x - y\|$$

— Lien moyen :

$$d(X_i, X_j) = \frac{1}{|X_i||X_j|} \sum_{x \in X_i, y \in X_j} \|x - y\|$$

— Lien de Ward :

$$\Delta(X_i, X_j) = \frac{n_i n_j}{n_i + n_j} \|c(X_i) - c(X_j)\|^2$$

Où :

— $\|c(X_i) - c(X_j)\|$ est la distance euclidienne entre les centroïdes des deux clusters.

— $c(X) = \frac{1}{|X|} \sum_{x \in X} x$ est le centroïde de l'ensemble X, qui est la moyenne des points dans X.

Afin d'effectuer le filtrage des boîtes englobantes, HCA [183] est appliqué sur les boîtes englobantes résultantes de la détection du modèle YOLOV3-tiny. Plus précisément, HCA est utilisé sur les centroïdes de ces boîtes englobantes pour générer des clusters. Chaque centroïde est caractérisé par deux coordonnées, x_{center} et y_{center}, représentant les positions centrales de la boîte englobante sur les axes horizontal et vertical, respectivement. En appliquant l'algorithme HCA sur ces centroïdes, il est possible de générer, dans un premier temps, des clusters de boîtes englobantes qui sont probablement associées à la voie ferrée principale, et de distinguer celles qui pourraient représenter d'autres voies dans la scène. De plus, afin de filtrer parfaitement les clusters représentant la voie ferrée principale, deux contraintes doivent être ajoutées lors de la sélection des clusters détectés par l'algorithme HCA.

— **Premièrement** : Un cluster doit être représenté par au moins trois individus (boîtes englobantes)

— **Deuxièment** : Nous sélectionnons uniquement les clusters pour lesquels les centres le long de la coordonnée x varient le moins. Mathématiquement, cela peut être réalisé en minimisant la variance le long de l'axe x. Soit X l'ensemble des coordonnées x_{center} pour un cluster donné, la variance V le long de l'axe x est donnée par :

$$V = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2 \tag{4.16}$$

où n est le nombre d'individus dans le cluster, x_i est la coordonnée x_center de l'individu i, et μ est la moyenne des coordonnées x_center. Le but est de sélectionner le cluster qui minimise cette variance. Ces deux étapes sont nécessaires afin de bien définir que les boîtes englobantes représentatives des RoI à extraire.

La figure 74 montre le résultat de détection de YOLOV3-Tiny suivi par le filtrage de l'algorithme HCA [183] avec les contraintes de sélection imposées.

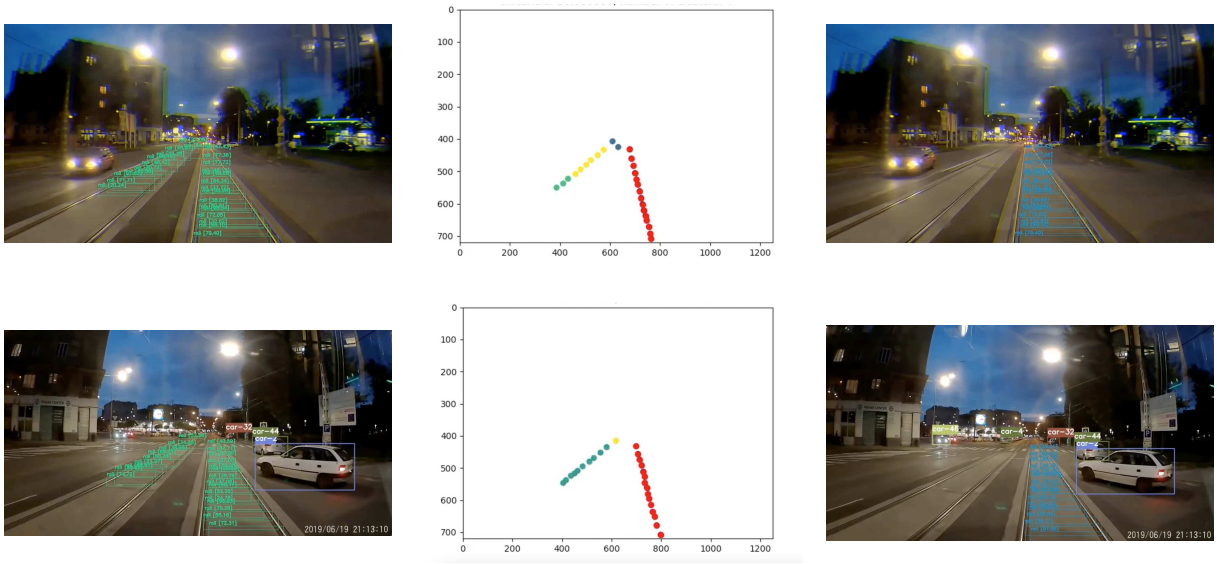

FIGURE 74: Figure montrant les étapes de filtrage des boîtes englobantes. Les figures de la première colonne représentent les images résultantes de la détection avec YOLOV3-Tiny. La deuxième colonne montre les différents clusters générés avec l'algorithme HCA [183], avec imposition des contraintes pour la séléction du cluster représentatif de la voie de circulation principale (en rouge). La troisième colonne montre la même scène ferroviaire initiale avec suppression des boîtes englobantes des autres voies de circulation.

4.3.3.3 Délimitation de la RoI

La troisième étape du processus consiste à créer une forme pour délimiter la Région d'Intérêt (RoI) et à l'extraire en utilisant les clusters filtrés. Cette étape repose sur le raccordement des centres de chaque bord gauche et droit d'une boîte englobante représentative de la voie ferrée principale avec d'autres boîtes englobantes pour créer une forme délimitant la RoI.

Soit $C = \{c_1, c_2, ..., c_n\}$ l'ensemble des centroïdes des boîtes englobantes filtrées, où chaque centroïde c_i est représenté par ses coordonnées (x_{center}, y_{center}). Pour chaque boîte englobante b_i de la boîte englobante principale, nous connectons les centres de son bord gauche $(x_{center} - w/2, y_{center})$ et de son bord droit $(x_{center} + w/2, y_{center})$ avec les centres correspondants des bords gauche et droit de la boîte englobante b_{i+1} suivante. Cela peut être formulé comme suit :

$$(x^i_{center} - w_i/2, y^i_{center}), (x^i_{center} + w_i/2, y^i_{center}), \tag{4.17}$$

$$(x^{i+1}_{center} - w_{i+1}/2, y^{i+1}_{center}), (x^{i+1}_{center} + w_{i+1}/2, y^{i+1}_{center}) \tag{4.18}$$

où w_i et w_{i+1} sont les largeurs des boîtes englobantes b_i et b_{i+1} respectivement. Cette

opération est répétée pour chaque paire consécutive de boîtes englobantes dans le cluster, formant une forme polygonale qui délimite la RoI.

En définitive, et sur la base de la délimitation de la RoI, un masque est créé pour représenter la RoI extraite. Ce masque est une image binaire de la même taille que l'image d'origine, où les pixels à l'intérieur de la RoI sont définis à 1 (ou blancs), et tous les autres pixels sont définis à 0 (ou noirs). Ce masque peut être utilisé pour filtrer l'image d'origine et ne garder que la partie de l'image qui correspond à la RoI.

La figure 75 montre les résultats de la délimitation des RoIs sur un cluster filtré des boîtes englobantes qui représentent la voie de circulation principale ainsi que la génération du masque de la RoI.



FIGURE 75: Génération d'une forme pour délimiter les RoIs à partir des clusters de la voie de circulation principale et extraction d'un masque

4.3.4 Dataset d'entraînement

Après avoir fait passer railsem19 par le pipeline pour extraire un sous-ensemble dédié à la détection d'anomalies (détection d'obstacles), le jeu de données résultant a été généré et se compose de 1353 images de RoI représentant des voies ferrées. Ce jeu de données est réparti en 80% pour l'entraînement et 20% pour la validation. De plus, les images ont été redimensionnées à une résolution de 256x256 pixels. Cette démarche constitue une étape essentielle pour préparer les données pour le processus d'apprentissage des autoencodeurs conventionnels, permettant à l'algorithme de se concentrer uniquement sur la RoI, qui est dans notre cas, les voies ferrées.

La figure 76 donne un aperçu du jeu de données résultant.

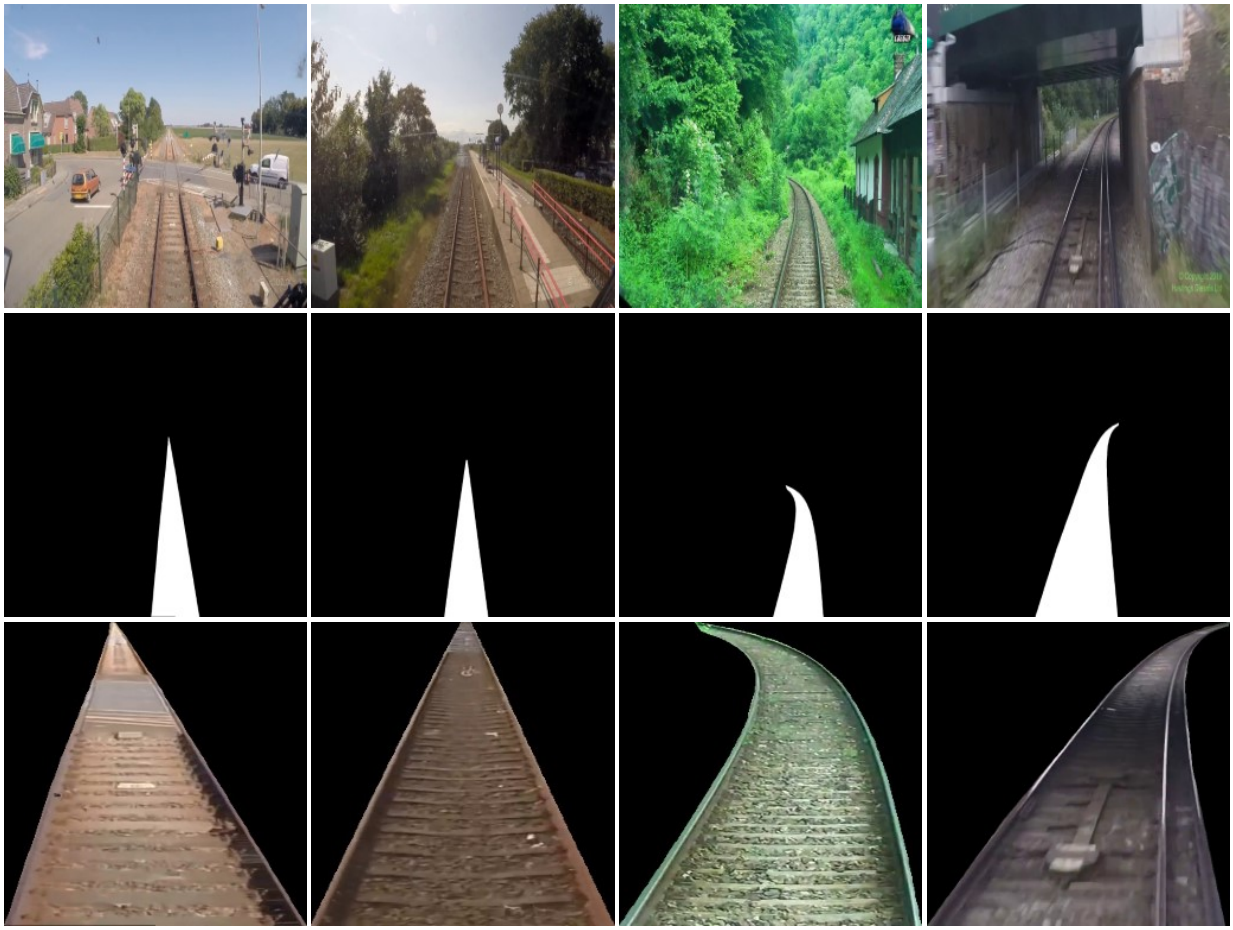

FIGURE 76: Aperçu du dataset résultant. La première ligne montre les images originales railsem19. La deuxième ligne montre les masques extraits à partir du pipeline de filtrage pour extraire les RoIs. La troisème ligne montre les images des RoIs.

4.3.5 Dataset de validation

La création d'un ensemble de validation basé sur le jeu de données filtré railsem19 présente certains défis. L'un des principaux défis pour valider les autoencodeurs convolutionnels est la nécessité d'avoir des données sur les obstacles, ceci est une tâche difficile. De plus, en ce qui concerne la littérature scientifique, aucun ensemble de données décrivant les obstacles ferroviaires n'a été rendu public jusqu'à présent. Nous proposons donc l'utilisation du réseau antagoniste génératif Gaussian-Poisson (GP-GAN) [149] pour incruster des obstacles sur les arrière-plans. Notre ensemble de validation propose une combinaison de :

— **Jeux de données d'arrière-plan (backgrounds)** : ce sont des images normales de RoI de voies ferrées ne contenant pas d'obstacles.

— **Images incrustées (incrustations)** : ce sont des images du même arrière-plan où nous incrustons des obstacles à l'aide du modèle GP-GAN [149].

L'utilisation du GP-GAN [149] est efficace car il donne du réalisme à la scène en fondant l'image de destination avec celle de la source et évite d'incruster manuellement

des images.

Les réseaux antagonistes génératifs (GANs) [221] sont constitués de deux parties, le générateur G et le discriminateur D. Le générateur tente de produire des données qui ressemblent aux vraies données, tandis que le discriminateur tente de distinguer les données générées des vraies données. Cette confrontation se fait selon la formule suivante :

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{4.19}$$

où x est une vraie donnée, z est un échantillon aléatoire et $G(z)$ est la donnée générée.

Pour le cas du modèle GP-GAN [149], le modèle cherche à créer une image correctement mélangée (blended) et de haute résolution en optimisant une fonction de perte qui se compose d'une contrainte de couleur et d'une contrainte de gradient. Cette fonction vise à rendre l'image générée plus réaliste et naturelle. Le framework GP-GAN [149] fonctionne en cherchant à optimiser une fonction de perte qui consiste en une contrainte de couleur et une contrainte de gradient. La contrainte de couleur vise à rendre l'image générée plus réaliste et naturelle, tandis que la contrainte de gradient capture les détails de haute résolution, tels que les textures et les bords. La contrainte de couleur est construite à partir d'une image réaliste de basse résolution, générée par un GAN spécifique appelé Blending GAN. Ce dernier apprend à mélanger une image copiée-collée et à générer une nouvelle image réaliste qui est sémantiquement similaire à l'entrée. La contrainte de gradient, quant à elle, vise à générer les détails de haute résolution de l'image composite. Pour cela, l'équation Gaussian-Poisson est utilisée pour contraindre l'image haute résolution à avoir un gradient similaire à l'image composite, tout en approximant la couleur de l'image de basse résolution.

En pratique, le GP-GAN [149] peut générer des images réalistes à n'importe quelle résolution. À partir d'une image composite donnée, une image de basse résolution est d'abord obtenue. Ensuite, l'équation Gaussian-Poisson est utilisée pour mettre à jour cette image à une échelle plus fine. Ce processus est répété jusqu'à obtenir une image réaliste finale de même résolution que l'image composite initiale.

La fonction de perte combinée est définie comme suit :

$$L(x, x_g) = \lambda L_{l2}(x, x_g) + (1\text{-}\lambda) L_{adv}(x, x_g) \tag{4.20}$$

où $\lambda = 0.999$. L_{l2} est défini comme suit :

$$L_{l2}(x, x_g) = ||G(x)\text{-}x_g||_2^2 \tag{4.21}$$

et L_{adv} est défini comme suit :

$$L_{adv}(x, x_g) = \max_{D} E_{x \in X}[D(x_g)\text{-}D(G(x))]. \tag{4.22}$$

Dans ce contexte, G représente le générateur d'un GAN, ayant pour objectif de produire des données qui ressemblent aux vraies données. D est le discriminateur du GAN, dont le rôle est de distinguer entre les données produites par G et les vraies données. La variable x correspond à une vraie donnée, tandis que z est un échantillon aléatoire utilisé comme entrée pour le générateur. La donnée générée par le générateur à partir de cet échantillon est notée $G(z)$. La fonction d'objectif du GAN, qui évalue la performance combinée du générateur G et du discriminateur D, est représentée par $V(D, G)$. Quant à x_g, elle fait référence à l'image générée par le GP-GAN [149]. Le coefficient λ détermine la contribution relative des deux termes dans la fonction de perte combinée. La perte $L2$ entre la donnée générée $G(x)$ et l'image générée x_g est notée $L_{l2}(x, x_g)$, tandis que la perte adversariale qui mesure la capacité du générateur à tromper le discriminateur est définie par $L_{adv}(x, x_g)$.

Ainsi, GP-GAN [149] parvient à réaliser un mélange réaliste d'images, en particulier pour des tâches comme l'incrustation d'obstacles sur des scènes de voies ferrées, en optimisant une combinaison de contraintes de couleur et de gradient.

La figure 77 montre le fonctionnement du framework GP-GAN [149]. La figure 78 montre le résultat d'inscrustation des obstacles sur les scènes ferroviaires du dataset de validation.

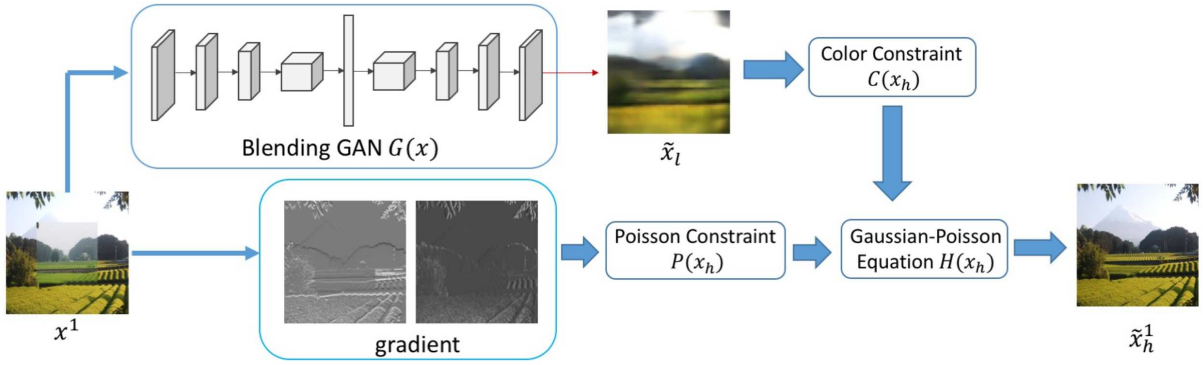

FIGURE 77: Fonctionnement du framework GP-GAN [149] : Pour une image composite x, on génère d'abord une version simplifiée, l'image de basse résolution $\tilde{x}_l$, grâce au Blending GAN $G(x)$. Cette version utilise x_1 comme entrée, qui représente le niveau le moins détaillé de l'image à partir d'une structure appelée pyramide laplacienne. L'ajustement de cette image utilise un modèle mathématique, l'équation de Gaussian-Poisson $H(x_h)$, qui est régulée par les contraintes $C(x_h)$ et $P(x_h)$ pour préserver les éléments essentiels tout en modifiant les détails. À partir de cette opération, une image plus détaillée, $\tilde{x}_{1h}$, est obtenue.

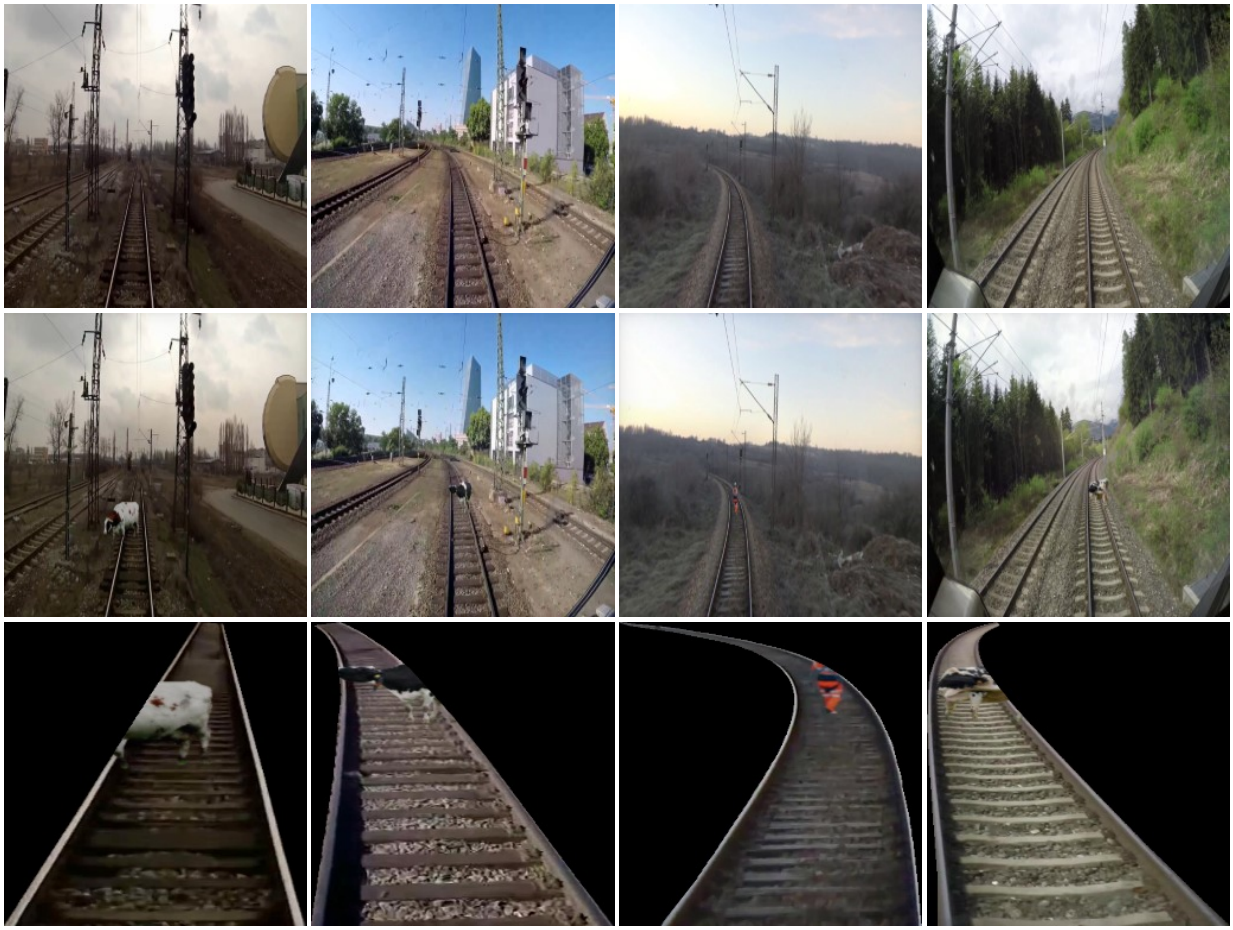

FIGURE 78: Aperçu du dataset de validation. La première ligne d'images montre les images de validation sans obstacles (backgrounds). La deuxième ligne d'images montre les mêmes backgrounds avec incrustation GP-GAN [149] (incrustations). La troisème ligne montres les RoIs d'images incrustées.

Un résumé du dataset de validation des RoIs sans et avec obstacles est décrit au niveau de la table 4.4

Dataset	Nombre d'images	Utilisation
Validation	271	Pre-Entraînement
Backgrounds	19	Post-Entraînement
Incrustations	10-20 pour chaque Background	Post-Entraînement

TABLE 4.4: Résumé du dataset de validation généré. On distingue un set de validation qui se compose uniquement d'images sans obstacles afin de juger les performances du modèle. De plus, à l'aide du modèle GP-GAN [149] on fixe 19 types de backgrounds, chacun avec un nombre d'incrustations qui varie entre 10 et 20.

4.4 Résultats expérimentaux

Dans cette section, nous allons examiner les résultats de l'entraînement des 240 autoencodeurs convolutionnels dans notre étude exploratoire. Par la suite, nous sélectionnerons

les 10 meilleurs modèles en utilisant une méthode de classement appelée "Technique for Order of Preference by Similarity to Ideal Solution" (TOPSIS) [148]. TOPSIS [148] classe les différentes alternatives en fonction de leur proximité à une solution idéale et de leur éloignement d'une solution anti-idéale.

Afin de juger la performance des modèles, nous introduisons le critère du gap-score. Ce critère permet de vérifier si le modèle maximise le score d'erreur lorsqu'il rencontre un obstacle dans la scène. L'idée derrière le gap-score est d'évaluer la capacité du modèle à identifier et réagir aux éléments problématiques ou inattendus dans les données.

Enfin, nous testerons notre meilleur modèle sur des cas réels et des cas synthétiques pour vérifier leur efficacité et leur applicabilité dans divers scénarios.

4.4.1 Critères de sélection et gap-score

Nous désignons par "*background*" l'image normale sans obstacles. Nous désignons également par "*incrustation*" la même image normale (background) avec des obstacles incrustés. Soit $\mathcal{X} = \{x_1, x_2, x_3, \ldots, x_n\}$ l'ensemble de données d'images de background contenant N images et $\hat{\mathcal{X}} = \{\hat{x}_{1,1}, \hat{x}_{1,2}, \hat{x}_{2,1}, \hat{x}_{2,2}, \ldots, \hat{x}_{n,m}\}$ l'ensemble de données d'images incrustées contenant M images avec $M \geq N$. Chaque image de background x_i peut avoir au moins une ou plusieurs contreparties incrustées. Un modèle performant est un modèle qui maximise les deux critères, C_1 et C_2, décrits ci-dessous :

— Pour chaque image de background et ses contreparties incrustées générées, le modèle doit maximiser la moyenne du gap-score (gap-score) entre chaque image incrustée et son fond original :

$$s_i = d(x_i, x_i'),\ \forall x_i \in X \tag{4.23}$$

$$s_{i,j} = d(\hat{x_{i,j}}, \hat{x_{i,j}'}),\ \forall x_i \in X, \forall x_j \in \hat{X} \tag{4.24}$$

$$C_1 = \frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{s_{i,j} - s_i}{s_i} \tag{4.25}$$

Où x_i et x_i' sont respectivement une image normale et sa reconstruction par un modèle d'autoencodeur convolutionnel donné en utilisant une métrique unique d. $\hat{x_{i,j}}$ et $\hat{x_{i,j}'}$ sont respectivement une contrepartie incrustée de l'image x_i et sa reconstruction par un modèle d'autoencodeur convolutionnel donné en utilisant une métrique unique d. Ici, d est l'erreur moyenne quadratique (MSE) [231]. s_i et $s_{i,j}$ sont respectivement les scores MSE pour une image de background et sa contrepartie incrustée. Nous désignons par C_1 le critère du gap-score moyen.

— Pour chaque background, un modèle doit maximiser le nombre de scores d'écart

positifs générés pour chaque background :

$$C_i = \begin{cases} 1, & \text{si } \sum_{j=1}^{m} \frac{s_{i,j} - s_i}{s_i} \times 100 > 0, \forall\ x_i \in X \\ 0, & \text{autrement} \end{cases} \tag{4.26}$$

$$C_2 = \sum_{i=1}^{n} C_i \tag{4.27}$$

Nous désignons par C_2 le critère des scores positifs.

4.4.2 Méthode Multi-critères TOPSIS

Afin d'évaluer les modèles générés à partir de notre étude exploratoire, nous utilisons les méthodes d'aide à la décision multicritère (MCDM), en particulier TOPSIS [148]. La matrice d'évaluation $(x_{ij})_{m\times n}$ est composée de n critères et de m alternatives. Dans notre cas, les critères sont C_1 et C_2, et les alternatives sont les modèles générés. Nous fixons une stratégie de maximisation pour savoir quel modèle maximise les deux critères en fonction des poids attribués à chaque critère. Enfin, nous calculons la distance d entre chaque alternative, la solution idéale et anti-idéale extraite de la matrice d'évaluation pour attribuer un score à chaque alternative afin de les classer selon leur distance à la solution idéale.

Nous résumons les étapes utilisées dans TOPSIS [148] comme suit :

— Considérons $T = (x_{ij})_{m\times n}$ notre matrice d'évaluation composée de $m = 240$ et $n = 2$ où m est le nombre de modèles générés et n le nombre de critères. Nous normalisons la matrice comme suit :

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{j=1}^{m} x_{ij}^2}}, i = 1, ..., m; j = 1, ..., n. \tag{4.28}$$

— Multipliez les colonnes de la matrice de décision normalisée par les poids associés pour obtenir la matrice de décision pondérée :

$$v_{ij} = w_j . r_{ij}, i = 1, ..., m; j = 1, ..., n \tag{4.29}$$

Sachant que nous avons deux critères, nous choisissons deux poids, w_1 et w_2 où $w_1 + w_2 = 1$.

— Déterminez les solutions idéales et anti-idéales. La solution idéale, notée A^+, et la solution anti-idéale, notée A^-, sont définies comme suit :

$$A^+ = \{v_1^+, v_2^+, .., v_n^+\} \tag{4.30}$$

$$A^- = \{v_1^-, v_2^-, ..., v_n^-\} \tag{4.31}$$

Avec $\{v_1^+, v_2^+, .., v_n^+\}$ et $\{v_1^-, v_2^-, ..., v_n^-\}$ les critères des individus les plus maximisés, respectivement les plus minimisés.

— Ensuite, nous calculons la distance L^2 entre l'alternative cible i et la solution idéale et anti-idéale respectivement :

$$S_i^+ = \sqrt{\sum_{j=1}^{n}(v_j^+ - v_{ij})^2}, i = 1, ..., m; j = 1, ..., n. \tag{4.32}$$

$$S_i^- = \sqrt{\sum_{j=1}^{n}(v_j^- - v_{ij})^2}, i = 1, ..., m; j = 1, ..., n. \tag{4.33}$$

— Enfin, nous calculons la similarité avec la solution idéale :

$$S_i^* = \frac{S_i^-}{S_i^- + S_i^+} \tag{4.34}$$

La méthode TOPSIS a été appliquée pour définir un ordre de préférence parmi les modèles, en se basant sur les critères définis dans la section 4.4.1.

4.4.3 Hyperparamètres

En termes d'architecture de l'autoencodeur convolutionnel, nous avons fixé le nombre de couches, la profondeur, les hyperparamètres, le suréchantillonnage (upsampling), le maxpooling, les convolutions transposées, les convolutions échelonnées et la taille du goulot d'étranglement comme indiqué au niveau de la figure 60. Les parties encodeur et décodeur se composent respectivement de 6 couches convolutionnelles pour sous-échantillonner et sur-échantillonner l'image. Pour chaque couche de l'encodeur, nous utilisons des convolutions échelonnées (stride) pour réduire la taille de l'image de 2 à chaque étape. Avec chaque couche, nous fixons les filtres de 32 à 128 avec des noyaux de taille 3x3, sauf pour les couches du goulot d'étranglement qui ont 48 filtres. Pour chaque couche de la partie décodeur, nous utilisons le suréchantillonnage au lieu de la convolution transposée et des filtres allant de 128 à 32 avec des noyaux de taille 3x3. Le but de cette partie est d'obtenir l'architecture adéquate en fonction du jeu de données d'entraînement utilisé pour initier notre étude exploratoire. En utilisant la même architecture, nous générons à chaque fois un modèle suivant une combinaison des composants de notre étude exploratoire, notamment les fonctions de pertes, les fonctions d'activations et aussi les optimiseurs utilisés.

Concernant les hyperparamètres des fonctions d'activation, MSE [231], MAE [231] et PSNR [232] sont sans paramètres. Seuls SSIM [233] et MS-SSIM [233] sont concernés. Nous avons fixé leurs paramètres aux paramètres par défaut recommandés dans leurs articles

originaux. Chaque fonction d'activation est couplée à une initialisation glorot uniforme [245] à l'exception de selu [240], qui est couplé à une initialisation Lecun normale [240].

Concernant les optimiseurs utilisés, nous distinguons différentes configurations pour chaque optimiseur utilisé. Pour Adam [234] nous utilisons une seule configuration 'brute' avec un taux d'apprentissage = 10^{-3}. Pour RAdam [235] nous fixons deux configurations différentes, une configuration 'brute' avec les paramètres suivants : taux d'apprentissage = 10^{-3}, pas = 0, échauffement = 10^{-1} et taux d'apprentissage minimum fixé à 0. L'autre configuration est paramétrée avec un taux d'apprentissage = 10^{-3}, pas = 10^{5}, échauffement (warmup) = 10^{-1} et taux d'apprentissage minimum fixé à 10^{-5}. Pour Novograd [236] nous distinguons trois types de configurations avec les paramètres suivants, une configuration 'brute' avec un taux d'apprentissage = 10^{-3}, décroissance = 0 et sans utilisation de la moyenne des gradients. Une seconde configuration paramétrée avec un taux d'apprentissage = 10^{-3}, décroissance = 10^{-1} avec moyenne des gradients et la dernière configuration pour Novograd [236] est une configuration paramétrée moyennée avec un taux d'apprentissage = 10^{-3}, décroissance = 10^{-3} avec moyenne des gradients.

Chaque modèle généré utilise la même architecture et les configurations décrites ci-dessus, et prend en compte des inputs d'images de résolution 256x256 adaptés au dataset décrit dans la section 4.3.4. Les modèles générés sont entraîné pendant 300 époques.

La table 4.5 résume les hyperparamètres choisis au niveau de l'entraînement des modèles.

TABLE 4.5: Résumé des hyperparamètres

Composant	**Hyperparamètre**	**Valeur**
Autoencodeur Convolutionnel	Profondeur (encodeur-decodeur)	6 couches
	Taille des filtres	32 à 128
	Taille des noyaux	3x3
	Goulot d'étranglement	48 filtres
	Epoques d'entraînement	300
Fonctions d'activation	Paramètres par défaut	-
Adam	Taux d'apprentissage	10^{-3}
RAdam	Taux d'apprentissage	10^{-3}
	Pas	0, 10^{5}
	Échauffement	10^{-1}
	Taux d'apprentissage minimum	0, 10^{-5}
Novograd	Taux d'apprentissage	10^{-3}
	Décroissance	0, 10^{-1}, 10^{-3}
	Moyenne des gradients	Non, Oui, Oui

4.4.4 Résultats

Dans cette section, nous présentons les résultats de notre étude exploratoire sur les images des backgrounds et leur contrepartie incrustée. Ensuite, nous montrons les résultats

de chaque modèle par rapport aux gap-scores calculés correspondants et leur classement en utilisant TOPSIS [148].

Pour simplifier les résultats de chaque modèle généré dans notre étude, nous ne prenons que les 10 meilleurs modèles parmi les 240 modèles. De plus, pour simplifier la présentation des résultats, une convention de nommage est utilisée pour chaque modèle afin de le combiner avec les composants correspondants comme suit : optimiseurs_perte_activation. Pour les optimiseurs, nous distinguons les conventions de nommage suivantes :

— radam_[par, raw] : Ici, l'étiquette 'par' signifie l'optimiseur RAdam personnalisé avec la configuration paramétrée et 'raw' signifie la configuration brute.

— novograd_[par, raw]_[avg] : Ici, l'étiquette 'par' signifie l'optimiseur Novograd avec la configuration paramétrée, 'raw' signifie la configuration brute et 'avg' signifie l'utilisation de la moyenne des gradients.

4.4.4.1 Evaluation et classement des modèles

Cette section présente les résultats finaux générés. Elle représente également la sortie de la méthode TOPSIS [148] pour classer tous les modèles selon les critères C_1 et C_2. La colonne 'gap-score moyen' représente le premier critère montré dans la section 4.4.1 où nous calculons la moyenne de tous les scores d'écart pour chaque background, 'Scores positifs' représente le deuxième critère montré dans la section 4.4.1. En appliquant TOPSIS [148] sur ces deux critères, nous obtenons la troisième colonne 'Rang' afin de classer les meilleurs modèles du meilleur au pire.

Pour le premier tableau 4.6, nous observons que la majorité des modèles ont une combinaison de RAdam [234] comme optimiseur et psnr [232] comme fonction de perte, à l'exception du meilleur modèle qui a Adam [234] comme optimiseur et deux autres modèles ayant MSE [231] comme fonction de perte. En ce qui concerne les scores d'écart, tous les modèles ont des scores allant de 52,97% à 68,17%. En termes de scores positifs moyens de l'ensemble de test, aucun des 240 modèles générés n'atteint un score parfait de 19 arrière-plans positifs, le maximum atteint étant de 18 sur 19. Les poids correspondants de la méthode TOPSIS [148] sont d'égale importance pour les deux critères, c'est-à-dire $w_1 = 0.5$, $w_2 = 0.5$. Avec cela, la méthode tente de trouver les meilleurs modèles qui respectent un compromis égal entre les scores d'écart maximisés et les scores positifs respectivement. Finalement, nous avons observé une variété de fonctions d'activation pour chaque modèle telles que mila [243], relu [238], mish [241], elu [239], selu [240] et tanh [237].

Cependant, pour le deuxième tableau 4.7, en accordant plus d'importance aux scores positifs, c'est-à-dire $w_1 = 0.1$, $w_2 = 0.9$, nous observons l'émergence de nouveaux modèles ayant différentes combinaisons comme le couplage de RAdam [234] avec MS-SSIM [233] et swish [242], le couplage de RAdam [234] avec SSIM [233] ainsi que le couplage de Novograd [236] et MSE [231]. Ces mêmes modèles sont les seuls à avoir le plus grand nombre de scores positifs, à l'exception du modèle utilisant SSIM [233] qui atteint 18 sur

Modèles	gap-score moyen	Scores positifs	Rang
adam_psnr_mila	68.17%	17	1
radam_par_psnr_mila	66.62%	16	2
radam_par_mse_relu	62.88%	15	3
radam_par_psnr_mish	61.14 %	15	4
radam_raw_psnr_mila	61.14%	15	5
radam_raw_psnr_elu	57.42 %	14	6
radam_raw_psnr_selu	55.00%	16	7
radam_par_mse_tanh	53.62 %	17	8
radam_raw_mse_relu	53.12 %	15	9
radam_par_psnr_elu	52.97 %	14	10

TABLE 4.6: Résultats de l'évaluation basée sur les modèles générés (top 10) avec une importance égale pour les deux critères avec les poids suivants, $w_1 = 0.5$ pour les scores d'écart et $w_2 = 0.5$ pour les scores positifs.

19 scores positifs sur tous les arrière-plans, mais au coût d'un gap-score moyen allant de 36,93% à 44,62%. Nous observons la même tendance concernant le meilleur modèle qui reste le même lorsque l'on change les poids pour chaque critère. La figure 79 présente les résultats du modèle optimal ayant la configuration **'adam_psnr_mila'**.

4.4.4.2 Test sur un scénario réel

Nous testons notre modèle le mieux évalué avec la configuration **'adam_psnr_mila'** sur une vidéo [246] décrivant un événement rare d'un scénario réel d'obstacle ferroviaire. La vidéo montre un scénario où un cheval se trouve sur la voie ferrée avec le conducteur du train. Nous extrayons de la vidéo deux segments qui mettent en évidence à la fois l'obstacle et le conducteur au niveau de la piste. Nous extrayons ensuite les régions d'intérêt correspondantes de chaque image pour chaque segment afin de les préparer pour le modèle. Dans le but de montrer les résultats sous forme de gap-scores sur chaque image, nous calculons le score pour une seule image au début de la vidéo qui ne montre aucun signe d'obstacle afin de calculer le gap-score pour les autres images contenant des obstacles. Les figures 80 et 81 sont des diagrammes montrant les résultats du modèle le mieux évalué (premier au classement) en utilisant les gap-scores pour le premier et le deuxième segment respectivement. Chaque diagramme montre le gap-score de chaque image en pourcentage le long de l'axe des Y, et le numéro de l'image dans le segment le long de l'axe des X. Les deux résultats peuvent être trouvés au niveau des liens cités dans [248, 249].

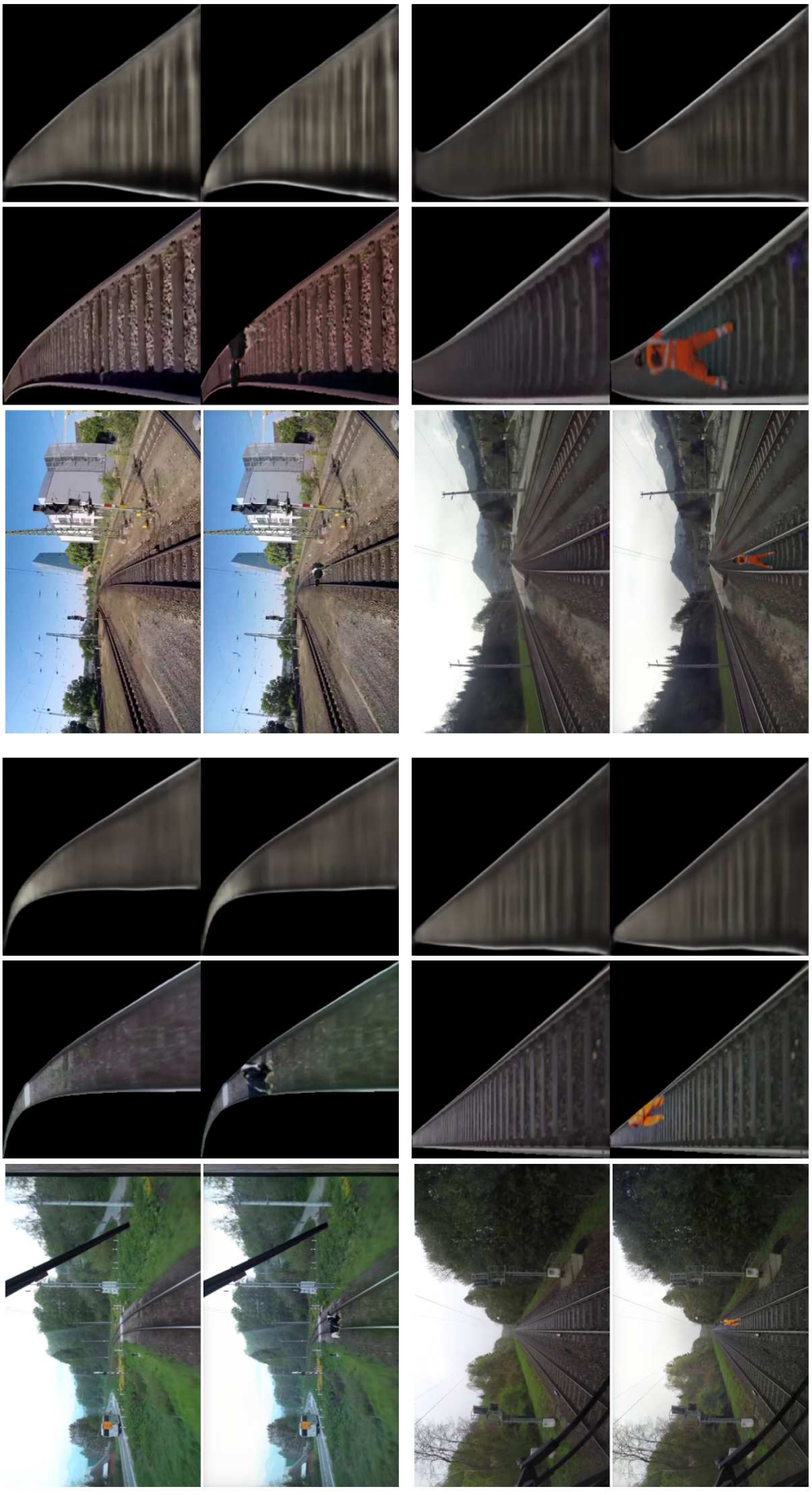

FIGURE 79: Aperçu des résultats qualitatifs montrant quatre scénarios différents. La première et la troisième ligne d'images montrent quatre exemples d'images de backgrounds sans aucun obstacle suivis de leur région d'intérêt et de la reconstruction de la même région par le meilleur modèle. La deuxième et dernière ligne d'images montrent les homologues incrustées des backgrounds précédemment montrés, chacun avec leurs régions d'intérêt respectives et les reconstructions par le même modèle..

Modèles	Scores d'écart moyens	Scores positifs	Rang
adam_psnr_mila	68.17%	17	1
radam_par_mse_tanh	53.62%	17	2
radam_par_msssim_relu	44.62%	18	3
radam_raw_psnr_mish	49.78%	17	4
radam_raw_ssim_relu	45.21%	17	5
radam_par_psnr_mila	66.62%	16	6
radam_raw_mse_mila	42.78%	17	7
novograd_par_avg_mse_tanh	36.93%	18	8
radam_par_msssim_swish	41.67%	17	9
adam_mse_tanh	40.72%	17	10

TABLE 4.7: Résultats de l'évaluation basée sur les modèles générés (top 10) avec plus d'importance accordée au nombre de scores positifs pour chaque arrière-plan avec les poids suivants, $w_1 = 0.1$ pour les scores d'écart et $w_2 = 0.9$ pour les scores positifs.



FIGURE 80: Résultat de détection sur le premier segment du travail proposé : l'axe x illustre l'ID des frames de la vidéo, tandis que l'axe y indique le gap-score en pourcentage (%)

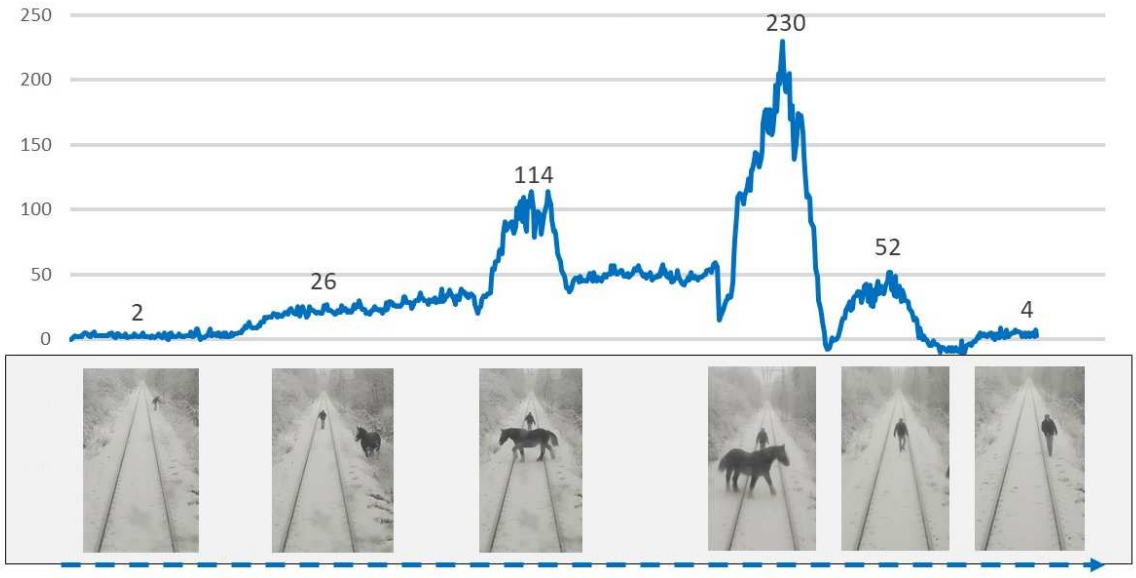

FIGURE 81: Résultat de détection sur le deuxième segment du travail proposé : l'axe x illustre l'ID des frames de la vidéo, tandis que l'axe y indique le gap-score en pourcentage (%)

En ce qui concerne les informations techniques, les modèles CAE issus de notre étude exploratoire ont été entraînés sur une station de travail équipée de 2 GPUs QUADRO P2000. Le temps total d'entraînement de l'ensemble des modèles a nécessité une période de 3 jours. Le temps d'inférence est de 20ms.

4.5 Conclusion

Dans ce chapitre, nous avons traité la problématique de la détection des obstacles ferroviaires sur la voie de circulation principale. Une architecture basée sur un autoencodeur convolutionnel a été mise en avant, complétée par une étude exploratoire qui a examiné divers aspects tels que les fonctions de pertes, d'activation et les optimiseurs. Le dataset a été préparé, prétraité et divisé en ensembles d'entraînement et de test à partir d'un dataset publique railsem19. Des techniques comme la génération de boîtes englobantes et l'extraction des clusters de la voie de circulation principale ont été employées pour déterminer les Régions d'intérêt (RoI). Les résultats expérimentaux ont été soumis à une série d'analyses, en utilisant des critères tels que le gap-score et la méthode Multi-critères TOPSIS. Toutefois, il est important de noter que bien que les autoencodeurs convolutionnels soient efficaces lorsqu'il s'agit de les utiliser sur des images bien définies avec des RoI et qui ne varient pas beaucoup, ils s'avèrent moins performants lorsqu'il s'agit de les utiliser sur un dataset varié, comportant plusieurs caractéristiques complexes. Cette observation nous pousse à envisager des améliorations pour ces modèles, qui seront discutées au niveau du chapitre 5, afin d'assurer le monitoring de l'environnement du train de fret autonome.

Chapitre 5

Détection au niveau de la scène ferroviaire

5.1 Introduction

Dans le chapitre précédent, une solution initiale pour la détection d'anomalies basée sur les autoencodeurs convolutionnels a été réalisée. Dans ce chapitre, nous présentons la troisième contribution : l'intégration des transformeurs de vision (ViT) dans une architecture d'autoencodeur, afin de combler les faiblesses de la solution autoencodeurs convolutionnels. En effet, une surveillance de haute qualité du milieu ferroviaire requiert un jeu de données exhaustif, couvrant l'ensemble des situations courantes sur les rails ainsi qu'au niveau de l'emprise ferroviaire. Les autoencodeurs convolutionnels sont appréciés pour leur habilité à analyser des images nettes, axées sur des régions d'intérêt stables en termes de propriétés visuelles. Le but étant de définir clairement ce qui est considéré comme "normal" de ce qui est "anormal". Cependant, face à des jeux de données étendus et variés, avec une conception changeante de la normalité, ces autoencodeurs convolutionnels peuvent montrer certaines limites.

Ce chapitre propose l'utilisation des Transformeurs de Vision (ViT) [134] afin de construire une architecture d'autoencodeur hybride. Cette dernière est conçue avec des ViTs intégrés dans la partie encodeur et des couches de convolutions dans la partie décodeur. La combinaison mise en avant cherche à tirer le meilleur des deux techniques : les ViTs pour extraire des caractéristiques complexes et maintenir le contexte de la normalité en matière de détection d'anomalies, même lorsque confrontés à des données hétérogènes qui varient considérablement. D'autre part, les couches convolutionnelles contribuent à alléger l'architecture tout en offrant une représentation précise des données. En intégrant ces deux approches, l'objectif est de répondre de manière efficace aux défis posés par la détection d'anomalies dans des jeux de données vastes et hétérogènes, en capitalisant sur les atouts de chaque méthode. Une analyse des jeux de données sera effectuée, suivie d'une revue des performances obtenues, afin de souligner les potentialités et les implications de la méthode hybride autoencodeurs convolutionnels avec les ViTs.

5.2 Composante transformeur de vision

Les transformeurs ont été introduits pour la première fois dans l'article [251]. Ils reposent sur des mécanismes d'auto-attention, qui permettent à chaque élément d'une séquence de pondérer l'importance des autres éléments en fonction de leur pertinence contextuelle. Cette capacité à capturer des dépendances à longue portée dans les données sans se limiter à une fenêtre locale a fait des transformeurs un outil puissant et flexible. C'est cette propriété qui leur a valu une adoption assez répandue dans le domaine. L'article original mettait principalement l'accent sur le traitement automatique du langage naturel [252]. C'est dans ce contexte que les transformeurs ont été le plus couramment utilisés dans la littérature. Cependant, leur application ne se limite pas à cette seule utilisation, et ils ont démontré leur pertinence dans d'autres types de données, comme les images [256].

En effet, les transformeurs de vision ou ViT (*Vision Transformers*) [134] ont montré des résultats prometteurs dans divers domaines, notamment la reconnaissance d'images, la détection d'objets et la segmentation [254]. Étonnamment, certains de ces transformeurs parviennent à égaler ou surpasser les résultats des méthodes de pointe en s'appuyant uniquement sur l'auto-attention, sans avoir recours aux réseaux neuronaux convolutionnels. De plus, contrairement aux CNNs, les transformeurs offrent l'avantage de permettre une parallélisation pour les données séquentielles.

5.2.1 Définition de l'attention dans les transformeurs

Le concept d'attention dans les réseaux neuronaux artificiels désigne la capacité d'un modèle à se concentrer sélectivement sur certaines parties des données d'entrée lors de leur traitement. Ces mécanismes d'attention permettent aux modèles de pondérer et de combiner dynamiquement différentes parties des données d'entrée en fonction de leur pertinence ou importance pour la tâche concernée [255].

Plusieurs types de mécanismes d'attention ont été développés pour une utilisation dans les réseaux neuronaux. Parmi les plus connus on trouve le mécanisme d'auto-attention. Il fonctionne en projetant les données d'entrée dans un espace de dimension supérieure où les points de données sont représentés sous forme de vecteurs. Le modèle calcule par la suite les produits scalaires entre ces vecteurs, représentant la similarité entre chaque paire de points de données. Ces produits scalaires peuvent être calculés à l'aide de l'équation suivante :

$$\text{produit-scalaire} = x_i^T x_j \tag{5.1}$$

où x_i^T et x_j représentent respectivement le vecteur transposé du i-ème point de données et le vecteur du j-ème point de données.

Ces produits scalaires sont ensuite normalisés à l'aide d'une fonction softmax, qui les convertit en un ensemble de poids reflétant l'importance de chaque point de données par rapport aux autres. La fonction softmax est définie comme suit :

$$\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum \exp(x_j)} \tag{5.2}$$

où x_i est le i-ème produit scalaire et x_j est le j-ème produit scalaire.

En matière de vision par ordinateur, les mécanismes d'attention ont été utilisés pour permettre aux modèles de se concentrer sur certaines parties des données d'entrée qui sont pertinentes ou importantes pour la tâche à accomplir. Cela peut être particulièrement utile pour des tâches impliquant des données d'entrée ayant des structures complexes ou des dépendances à long terme, telles que la détection d'objets, la segmentation d'images ou la détection d'anomalies avec des données complexes.

Les mécanismes d'attention ont prouvé leur efficacité pour capturer les dépendances et relations à long terme dans les données et constituent un élément clé de nombreuses architectures modernes de réseaux neuronaux. Ils ont été largement utilisés dans les tâches de traitement du langage naturel et commencent également à être appliqués aux tâches de vision par ordinateur [253].

5.2.2 Le mécanisme de l'attention

Il existe plusieurs types de mécanismes d'auto-attention (*self-attention)* utilisés dans les transformeurs et autres architectures de réseaux neuronaux. Un type d'auto-attention est l'attention de produit scalaire qui fonctionne en calculant les produits scalaires entre les points de données d'entrée, comme décrit précédemment. Cette forme d'attention est relativement simple à mettre en œuvre, mais peut être limitée dans sa capacité à saisir des relations plus complexes dans les images d'entrées.

Un autre type d'auto-attention est l'attention multi-têtes. Elle consiste à diviser les données d'entrée en plusieurs "têtes" et à effectuer l'auto-attention indépendamment sur chaque tête. Le mécanisme d'auto-attention calcule alors les produits scalaires entre les points de données d'entrée, les normalise à l'aide de la fonction softmax et produit une somme pondérée des points de données d'entrée.

Pour effectuer l'attention multi-têtes, les données d'entrée sont d'abord divisées en H têtes, où H est le nombre de têtes. Le mécanisme d'auto-attention est ensuite appliqué indépendamment à chaque tête, produisant H sommes pondérées des points de données d'entrée. Ces H sommes pondérées sont ensuite concaténées et transformées à l'aide d'une couche linéaire, produisant une seule sortie combinée. La sortie combinée de l'attention multi-têtes peut être calculée à l'aide de l'équation suivante :

$$\text{sortie} = \text{transformation_linéaire}(\text{concat}(\text{tête}_1, \text{tête}_2, \ldots, \text{tête}_H)) \tag{5.3}$$

où tête_i est la somme pondérée pour la i-ème tête et transformation_linéaire est une transformation linéaire (par exemple, une couche entièrement connectée) qui transforme les têtes concaténées en sortie finale.

L'attention multi-têtes présente plusieurs avantages par rapport à l'auto-attention mono-tête. D'abord, elle permet au modèle de se concentrer sur plusieurs parties des données d'entrée simultanément, ce qui peut être utile pour des tâches nécessitant une compréhension globale des données d'entrée. Ensuite, elle permet au modèle d'apprendre plusieurs motifs d'attention différents, utiles pour des tâches avec des relations complexes dans les données d'entrée. Enfin, l'attention multi-têtes peut améliorer l'expressivité du modèle, car elle permet au modèle d'apprendre des combinaisons plus complexes des données d'entrée.

L'attention multi-têtes est un composant clé de l'architecture du transformeur. Cette solution a connu un grand succès dans les tâches de traitement du langage naturel. Elle a également été appliquée à des tâches de vision par ordinateur, où elle est utilisée dans l'architecture du transformeur de vision. On peut résumer le fonctionnement de ces composantes comme suivant :

Comme vu précédemment, nous aborderons ici deux principes fondamentaux des transformeurs :

— **Auto-attention** : Pour des matrices d'interrogation Q, de clé K, et de valeur V, l'attention est déterminée en prenant le produit scalaire de Q et K, en le mettant à l'échelle, puis en utilisant une fonction softmax avant de pondérer V.

$$\text{SA}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5.4}$$

— **Auto-attention multi-têtes** : Cette approche utilise plusieurs "têtes" ou séries de matrices Q, K, V pour obtenir diverses représentations de l'attention, qui sont ensuite concaténées pour produire la sortie finale.

$$\text{MSA}(Q, K, V) = \text{Concat}(\text{SA}_1, \text{SA}_2, ..., \text{SA}_h) \tag{5.5}$$

La définition des matrices Q, K, et V dans les architectures basées sur des transformeurs est la suivante :

— **Interrogation (Q)** : L'interrogation est un vecteur de caractéristiques décrivant l'information recherchée dans la séquence ou l'ensemble de données. Cette matrice sert à déterminer quels éléments ou aspects des données devraient être mis en avant.

— **Clés (K)** : Chaque élément ou segment des données possède une clé, un vecteur de caractéristiques définissant les attributs ou les propriétés de cet élément. Les clés évaluent la pertinence de chaque élément par rapport à l'interrogation.

— **Valeurs (V)** : Associé à chaque élément ou segment des données se trouve un

vecteur de valeurs. Ces vecteurs contiennent les informations que le mécanisme d'attention souhaite agréger ou mettre en évidence en fonction de la pertinence déterminée par l'interaction entre l'interrogation et les clés.

Dans le cadre des applications visuelles, ces concepts sont transposés à la manipulation d'images. Les ViTs découpent généralement une image en patchs, chacun représenté par des matrices d'interrogation, de clé et de valeur. Grâce à ce mécanisme, le modèle peut mettre en évidence des caractéristiques spécifiques ou capturer des relations spatiales entre différentes parties de l'image, offrant ainsi une analyse approfondie des caractéristiques visuelles.

Les figures 82 et 83 représentent respectivement le mécanisme d'auto-attention par produit scalaire ainsi que le mécanisme d'attention multi-têtes utilisant la même composante par produit scalaire.

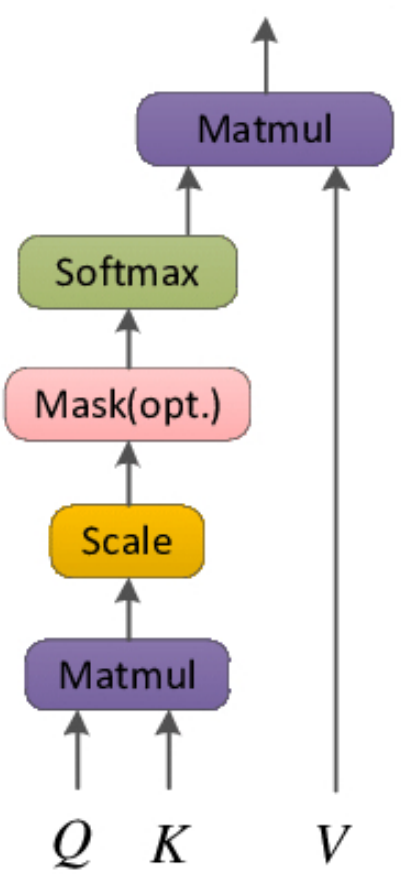

FIGURE 82: Tête d'auto-attention par produit scalaire (Scaled Dot-product Self Attention head)

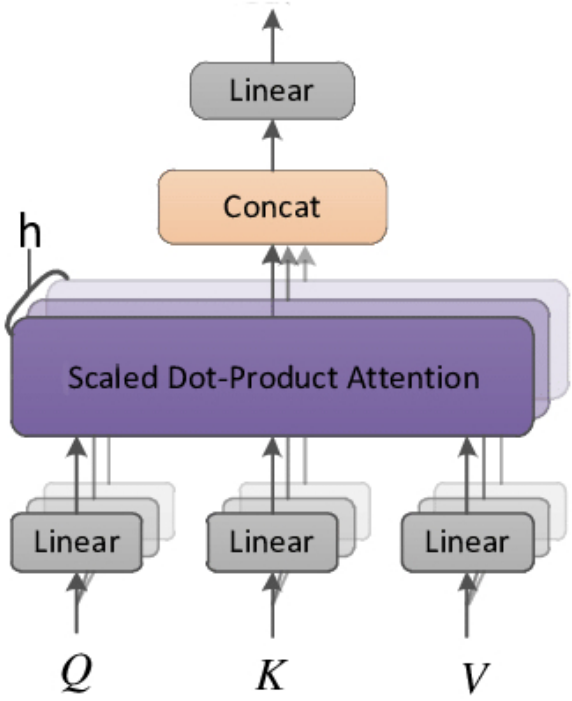

FIGURE 83: Auto-attention multi-têtes (Multi-head self-attention) utilisant plusieurs têtes d'auto-attention par produit scalaire

5.3 Autoencodeur à base de transformeur de vision

Le modèle hybride proposé dans ce travail combine les techniques traditionnelles basées sur la reconstruction et les avantages d'une méthodologie basée sur des patchs. Le processus commence par subdiviser l'image d'entrée en patchs, qui sont ensuite encodés via un transformeur de vision. Par la suite, les caractéristiques dérivées sont canalisées dans un décodeur pour reconstruire l'image initiale. Ceci oblige le réseau à assimiler des caractéristiques représentatives des caractéristiques normales, c'est-à-dire les caractéristiques issues des images de contexte normal sans anomalies, de l'image, basées exclusivement sur les données formées. Parallèlement, un réseau de densité de mélange gaussien évalue la distribution des caractéristiques encodées par le transformeur pour déduire la distribution des données standard dans l'espace latent associé. Un avantage supplémentaire de ce modèle est sa capacité intrinsèque à localiser les anomalies, du faite que les caractéristiques encodées par le transformeur conservent des données positionnelles.

5.3.1 Encodeur ViT

L'encodeur est conçu en s'inspirant directement des modèles de transformeurs de vision existants [256]. Il reçoit en entrée les patchs, notés $E \in \mathbb{R}^{(N+1)\times D}$. Ces représentations apprennent les relations entre chaque patch grâce à k opérations de SA, plus précisément, l'auto-attention multi-têtes MSA. Les équations ci-dessous décrivent le processus

mathématique sous-jacent :

$$[q, k, v] = EU_{qkv}, \quad U_{qkv} \in \mathbb{R}^{D \times 3D_h} \tag{5.6}$$

où q, k et v sont les vecteurs de requête, de clé et de valeur, respectivement. Ils sont dérivés des représentations vectorielles (*embeddings*) E et d'une matrice de transformation U_{qkv}.

$$A = \text{softmax}\left(\frac{qk^T}{\sqrt{D_h}}\right), \quad A \in \mathbb{R}^{(N+1)\times(N+1)} \tag{5.7}$$

Ici, A représente les poids d'attention, qui sont calculés en utilisant la fonction softmax sur le produit scalaire des requêtes et des clés, normalisé par la racine carrée de la dimension D_h.

La sortie du mécanisme d'auto-attention est ensuite calculée en multipliant les poids d'attention A par les vecteurs de valeur v.

$$\text{SA}(E) = A \times v \tag{5.8}$$

Pour l'auto-attention multi-têtes :

$$\text{MSA}(E) = [\text{SA}_1(E), \text{SA}_2(E), ..., \text{SA}_k(E)]U_{msa}, \quad U_{msa} \in \mathbb{R}^{k \times D_h \times D} \tag{5.9}$$

Dans le mécanisme d'auto-attention multi-têtes, les représentations vectorielles sont divisées et traitées simultanément à travers plusieurs têtes d'auto-attention, spécifiquement désignées par $\text{SA}_1, \text{SA}_2, ..., \text{SA}_k$. Après cette étape, les sorties de ces différentes têtes sont concaténées, puis transformées en utilisant la matrice U_{msa}.

Après avoir défini les mécanismes d'auto-attention, nous pouvons maintenant décrire les principales étapes du transformeur de vision :

$$Z_0 = [X_1 pE; X_2 pE; ...; X_N pE] + E_{\text{pos}} \tag{5.10}$$

où

$$E \in \mathbb{R}^{P^2 \times C \times D}, \quad E_{\text{pos}} \in \mathbb{R}^{(N+1)\times D} \tag{5.11}$$

$$Z_0^l = \text{MSA}(\text{LN}(Z_{l-1})) + Z_{l-1}, \quad l = 1..L \tag{5.12}$$

$$Z_l = \text{MLP}(\text{LN}(Z_0^l)) + Z_0^l, \quad l = 1..L \tag{5.13}$$

Les représentations d'un patch sont obtenues en intégrant les informations des patchs avoisinants. Elles sont denses en informations et reflètent le contexte global de l'image grâce à l'encodeur ViT basé sur l'attention. Dans l'encodeur ViT, chaque représentation

de patch d'image est utilisée pour générer sa sortie, capturant ainsi des informations détaillées de chaque zone du patch.

La précision dans la détection et la localisation des anomalies nécessite une évaluation minutieuse de l'erreur de reconstruction au niveau de l'image et du pixel. Par conséquent, une représentation riche de l'image originale est essentielle pour estimer cette erreur lors de la reconstruction. Ainsi, l'encodeur est spécifiquement conçu pour extraire des représentations de patchs riches en détails de chaque segment d'image.

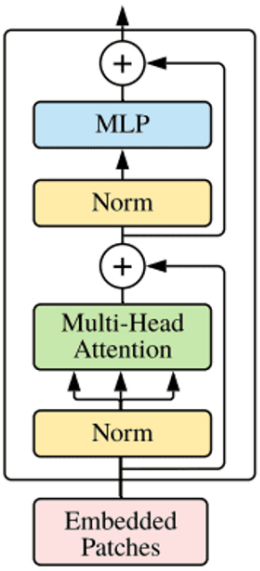

FIGURE 84: Représentation d'un bloc ViT

.

5.3.2 Décodeur

Le rôle du décodeur est de convertir le vecteur latent, issu de l'espace $\mathbb{R}^{768}$ et représentant la sortie de l'encodeur ViT, en une image de dimensions $\mathbb{R}^{384\times384\times3}$, correspondant à la taille originale de l'image d'entrée. Ce modèle est composé de cinq couches convolutionnelles transposées, complétées par des couches de normalisation (batchnorm) et des fonctions d'activation ReLU. Quant à la couche terminale, elle utilise une fonction d'activation tanh.

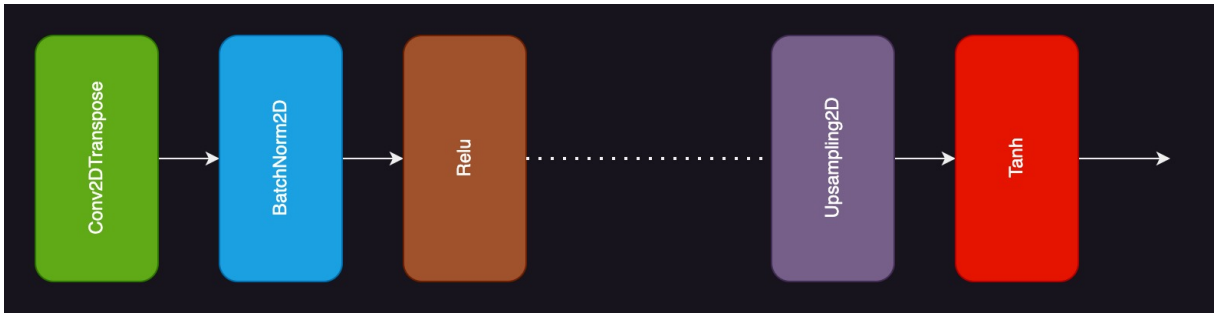

FIGURE 85: Représentation du decodeur à base de convolutions

.

5.3.3 Architecture globale

L'architecture proposée associe un autoencodeur basé sur le Vision Transformer (ViT) et un décodeur convolutionnel. L'encodeur ViT traite une image en segments, les convertit en vecteurs, puis utilise des mécanismes d'attention pour capter des informations détaillées de l'image. Le décodeur convolutionnel se charge ensuite de la reconstitution de l'image à partir de ces informations. La méthode repose sur la mesure des différences entre l'image originale et sa version reconstruite. Pour chaque pixel de l'image, une distance, définie par la norme l2, est déterminée entre l'image initiale et celle obtenue par le modèle hybride. Cette distance peut être représentée par l'équation :

$$\text{Erreur}(i,j) = |\text{Pixel original}(i,j) - \text{Pixel reconstruit}(i,j)|^2$$

En moyennant ces erreurs sur tous les canaux, une "heatmap d'erreur" est générée. Cette heatmap est cruciale pour la détection d'anomalies. Une image standard présentera une faible erreur de reconstruction, tandis qu'une image avec des anomalies montrera des zones à forte erreur. La localisation des anomalies se fait en identifiant ces zones à forte erreur sur la heatmap. La valeur maximale de cette heatmap donne une indication de la présence d'anomalies, tandis que l'ensemble de la heatmap aide à localiser ces anomalies dans l'image.

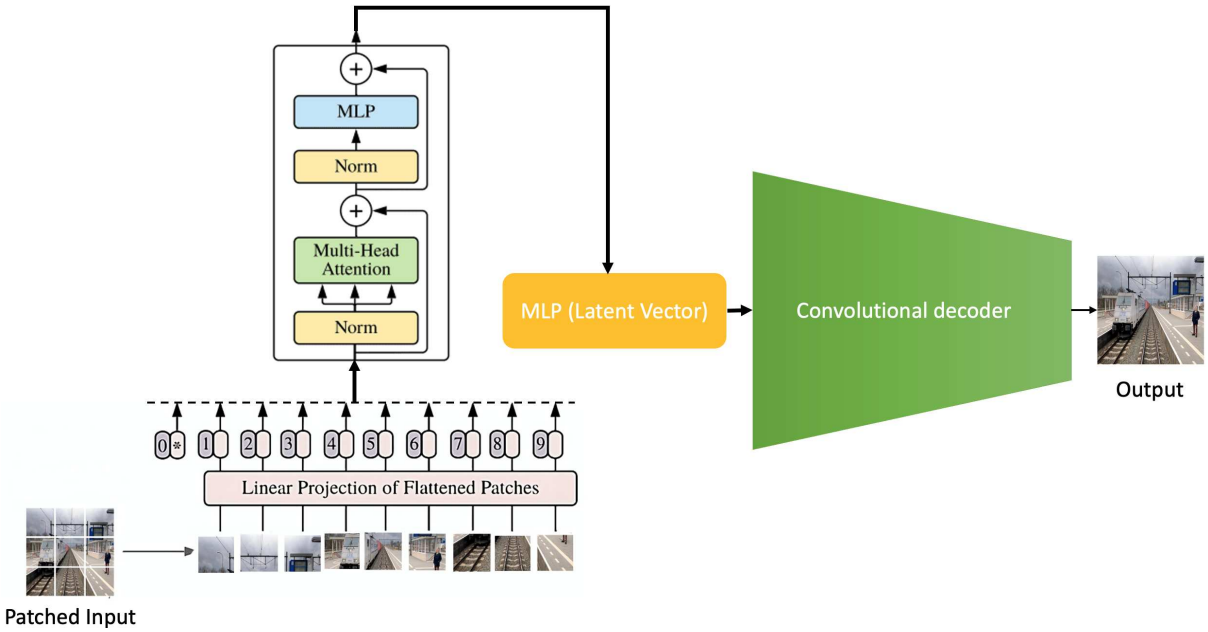

FIGURE 86: Architecture globale hybride ViT

.

5.4 Datasets

Le dataset dédié aux scènes ferroviaires a été rassemblé au cours de nombreuses phases d'essai, ici désignées comme des *baselines*. Dans ce contexte, une baseline désigne une phase d'essai durant laquelle la locomotive est testée avec ses capteurs pour confirmer leur bon fonctionnement et précision. Ces phases d'essai fournissent un point de référence essentiel pour mesurer et comparer les résultats des différents tests réalisés. Le principal objectif de ces phases d'essai était d'examiner la capacité opérationnelle de la locomotive et l'efficience de ses capteurs. Pour cela, une multitude de situations ont été reproduites et étudiées. Suite à ces tests, le dataset final, composé de plus de 100 enregistrements vidéo, offre un large panel représentatif pour des études ultérieures où chaque enregistrement dure entre 20 et 90 minutes.

5.4.1 Dataset d'entraînement

La constitution du dataset d'entraînement a nécessité plusieurs étapes clés, notamment l'extraction d'images des vidéos acquises. Les vidéos ont d'abord été organisées par catégorie pertinente. Puis, des images ont été extraites de chaque vidéo, à une fréquence d'une image toutes les 2 secondes.

Lorsque le train est immobile en gare, l'Erreur Quadratique Moyenne (MSE) a été utilisée pour distinguer les images. Cette mesure se calcule comme suit :

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(I_{\text{actuelle}} - I_{\text{suivante}})^2 \tag{5.14}$$

Si la MSE dépasse un seuil S, cela indique que le train est en mouvement et l'image est conservée. Autrement, elle est écartée. Afin d'éliminer efficacement les images redondantes, les images sont d'abord normalisées et converties en niveaux de gris (greyscale). Cette étape de normalisation et de conversion est effectuée exclusivement pour faciliter la comparaison et l'élimination des redondances. Ce mécanisme permet d'optimiser le dataset. Dans cette étude, le seuil S est fixé à 0.05.

Diverses transformations ont été ensuite appliquées aux images :

1. **Flou gaussien** : Technique utilisée pour atténuer le bruit, définie par l'équation :

$$G(x, y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{5.15}$$

2. **Cropping** : Isolation des parties essentielles de l'image, notamment les rails et l'emprise ferroviaire.
3. **Redimensionnement** : Adaptation des images à une taille standard pour faciliter leur traitement.

Le dataset obtenu comprend 10 624 images de scènes ferroviaires ayant des dimensions de 384x384 pixels. La figure 87 présente des images brutes provenant d'une baseline donnée. La figure 88 illustre le résultat du prétraitement de cette même baseline, tandis que la figure 89 expose d'autres exemples issus de différentes baselines.



FIGURE 87: Exemples d'images à partir des acquisitions vidéos des baselines



FIGURE 88: Exemples d'images du dataset après prétraitement

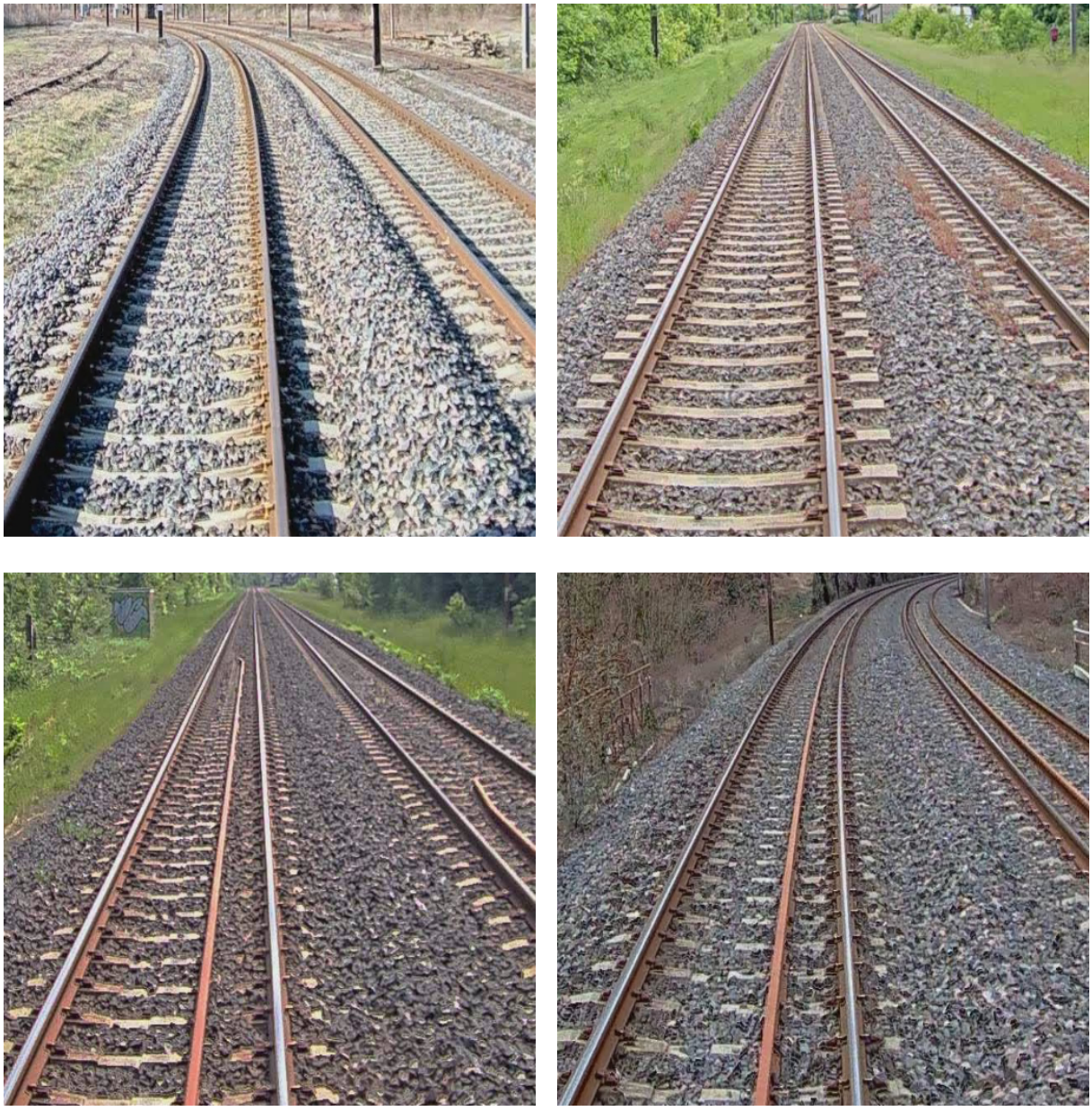

FIGURE 89: Autres exemples d'images du dataset d'entraînement

5.4.2 Dataset de test

Notre dataset d'entraînement a été méticuleusement conçu en intégrant une vaste gamme de cas normaux, englobant diverses conditions météorologiques et autres situations courantes dans le milieu ferroviaire. Cependant, pour le dataset de test, la collecte de scénarios avec des obstacles s'avère plus délicate. Toutefois, nous avons capitalisé sur les vidéos des acquisitions baselines. De ces vidéos, nous avons extrait des événements rares, tels que la présence de vaches ou d'autres animaux traversant la voie. De plus, pour enrichir ce dataset, nous avons manuellement intégré des obstacles sur la voie de circulation, comme des chariots, des torches à flamme rouge, ou encore la présence de travailleurs SNCF sur l'emprise ferroviaire. Le dataset de test résultant regroupe 1 000

images d'anomalies. Le tableau 5.1 donne une vue d'ensemble des vidéos utilisées, leurs identifiants, les segments exploités, ainsi que les événements ou anomalies associés.

Vidéo de référence	ID	Début crop	Fin crop	Événement/Anomalie
2021-10-18(BL8)	20211018_145824	-	-	Aucun
2021-10-18(BL8)	20211018_155825	-	-	Aucun
2021-10-18(BL8)	20211018_165825	21 :18	21 :38	Présence de vache
2021-10-18(BL8)	20211018_175825	28 :42	28 :59	Torches à flammes rouge
2021-10-19(BL8)	20211019_095026	00 :45	00 :55	Travailleur SNCF
2021-10-19(BL8)	20211019_102026	10 :56	11 :03	Chariot
2021-10-19(BL8)	20211019_105026	26 :50	27 :03	Voiture
2021-10-19(BL8)	20211019_122647	-	-	Aucun
2021-10-19(BL8)	20211019_125647	19 :06	19 :35	Mannequin humain
2021-10-19(BL8)	20211019_132647	-	-	Aucun
2021-10-20(BL8)	20211020_091335	00 :00	30 :00	Animal traversant la voie
2021-10-20(BL8)	20211020_094335	02 :02	06 :49	Travailleur SNCF
2021-10-21(BL8)	20211021_161714	-	-	Condition météo : Pluie
2021-10-21(BL8)	20211021_162714	-	-	Condition météo : Pluie
2022-01-24 (B8bis)	20220124_094017	03 :43	03 :46	Torches à flammes rouge
2022-01-24 (B8bis)	20220124_174912	-	-	Scène de nuit
2022-01-25(B8bis)	20220125_075558	-	-	Tache sur la caméra
2022-01-26(B8bis)	20220126_091111	-	-	Aucun
2022-04-25 (B9)	20220425_100415	-	-	Tache sur la caméra
2022-04-25 (B9)	20220425_100415	-	-	Aucun

TABLE 5.1: Synthèse des segments d'anomalies et événements divers dans les vidéos des baselines 8, 8 Bis et 9

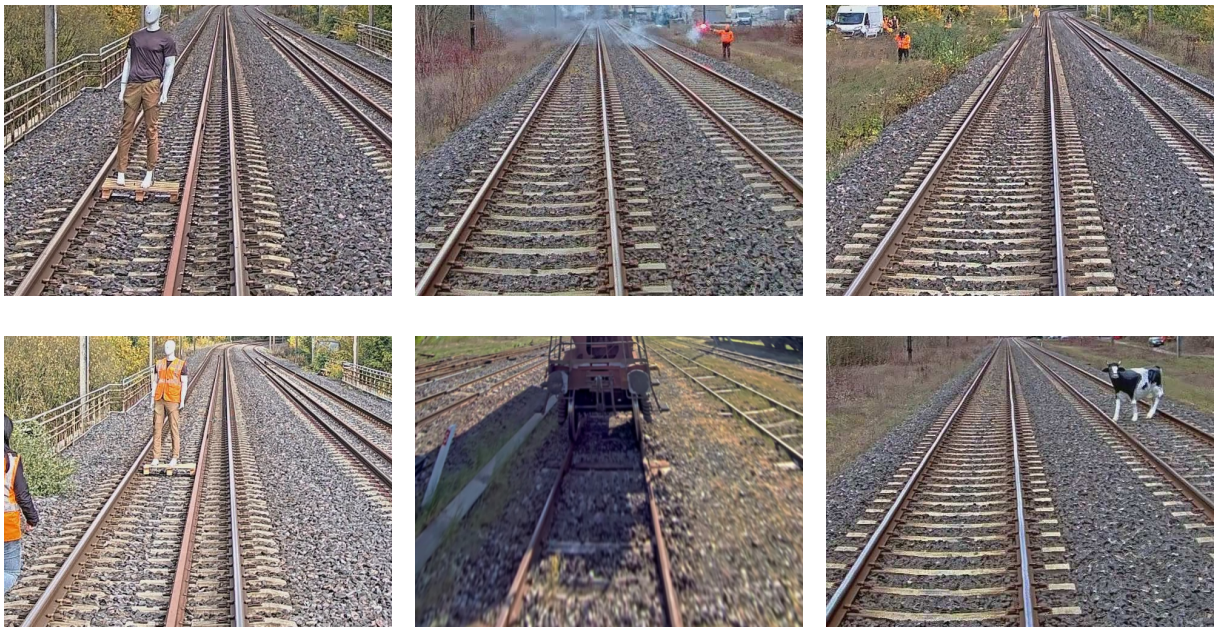

FIGURE 90: Exemples d'images prétraitées illustrant le dataset de test

5.5 Résultats expérimentaux

La performance du modèle hybride, préalablement entraîné sur un ensemble de 10 623 images, est scrutée dans ce chapitre. Pour appréhender sa capacité de détection, nous l'avons mis à l'épreuve sur un ensemble de données reflétant une variété de scénarios typiques du milieu ferroviaire, incluant obstacles potentiels et divers éléments caractéristiques de l'environnement d'un train de fret.

Du côté qualitatif, nous observons la capacité du modèle à produire des cartes de prédiction de chaleur, ou "*heatmaps*". Ces heatmaps visualisent les zones de l'image que le modèle juge pertinentes pour la détection d'anomalies.

En termes quantitatifs, nous avons opté pour la métrique pixel-wise AUROC [193]. Cette métrique, prisée en détection d'anomalies lorsque les résultats se manifestent sous forme de cartographies thermiques, évalue la finesse avec laquelle le modèle détecte les anomalies à l'échelle du pixel.

Cette section dresse un bilan des capacités du modèle hybride pour la détection d'anomalies en environnement ferroviaire, en alliant approches qualitatives et quantitatives. La table 5.2 liste les hyperparamètres du modèle suggéré. Les résultats qualitatifs sont présentés dans la section 5.5.1, tandis que la section 5.5.2 détaille les résultats quantitatifs obtenus.

TABLE 5.2: Résumé des hyperparamètres et configurations

Composant/Paramètre	**Description**
Encodeur :	
Architecture	Transformeur de vision
Taille du Patch	16x16
Dimension de l'espace latent	768
Nombre de têtes d'attention	8
Taille de l'image	384x384
Taille du batch	64
Époques	100
Taux d'apprentissage	2×10^{-4}
Décroissance de poids	1×10^{-5}
Décodeur :	
Nombre de blocs	Cinq blocs
Structure du bloc	1. Cinq couches convolutionnelles transposées 2. Batchnorm2D 3. Activations ReLU 4. Suréchantillonnage 5. Tanh comme fonction d'activation finale
Dataset :	
Dataset d'entraînement	Ferroviaire Rassemblé
Dataset de test	Ferroviaire Rassemblé
Taille du dataset d'entraînement	10 624 images
Taille du dataset de test	1 000 images
Résolution de base	1920x1080
Résolution exploitable	384x384

5.5.1 Résultats qualitatifs

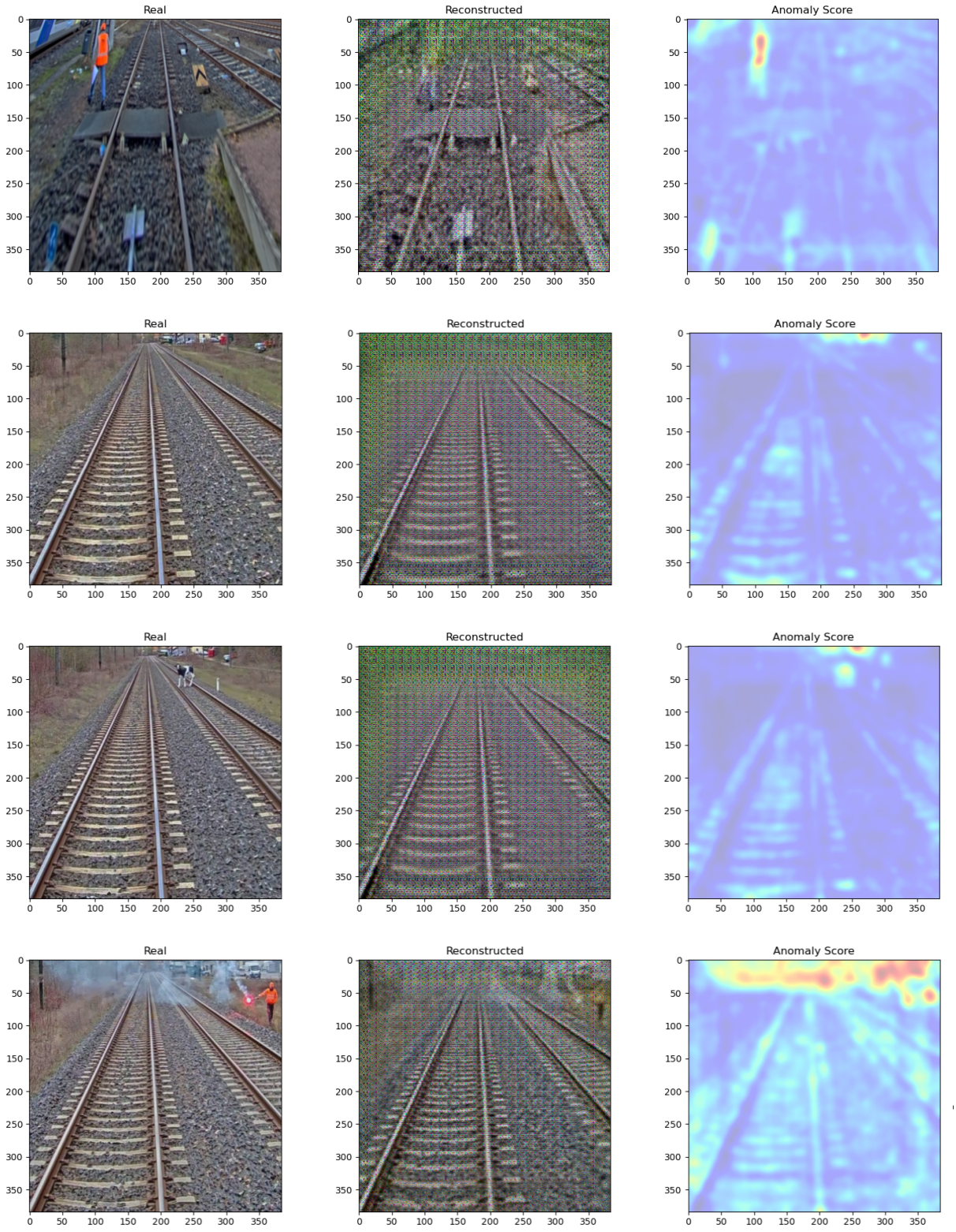

FIGURE 91: Résultats de détection d'anomalies sur le dataset via des heatmaps. Les zones bleues indiquent des erreurs faibles, signifiant probablement aucune anomalie, tandis que les zones rouges signalent des erreurs élevées, pointant des anomalies. Ces couleurs facilitent une analyse visuelle rapide des anomalies

5.5.2 Résultats quantitatifs

L'aire sous la courbe caractéristique de fonctionnement du récepteur notée AUROC (*Area Under the Receiver Operating Characteristic*) [193] est une métrique essentielle dans l'évaluation des performances d'un modèle de détection d'anomalies. Sa mise en œuvre nécessite préalablement la génération de vérités terrains. Dans le contexte de détection d'obstacles ferroviaires, cela se traduit par la création manuelle de masques pour chaque image, illustrant les obstacles présents dans la scène. Une fois ces masques établis, l'AUROC [193] mesure la capacité du modèle à différencier les observations normales des observations anormales. La courbe ROC [193] est construite en traçant les taux de vrais positifs (TPR) et de faux positifs (FPR) pour divers seuils de classification. L'aire sous cette courbe, l'AUROC [193], permet d'évaluer la performance du modèle. Une AUROC [193] de 1.0 indique une distinction parfaite entre les anomalies et les non-anomalies ; une valeur de 0.5 suggère une performance équivalente à un tirage aléatoire ; et une valeur de 0 indique une inversion totale dans la distinction. Cette métrique est d'une importance particulière lorsqu'il y a une faible prévalence d'anomalies, car elle n'est pas biaisée par leur fréquence d'occurrence.

Pour évaluer la performance d'un modèle de détection d'anomalies à l'échelle des pixels avec l'AUROC [193], il est essentiel de comprendre d'abord comment la courbe ROC est déterminée. Cette courbe illustre la capacité d'un modèle à discriminer entre les classes positives et négatives à divers seuils de décision. Elle est tracée en plaçant le TPR en fonction du FPR pour différents seuils sur la heatmap.

Le TPR, également appelé sensibilité, mesure la proportion des valeurs positives réelles qui sont correctement identifiées par le modèle. Il est mathématiquement défini par :

$$TPR = \frac{TP}{TP + FN} \tag{5.16}$$

où TP est le nombre de vrais positifs et FN est le nombre de faux négatifs.

Le FPR évalue la proportion des valeurs négatives réelles que le modèle identifie incorrectement comme positives. Sa définition mathématique est :

$$FPR = \frac{FP}{FP + TN} \tag{5.17}$$

où FP est le nombre de faux positifs et TN est le nombre de vrais négatifs.

Avec ces informations, les valeurs TPR et FPR sont dérivées pour différents seuils, aboutissant à la courbe ROC qui est simplement un graphique de TPR en fonction de FPR. L'aire sous cette courbe, nommée AUROC, est calculée comme :

$$AUROC = \int_0^1 TPR(FPR)\, dFPR \tag{5.18}$$

Ce processus est répété pour chaque paire d'images (masque et heatmap). En fin de compte, l'AUROC moyen est déterminé en calculant la moyenne des AUROC pour toutes les images de test :

$$\text{AUROC moyen} = \frac{1}{N} \sum_{i=1}^{N} AUROC_i \tag{5.19}$$

où N est le nombre total d'images dans l'ensemble de test.

La figure 92 montre un aperçu des masques préparés pour la phase de l'évaluation du modèle hybride. La figure 93 montre la courbe ROC [193] sur l'ensemble du dataset de test ainsi que la figure 94 qui montre l'aire AUROC [193].

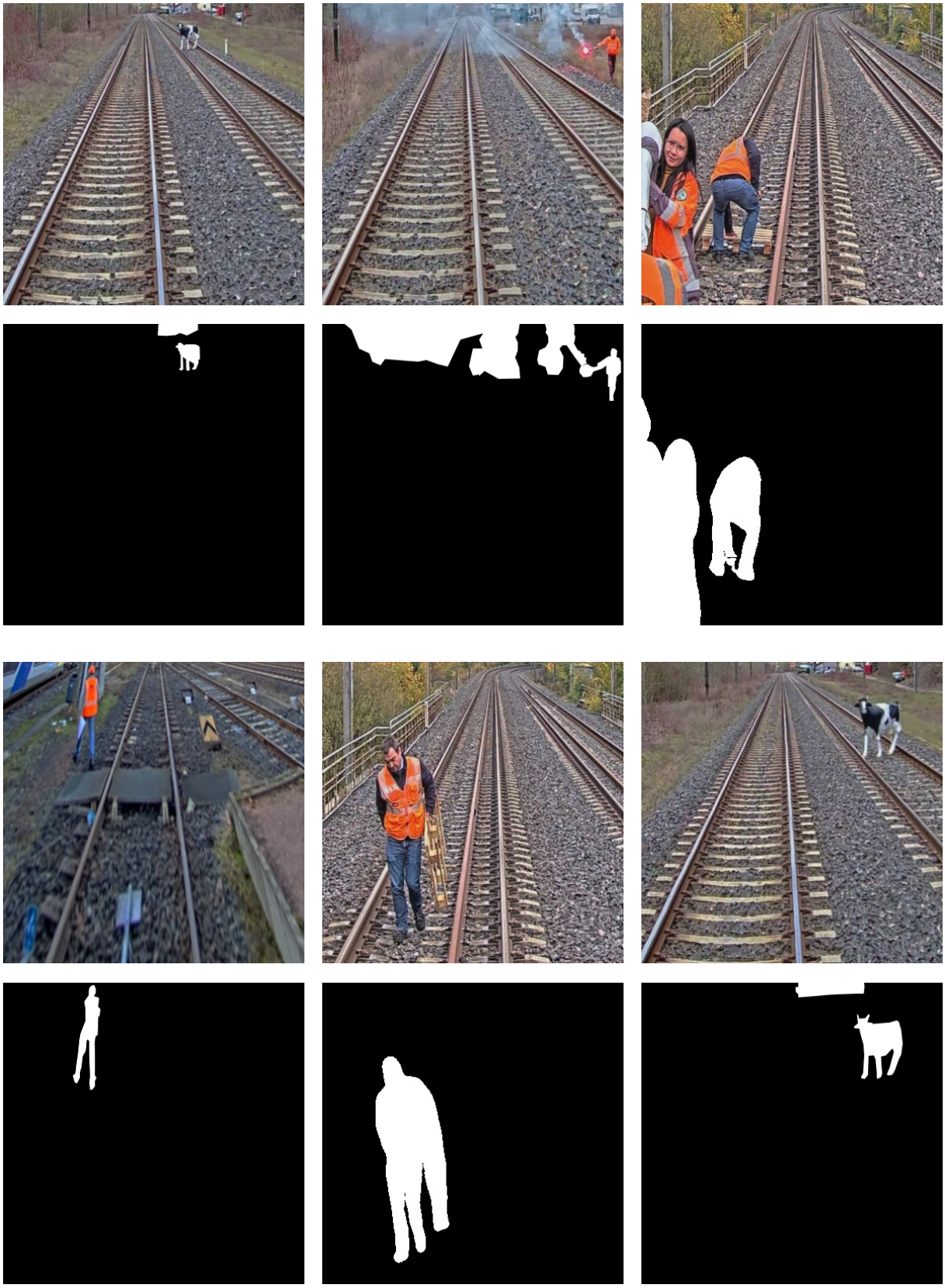

FIGURE 92: Exemples d'images générées représentants des masques d'obstacles ainsi que les images originales

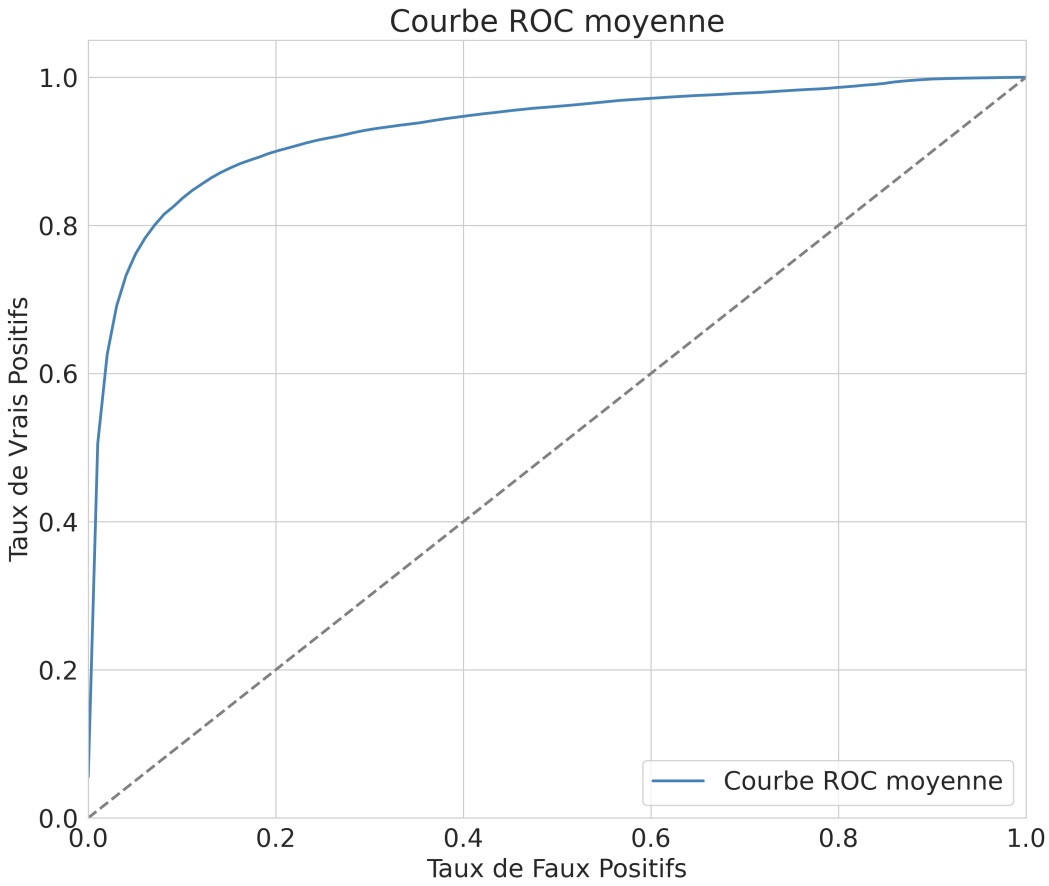

FIGURE 93: La courbe ROC moyenne pour l'ensemble du dataset de test. La courbe illustre la performance du modèle en traçant le taux de vrais positifs (TPR ou sensibilité) en fonction du taux de faux positifs (FPR). La ligne diagonale représente une prédiction aléatoire ; un modèle performant se situe au-dessus de cette ligne, et un modèle non performant en dessous.

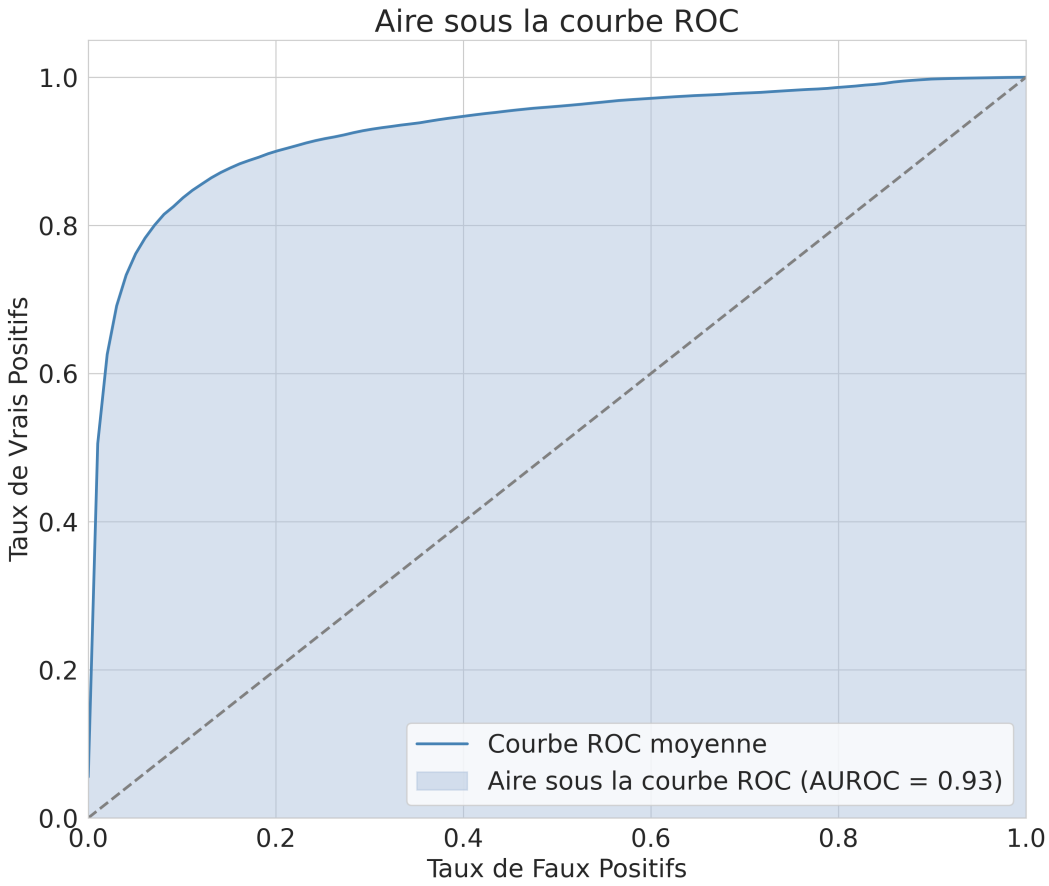

FIGURE 94: La zone sous la courbe ROC (AUROC) pour l'ensemble du dataset de test. L'AUROC représente la probabilité qu'une anomalie détectée (via la heatmap) soit classée au-dessus d'une zone non-anormale (mask). La zone colorée illustre l'aire sous la courbe ROC, qui quantifie la capacité globale du modèle à détecter les anomalies.

De plus, les performances du modèle hybride ViT ont été comparées à celles de trois autres modèles de référence. Le modèle CAE, issu de l'étude exploratoire dans 4 et configuré avec mila_psnr_adam, a obtenu une AUROC de 0.66. Le modèle f-Anogan [175], basé sur les GANs [221], a présenté une AUROC de 0.65. VT-ADL [178], qui intègre les caractéristiques des ViTs [134] et des convolutions aussi, a atteint une AUROC de 0.75. Le modèle hybride ViT s'est distingué en surpassant ces modèles avec une AUROC de 0.93.

La figure 95 montre le comparatif entre les modèles en terme de courbe AUROC. Le tableau 5.3 illustre l'écart relatif en % de chaque modèle par rapport au modèle hybride ViT.

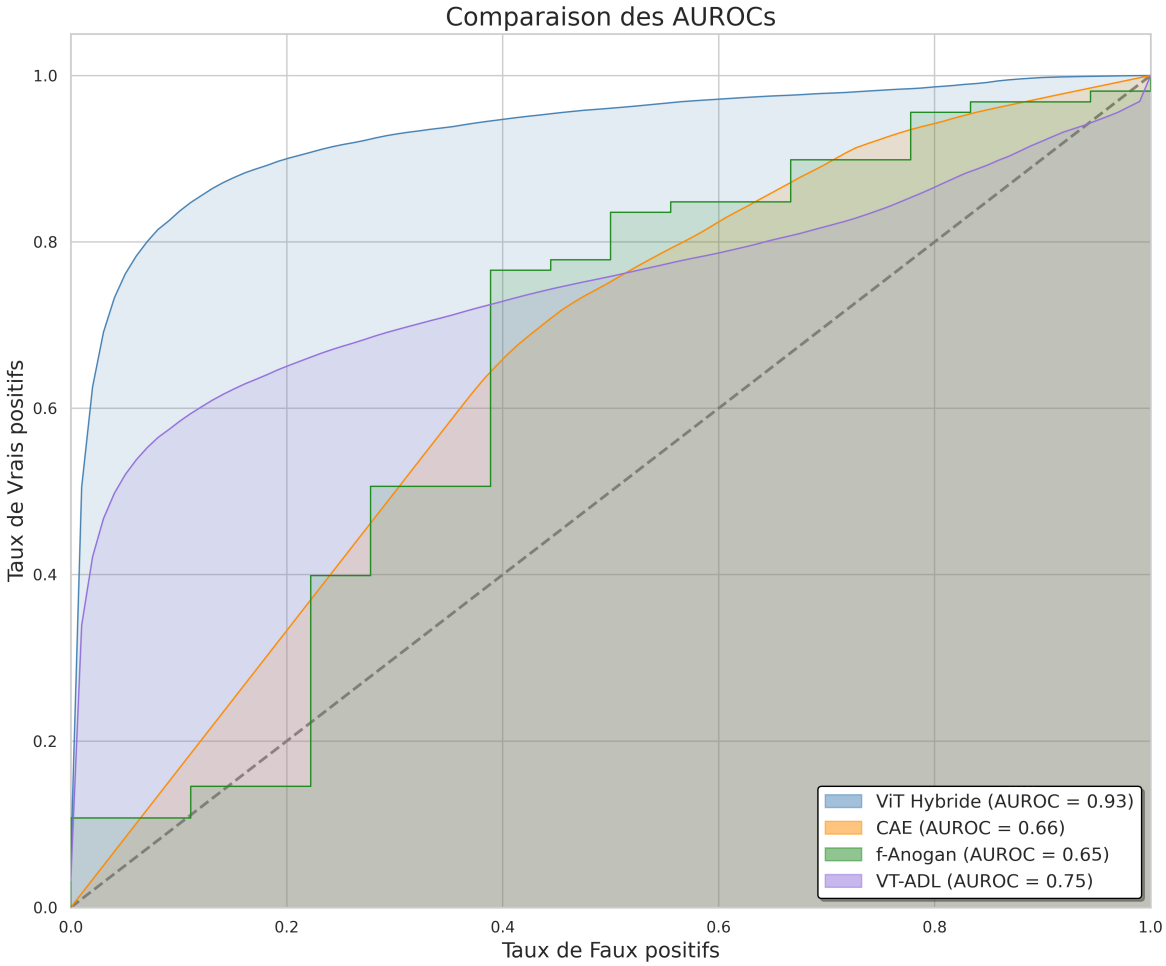

FIGURE 95: Comparaison du modèle proposée avec d'autres modèles.

Modèle	AUROC	Différence (%)
hybride ViT	0.93	-
CAE	0.66	+27%
f-Anogan	0.65	+28%
VT-ADL	0.75	+18%

TABLE 5.3: Comparaison des AUROCs et gain relatif des modèles par rapport au modèle hybride ViT.

En relation avec les détails techniques de l'entraînement, il est à noter que le modèle a été entraîné sur la solution NVIDIA DGX Workstation, équipée de 4 GPUs Tesla V-100. Le temps d'inférence sur la même architecture est de 50ms. Le temps total requis pour l'entraînement a été de 24 heures. Par ailleurs, l'architecture utilisée dans le train de fret autonome est un ADLINK DLAP-8001 AC/M32G doté de 2 GPUs QUADRO RTX4000. En définitive, le modèle proposé hybride présente une valeur AUROC de 0.93, qui est élevée comparée aux autres modèles de comparaisons décrits dans 95 et la table 5.3.

5.6 Conclusion

Ce chapitre se consacre à la détection d'anomalies dans le contexte ferroviaire à l'aide de techniques avancées d'apprentissage profond. Le transformeur de vision (ViT) a été présenté comme un élément clé pour l'analyse des images de scène ferroviaire. Il a été observé que les ViTs sont particulièrement adaptés à la manipulation d'images complexes pour la détection d'anomalies.

En utilisant cette architecture comme base, une architecture hybride pour un autoencodeur a été développée. L'encodeur intègre les transformeurs de vision (ViT) pour extraire des caractéristiques pertinentes des images, tandis que le décodeur fait appel à des couches de convolution pour la phase de reconstruction. Cette combinaison assure une reconstruction efficace de l'image tout en optimisant le nombre de paramètres de l'architecture.

La constitution d'un dataset adéquat s'est avérée cruciale pour cette recherche. Ce dataset a été constitué à partir de mesures effectuées par des capteurs installés sur des équipements ferroviaires, notamment les locomotives. Cette méthodologie garantit la qualité et la fiabilité des données, fournissant ainsi un contexte réel pour l'entraînement et la validation du modèle. En séparant de manière appropriée les données d'entraînement et de test, une évaluation précise du modèle a été réalisée.

Les analyses menées ont confirmé l'efficacité de l'approche hybride. Les évaluations, tant quantitatives que qualitatives, sur un dataset réel, ont souligné la précision de la détection et la pertinence du modèle pour identifier des anomalies spécifiques au contexte ferroviaire.

Pour conclure, l'utilisation d'une architecture hybride combinée à un dataset bien structuré favorise l'émergence de solutions pour améliorer la sécurité et l'efficience des réseaux ferroviaires grâce à une détection automatisée d'anomalies.

Chapitre 6

Conclusion

6.1 Conclusion

La détection de situations anormales, spécialement dans le cadre du projet Train Fret Autonome (TFA), représente un enjeu majeur pour assurer la sécurité des opérations. L'angle d'approche adopté dans cette thèse a été l'utilisation des possibilités offertes par l'apprentissage profond, en se focalisant principalement sur l'apprentissage non-supervisé.

La première contribution de ce travail a été la constitution d'un état de l'art détaillé. Grâce à un survey rigoureux, nous avons dressé un panorama des différentes techniques existantes, qu'elles soient supervisées, non-supervisées, basées sur le traitement d'images et sur divers capteurs. Cette cartographie a révélé que, bien que la détection d'obstacles ait été largement traitée par des méthodes supervisées, un vide est palpable concernant les approches non-supervisées.

Pour la détection d'obstacles sur la voie de circulation principale, une étude exploratoire a été réalisée en se basant sur l'utilisation d'autoencodeurs convolutionnels. Ces derniers ont été entraînés sur des régions d'intérêt des voies de circulation ferroviaires du dataset Railsem. Afin d'adapter ces données à la détection d'anomalies, un prétraitement rigoureux a été réalisé. L'étude s'est appuyée sur des aspects intrinsèques des modèles, tels que les fonctions de pertes, les fonctions d'activation et les optimiseurs. Devant le manque de données représentant des obstacles, un dataset synthétique a été conçu en utilisant le GP-GAN pour incruster des obstacles sur des scènes réelles, offrant ainsi un rendu plus réaliste par rapport aux méthodes d'incrustation traditionnelles. Parallèlement, l'introduction du gap-score comme métrique a eu pour avantage de juger les performances des modèles de manière simplifiée, sans nécessiter la préparation de vérités terrains, tels que des masques, lors de l'utilisation d'incrustations. Face à la diversité des 240 modèles générés dans cette étude exploratoire, l'algorithme d'aide à la décision multi-critère TOPSIS a été utilisé pour distinguer et extraire le modèle le plus performant. Ce modèle optimal a ensuite été testé avec succès dans un scénario d'obstacles réel.

Enfin, la dernière contribution de la thèse réside dans la proposition d'une architecture d'autoencodeur hybride ViT pour le monitoring de l'environnement du train de fret autonome. L'adoption des transformeurs vise principalement à capter les caractéristiques complexes présentes dans des images variées et diversifiées. Le recours à ces mécanismes

est essentiel pour traiter les particularités inhérentes aux images complexes. Cette contribution a été élaborée pour pallier aux lacunes des autoencodeurs convolutionnels. Bien que ces derniers demeurent d'une grande efficacité et se montrent moins gourmands en ressources de calcul pour traiter des images simples, bien définies et peu variables, ils peuvent présenter des limites dans des contextes où les images sont complexes et variées. L'architecture hybride proposée vise donc à offrir une solution adaptée à ces scénarios plus exigeants.

Pour la mise en œuvre de cette architecture, un dataset regroupant plus de 10k d'images a été constitué. Outre les scènes ferroviaires typiques, ce dataset intègre également des images d'obstacles réels visibles dans l'environnement du train de fret autonome. Le dataset d'entraînement cible les caractéristiques intrinsèques des scènes ferroviaires, tandis que le dataset de test évalue la robustesse et l'efficacité de l'architecture dans des situations d'anomalies (obstacles) au niveau de la scène ferroviaire.

Les résultats, qu'ils soient qualitatifs ou quantitatifs, témoignent de la validité de cette approche. Néanmoins, il est reconnu que des optimisations et des ajustements sont envisageables pour perfectionner davantage la solution proposée. Cette contribution s'inscrit dans le cadre des efforts continus pour améliorer la surveillance de l'environnement ferroviaire.

6.2 Travaux futurs et améliorations

Les méthodes introduites dans cette thèse ont déjà montré une grande efficacité dans la détection d'obstacles sur les voies ferrées. Cependant, comme toute recherche, il existe toujours un potentiel d'optimisation et d'extension. En tirant parti des avancées technologiques et en approfondissant les techniques présentées, nous pouvons aller encore plus loin dans l'amélioration de la robustesse et de la précision. Voici donc une série d'améliorations et suggestions pour enrichir et étendre les méthodes établies dans cette thèse :

- **Dataset de référence publique** : La création d'un tel ensemble de données permettrait non seulement de standardiser les benchmarks, mais également d'enrichir la communauté de recherche. Assurer sa variabilité, ses annotations détaillées et sa taille est primordial.
- **Détecteurs avancés** : Exploiter des capteurs comme le *LiDAR* pour une cartographie 3D détaillée, les *caméras stéréo* pour une perception en profondeur, et le *radar à ondes millimétriques* pour la détection dans des conditions météorologiques défavorables.
- **Méthodes d'apprentissage profond non-explorées** : Poursuivre l'exploration d'autres techniques, en particulier des méthodes comme le *few-shot learning* qui représente un compromis efficace lorsqu'il s'agit de l'utilisation des modèles supervisés, car il requiert peu de données d'entraînement, ce qui permet de se concentrer

sur des classes ou des obstacles critiques tels que les piétons, les voitures, etc., ainsi que *online learning* qui permet de s'adapter en temps réel aux données nouvellement acquises, améliorant ainsi la performance des modèles sans nécessiter de grandes quantités de données d'entraînement préalables.

— **Approches multi-modales** : Fusionner les données de différents capteurs pour renforcer la robustesse et la précision de la détection.

Bibliographie

[1] Badrloo, Samira, Masood Varshosaz, Saied Pirasteh, and Jonathan Li. "Image-based obstacle detection methods for the safe navigation of unmanned vehicles : A review." Remote Sensing 14, no. 15 (2022) : 3824.

[2] Ignatious, H. A., & Khan, M. (2022). An overview of sensors in Autonomous Vehicles. Procedia Computer Science, 198, 736-741.

[3] Vatakov, Vasil, Evelina Pencheva, and Emilia Dimitrova. "Recent Advances in Artificial Intelligence for Improving Railway Operations." In 2022 30th National Conference with International Participation (TELECOM), pp. 1-4. IEEE, 2022.

[4] Khan, Abdullah Ayub, Asif Ali Laghari, and Shafique Ahmed Awan. "Machine learning in computer vision : a review." EAI Endorsed Transactions on Scalable Information Systems 8, no. 32 (2021) : e4-e4.

[5] Ji, Albert, Wai Lok Woo, Eugene Wai Leong Wong, and Yang Thee Quek. "Rail track condition monitoring : A review on deep learning approaches." Intell. Robot 1 (2021) : 151-175.

[6] Chouchani, Nadia, Sana Debbech, and Matthieu Perin. "Model-based safety engineering for autonomous train map." Journal of Systems and Software 183 (2022) : 111082.

[7] Dalzochio, Jovani, Rafael Kunst, Edison Pignaton, Alecio Binotto, Srijnan Sanyal, Jose Favilla, and Jorge Barbosa. "Machine learning and reasoning for predictive maintenance in Industry 4.0 : Current status and challenges." Computers in Industry 123 (2020) : 103298.

[8] Omeiza, D., Webb, H., Jirotka, M., & Kunze, L. (2021). Explanations in autonomous driving : A survey. IEEE Transactions on Intelligent Transportation Systems, 23(8), 10142-10162.

[9] Hopkins, Debbie, and Tim Schwanen. "Talking about automated vehicles : What do levels of automation do ?." Technology in Society 64 (2021) : 101488.

[10] Habib, Lydia, Ouazna Oukacha, and Simon Enjalbert. "Towards tramway safety by managing advanced driver assistance systems depending on grades of automation." IFAC-PapersOnLine 54, no. 2 (2021) : 227-232.

[11] Hancock, Peter A., Illah Nourbakhsh, and Jack Stewart. "On the future of transportation in an era of automated and autonomous vehicles." Proceedings of the National Academy of Sciences 116, no. 16 (2019) : 7684-7691.

[12] https://datamyte.com/autonomous-vehicle/

[13] McCarthy, John, Marvin L. Minsky, Nathaniel Rochester, and Claude E. Shannon. "A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955." AI magazine 27, no. 4 (2006) : 12-12.

[14] Jordan, M. I., & Mitchell, T. M. (2015). Machine learning : Trends, perspectives, and prospects. Science, 349(6245), 255-260.

[15] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.

[16] Fan, Deng-Ping, Ge-Peng Ji, Guolei Sun, Ming-Ming Cheng, Jianbing Shen, and Ling Shao. "Camouflaged object detection." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 2777-2787. 2020.

[17] Wen, Li-Hua, and Kang-Hyun Jo. "Deep learning-based perception systems for autonomous driving : A comprehensive survey." Neurocomputing 489 (2022) : 255-270.

[18] Stockman, George, and Linda G. Shapiro. Computer vision. Prentice Hall PTR, 2001.

[19] Shi, Xiupeng, Yiik Diew Wong, Chen Chai, and Michael Zhi-Feng Li. "An automated machine learning (AutoML) method of risk prediction for decision-making of autonomous vehicles." IEEE Transactions on Intelligent Transportation Systems 22, no. 11 (2020) : 7145-7154.

[20] Gerke, Sara, Timo Minssen, and Glenn Cohen. "Ethical and legal challenges of artificial intelligence-driven healthcare." In Artificial intelligence in healthcare, pp. 295-336. Academic Press, 2020.

[21] Bécue, Adrien, Isabel Praça, and João Gama. "Artificial intelligence, cyber-threats and Industry 4.0 : Challenges and opportunities." Artificial Intelligence Review 54, no. 5 (2021) : 3849-3886.

[22] Hayes-Roth, Frederick. "Rule-based systems." Communications of the ACM 28, no. 9 (1985) : 921-932.

[23] https://academic-accelerator.com/encyclopedia/civic-intelligence

[24] Shrestha, Prajwal. "Application of Machine Learning and Deep Learning Techniques for Nepal Stock Market Price Prediction." (2021).

[25] https://augnitive.com/introduction-to-machine-learning/

[26] https://levity.ai/blog/neural-networks-cnn-ann-rnn

[27] Qian, Tracy, Jackson Kaunismaa, and Tony Chung. "Cmelgan : An efficient conditional generative model based on mel spectrograms." arXiv preprint arXiv :2205.07319 (2022).

[28] Mosser, Lukas, Olivier Dubrule, and Martin J. Blunt. "Stochastic reconstruction of an oolitic limestone by generative adversarial networks." Transport in Porous Media 125, no. 1 (2018) : 81-103.

[29] Liao, Shu-Hsien. "Expert system methodologies and applications a decade review from 1995 to 2004." Expert systems with applications 28, no. 1 (2005) : 93-103.

[30] El Naqa, Issam, and Martin J. Murphy. What is machine learning ?. Springer International Publishing, 2015.

[31] Kapoor, Amita, Antonio Gulli, Sujit Pal, and Francois Chollet. Deep Learning with TensorFlow and Keras : Build and deploy supervised, unsupervised, deep, and reinforcement learning models. Packt Publishing Ltd, 2022.

[32] Ericsson, Linus, Henry Gouk, Chen Change Loy, and Timothy M. Hospedales. "Self-supervised representation learning : Introduction, advances, and challenges." IEEE Signal Processing Magazine 39, no. 3 (2022) : 42-62.

[33] Zhai, Xiaohua, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. "S4l : Self-supervised semi-supervised learning." In Proceedings of the IEEE/CVF international conference on computer vision, pp. 1476-1485. 2019.

[34] Hopfield, John J. "Artificial neural networks." IEEE Circuits and Devices Magazine 4, no. 5 (1988) : 3-10.

[35] Janiesch, Christian, Patrick Zschech, and Kai Heinrich. "Machine learning and deep learning." Electronic Markets 31, no. 3 (2021) : 685-695.

[36] Hinton, Geoffrey E. "How neural networks learn from experience." Scientific American 267, no. 3 (1992) : 144-151.

[37] LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86, no. 11 (1998) : 2278-2324.

[38] Akram, Saba, and Quarrat Ul Ann. "Newton raphson method." International Journal of Scientific & Engineering Research 6, no. 7 (2015) : 1748-1752.

[39] Ruder, Sebastian. "An overview of gradient descent optimization algorithms." arXiv preprint arXiv :1609.04747 (2016).

[40] Sharma, Sagar, Simone Sharma, and Anidhya Athaiya. "Activation functions in neural networks." Towards Data Sci 6, no. 12 (2017) : 310-316.

[41] Medsker, Larry R., and L. C. Jain. "Recurrent neural networks." Design and Applications 5, no. 64-67 (2001) : 2.

[42] Pang, Yanwei, Manli Sun, Xiaoheng Jiang, and Xuelong Li. "Convolution in convolution for network in network." IEEE transactions on neural networks and learning systems 29, no. 5 (2017) : 1587-1597.

[43] Sze, Vivienne, Yu-Hsin Chen, Tien-Ju Yang, and Joel S. Emer. "Efficient processing of deep neural networks : A tutorial and survey." Proceedings of the IEEE 105, no. 12 (2017) : 2295-2329.

[44] Gholamalinezhad, Hossein, and Hossein Khosravi. "Pooling methods in deep neural networks, a review." arXiv preprint arXiv :2009.07485 (2020).

[45] Dumoulin, Vincent, and Francesco Visin. "A guide to convolution arithmetic for deep learning." arXiv preprint arXiv :1603.07285 (2016).

[46] http://yann.lecun.com/exdb/mnist/

[47] https://www.cs.toronto.edu/ kriz/cifar.html

[48] Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet : A large-scale hierarchical image database." In 2009 IEEE conference on computer vision and pattern recognition, pp. 248-255. Ieee, 2009.

[49] Parkhi, Omkar, Andrea Vedaldi, and Andrew Zisserman. "Deep face recognition." In BMVC 2015-Proceedings of the British Machine Vision Conference 2015. British Machine Vision Association, 2015.

[50] Zhou, Bolei, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. "Scene parsing through ade20k dataset." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 633-641. 2017.

[51] LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86, no. 11 (1998) : 2278-2324.

[52] Bridle, John. "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters." Advances in neural information processing systems 2 (1989).

[53] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25.

[54] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv :1409.1556 (2014).

[55] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2015.

[56] Sanjay Singh, Anurag Kishore, Natural Language Image Descriptor, Conference : Third IEEE International Conference on Recent Advances in Intelligent Computational System (RAICS 2015)At : Trivandrum, India.

[57] Carion, Nicolas, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. "End-to-end object detection with transformers." In European conference on computer vision, pp. 213-229. Cham : Springer International Publishing, 2020.

[58] Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali 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.

[59] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net : Convolutional networks for biomedical image segmentation." In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015 : 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18, pp. 234-241. Springer International Publishing, 2015.

[60] Iandola, Forrest N., Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. "SqueezeNet : AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size." arXiv preprint arXiv :1602.07360 (2016).

[61] Chandola, Varun, Arindam Banerjee, and Vipin Kumar. "Anomaly detection : A survey." ACM computing surveys (CSUR) 41, no. 3 (2009) : 1-58.

[62] Carmona, Chris U., François-Xavier Aubet, Valentin Flunkert, and Jan Gasthaus. "Neural contextual anomaly detection for time series." arXiv preprint arXiv :2107.07702 (2021).

[63] Jiang, Fan, Ying Wu, and Aggelos K. Katsaggelos. "Detecting contextual anomalies of crowd motion in surveillance video." In 2009 16th IEEE International Conference on Image Processing (ICIP), pp. 1117-1120. IEEE, 2009.

[64] Goldberger, Ary L., Luis AN Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. "PhysioBank, PhysioToolkit, and PhysioNet : components of a new research resource for complex physiologic signals." circulation 101, no. 23 (2000) : e215-e220.

[65] Görnitz, Nico, Marius Kloft, Konrad Rieck, and Ulf Brefeld. "Toward supervised anomaly detection." Journal of Artificial Intelligence Research 46 (2013) : 235-262.

[66] Kiran, B. Ravi, Dilip Mathew Thomas, and Ranjith Parakkal. "An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos." Journal of Imaging 4, no. 2 (2018) : 36.

[67] Bank, Dor, Noam Koenigstein, and Raja Giryes. "Autoencoders." arXiv preprint arXiv :2003.05991 (2020).

[68] Liu, Fei Tony, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation forest." In 2008 eighth ieee international conference on data mining, pp. 413-422. IEEE, 2008.

[69] Alghushairy, Omar, Raed Alsini, Terence Soule, and Xiaogang Ma. "A review of local outlier factor algorithms for outlier detection in big data streams." Big Data and Cognitive Computing 5, no. 1 (2020) : 1.

[70] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9, no. 8 (1997) : 1735-1780.

[71] Ibrahim, Mariam, Ahmad Alsheikh, Feras M. Awaysheh, and Mohammad Dahman Alshehri. "Machine learning schemes for anomaly detection in solar power plants." Energies 15, no. 3 (2022) : 1082.

[72] Di Mattia, Federico, Paolo Galeone, Michele De Simoni, and Emanuele Ghelfi. "A survey on gans for anomaly detection." arXiv preprint arXiv :1906.11632 (2019).

[73] Dang, Taurus T., Henry YT Ngan, and Wei Liu. "Distance-based k-nearest neighbors outlier detection method in large-scale traffic data." In 2015 IEEE International Conference on Digital Signal Processing (DSP), pp. 507-510. IEEE, 2015.

[74] Ester, Martin, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. "A density-based algorithm for discovering clusters in large spatial databases with noise." In kdd, vol. 96, no. 34, pp. 226-231. 1996.

[75] Aganagic, Mina, and Andrei Okounkov. "Elliptic stable envelopes." arXiv preprint arXiv :1604.00423 (2016).

[76] Pol, Adrian Alan, Victor Berger, Cecile Germain, Gianluca Cerminara, and Maurizio Pierini. "Anomaly detection with conditional variational autoencoders." In 2019 18th IEEE international conference on machine learning and applications (ICMLA), pp. 1651-1657. IEEE, 2019.

[77] Cansado, Antonio, and Alvaro Soto. "Unsupervised anomaly detection in large databases using Bayesian networks." Applied Artificial Intelligence 22, no. 4 (2008) : 309-330.

[78] Lukovics, Miklós, Bence Zuti, Erik Fisher, and Béla Kézi. "Autonomous cars and responsible innovation." (2021).

[79] Lopez, Marc Moreno, and Jugal Kalita. "Deep Learning applied to NLP." arXiv preprint arXiv :1703.03091 (2017).

[80] Matsuo, Y., LeCun, Y., Sahani, M., Precup, D., Silver, D., Sugiyama, M., Uchibe, E. and Morimoto, J., 2022. Deep learning, reinforcement learning, and world models. Neural Networks, 152, pp.267-275.

[81] Pang, Guansong, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. "Deep learning for anomaly detection : A review." ACM computing surveys (CSUR) 54, no. 2 (2021) : 1-38.

[82] injuryfacts.nsc.org

[83] www.bts.gov

[84] Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., Arshad, H. (2018). State-of-the-art in artificial neural network applications : A survey. Heliyon, 4(11), e00938.

[85] Schneble CA, Raymond J, Loder RT. The Demographics of Non-motor Vehicle Associated Railway Injuries Seen at Trauma Centers in the United States 2007 - 2014. Cureus. 2019 Oct 23 ;11(10) :e5974. doi : 10.7759/cureus.5974. PMID : 31803556 ; PMCID : PMC6874290

[86] Lobb, B. (2006). Trespassing on the tracks : A review of railway pedestrian safety research. Journal of Safety Research, 37(4), 359-365.

[87] Pamuła, Teresa, and Wiesław Pamuła. "Detection of safe passage for trains at rail level crossings using deep learning." Sensors 21, no. 18 (2021) : 6281.

[88] Amaral, Vítor, Francisco Marques, André Lourenço, José Barata, and Pedro Santana. "Laser-based obstacle detection at railway level crossings." Journal of Sensors 2016 (2016).

[89] Khoudour, Louahdi, Mohamed Ghazel, Fouzia Boukour, Marc Heddebaut, and El-Miloudi El-Koursi. "Towards safer level crossings : existing recommendations, new applicable technologies and a proposed simulation model." European transport research review 1, no. 1 (2009) : 35-45.

[90] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., ... & Beijbom, O. (2020). nuScenes : A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 11621-11631).

[91] McKinsey & Company. (2016). Ten ways autonomous driving could redefine the automotive world.

[92] Montero, J. and Finger, M., 2020. Railway regulation : A comparative analysis of a diverging reality. In Handbook on railway regulation (pp. 1-20). Edward Elgar Publishing.

[93] Sun, Pei, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo et al. "Scalability in perception for autonomous driving : Waymo open dataset." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 2446-2454. 2020.

[94] Huang, Xinyu, Peng Wang, Xinjing Cheng, Dingfu Zhou, Qichuan Geng, and Ruigang Yang. "The apolloscape open dataset for autonomous driving and its application." IEEE transactions on pattern analysis and machine intelligence 42, no. 10 (2019) : 2702-2719.

[95] Ristić-Durrant, Danijela, Marten Franke, and Kai Michels. "A review of vision-based on-board obstacle detection and distance estimation in railways." Sensors 21, no. 10 (2021) : 3452.

[96] Gakkhar, Shobhit, and Bhupendra Panchal. "A review on accident prevention methods at railway line crossings." Int. Res. J. Eng. Technol. 5 (2018) : 1102-1107.

[97] Oh, Kyuetaek, Mintaek Yoo, Nayoung Jin, Jisu Ko, Jeonguk Seo, Hyojin Joo, and Minsam Ko. "A Review of Deep Learning Applications for Railway Safety." Applied Sciences 12, no. 20 (2022) : 10572.

[98] Liu, Scarlett, Quandong Wang, and Yiping Luo. "A review of applications of visual inspection technology based on image processing in the railway industry." Transportation Safety and Environment 1, no. 3 (2019) : 185-204.

[99] Yu, Xiaoyan, and Marin Marinov. "A study on recent developments and issues with obstacle detection systems for automated vehicles." Sustainability 12, no. 8 (2020) : 3281.

[100] Pappaterra, Mauro José, Francesco Flammini, Valeria Vittorini, and Nikola Bešinović. "A systematic review of artificial intelligence public datasets for railway applications." Infrastructures 6, no. 10 (2021) : 136.

[101] Pappaterra, Mauro José. "A Review of Literature and Public Datasets for the Application of Artificial Intelligence in the Railway Industry." (2022).

[102] Rosić, Slobodan, Dušan Stamenković, Milan Banić, Miloš Simonović, Danijela Ristić-Durrant, and Cristian Ulianov. "Analysis of the Safety Level of Obstacle Detection in Autonomous Railway Vehicles." Acta Polytech. Hung 1 (2022) : 187-205.

[103] Bešinović, Nikola, Lorenzo De Donato, Francesco Flammini, Rob MP Goverde, Zhiyuan Lin, Ronghui Liu, Stefano Marrone, Roberto Nardone, Tianli Tang, and Valeria Vittorini. "Artificial Intelligence in Railway Transport : Taxonomy, Regulations, and Applications." IEEE Transactions on Intelligent Transportation Systems 23, no. 9 (2021) : 14011-14024.

[104] Schmidhuber, Jürgen. "Deep learning in neural networks : An overview." Neural networks 61 (2015) : 85-117.

[105] Dike, Happiness Ugochi, Yimin Zhou, Kranthi Kumar Deveerasetty, and Qingtian Wu. "Unsupervised learning based on artificial neural network : A review." In 2018 IEEE International Conference on Cyborg and Bionic Systems (CBS), pp. 322-327. IEEE, 2018.

[106] Kafetzis, Dimitrios, Ioannis Fourfouris, Savvas Argyropoulos, and Iordanis Koutsopoulos. "UAV-assisted aerial survey of railways using deep learning." In 2020 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 1491-1500. IEEE, 2020.

[107] Girshick, Ross. "Fast r-cnn." In Proceedings of the IEEE international conference on computer vision, pp. 1440-1448. 2015.

[108] Henderson, Paul, and Vittorio Ferrari. "End-to-end training of object class detectors for mean average precision." In Computer Vision–ACCV 2016 : 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part V 13, pp. 198-213. Springer International Publishing, 2017.

[109] Ye, Tao, Zhihao Zhang, Xi Zhang, and Fuqiang Zhou. "Autonomous railway traffic object detection using feature-enhanced single-shot detector." IEEE Access 8 (2020) : 145182-145193.

[110] Kong, T., Sun, F., Yao, A., Liu, H., Lu, M., & Chen, Y. (2017). Ron : Reverse connection with objectness prior networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 5936-5944).

[111] Li, Juan, Fuqiang Zhou, and Tao Ye. "Real-world railway traffic detection based on faster better network." IEEE Access 6 (2018) : 68730-68739.

[112] Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "SSD : Single shot multi-box detector." In Computer Vision–ECCV 2016 : 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14, pp. 21-37. Springer International Publishing, 2016.

[113] Ding, Xiaohan, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. "Repvgg : Making vgg-style convnets great again." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 13733-13742. 2021.

[114] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

[115] Ou, Xianfeng, Pengcheng Yan, Yiming Zhang, Bing Tu, Guoyun Zhang, Jianhui Wu, and Wujing Li. "Moving object detection method via ResNet-18 with encoder–decoder structure in complex scenes." IEEE Access 7 (2019) : 108152-108160.

[116] Wang, Hui, Fan Zhang, and Li Wang. "Fruit classification model based on improved Darknet53 convolutional neural network." In 2020 International Conference on Intelligent Transportation, Big Data and Smart City (ICITBS), pp. 881-884. IEEE, 2020.

[117] Pan, Haixia, Yanan Li, Hongqiang Wang, and Xiaomeng Tian. "Railway Obstacle Intrusion Detection Based on Convolution Neural Network Multitask Learning." Electronics 11, no. 17 (2022) : 2697.

[118] Rampriya, R. S., R. Suganya, Sabari Nathan, and P. Shunmuga Perumal. "A comparative assessment of deep neural network models for detecting obstacles in the real time aerial railway track images." Applied Artificial Intelligence 36, no. 1 (2022) : 2018184.

[119] Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. "Mobilenets : Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv :1704.04861 (2017).

[120] Yu, Mingyang, Peng Yang, and Sen Wei. "Railway obstacle detection algorithm using neural network." In AIP Conference Proceedings, vol. 1967, no. 1, p. 040017. AIP Publishing LLC, 2018.

[121] Albawi, Saad, Tareq Abed Mohammed, and Saad Al-Zawi. "Understanding of a convolutional neural network." In 2017 international conference on engineering and technology (ICET), pp. 1-6. Ieee, 2017.

[122] He, Deqiang, Zhiheng Zou, Yanjun Chen, Bin Liu, Xiaoyang Yao, and Sheng Shan. "Obstacle detection of rail transit based on deep learning." Measurement 176 (2021) : 109241.

[123] Redmon, Joseph, and Ali Farhadi. "YOLO9000 : better, faster, stronger." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7263-7271. 2017.

[124] Redmon, Joseph, and Ali Farhadi. "Yolov3 : An incremental improvement." arXiv preprint arXiv :1804.02767 (2018).

[125] Xu, Zhi-Feng, Rui-Sheng Jia, Yan-Bo Liu, Chao-Yue Zhao, and Hong-Mei Sun. "Fast method of detecting tomatoes in a complex scene for picking robots." IEEE Access 8 (2020) : 55289-55299.

[126] Bochkovskiy, A., Wang, C. Y., and Liao, H. Y. M. (2020). Yolov4 : Optimal speed and accuracy of object detection. arXiv preprint arXiv :2004.10934.

[127] Wu, Wentong, Han Liu, Lingling Li, Yilin Long, Xiaodong Wang, Zhuohua Wang, Jinglun Li, and Yi Chang. "Application of local fully Convolutional Neural Network combined with YOLO v5 algorithm in small target detection of remote sensing image." PloS one 16, no. 10 (2021) : e0259283.

[128] Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft coco : Common objects in context." In Computer Vision–ECCV 2014 : 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, pp. 740-755. Springer International Publishing, 2014.

[129] Geiger, Andreas, Philip Lenz, Christoph Stiller, and Raquel Urtasun. "Vision meets robotics : The kitti dataset." The International Journal

[130] Vicente, Sara, Joao Carreira, Lourdes Agapito, and Jorge Batista. "Reconstructing pascal voc." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 41-48. 2014.

[131] Li, S., Zhao, H., and Ma, J. (2021). An edge computing-enabled train obstacle detection method based on YOLOv3. Wireless Communications and Mobile Computing, 2021, 1-9.

[132] Chen, Weixun, Siming Meng, and Yuelong Jiang. "Foreign object detection in railway images based on an efficient two-stage convolutional neural network." Computational Intelligence and Neuroscience 2022 (2022).

[133] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2 : Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510-4520).

[134] Khan, Salman, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. "Transformers in vision : A survey." ACM computing surveys (CSUR) 54, no. 10s (2022) : 1-41.

[135] Deng, Yubin, Ping Luo, Chen Change Loy, and Xiaoou Tang. "Pedestrian attribute recognition at far distance." In Proceedings of the 22nd ACM international conference on Multimedia, pp. 789-792. 2014.

[136] Mahtani, Ankur, Wael Ben-Messaoud, Abdelmalik Taleb-Ahmed, Smail Niar, and Clément Strauss. "Pedestrian detection and classification for the autonomous train." In 2020 IEEE 4th International Conference on Image Processing, Applications, and Systems (IPAS), pp. 52-57. IEEE, 2020.

[137] Noble, William S. "What is a support vector machine ?." Nature biotechnology 24, no. 12 (2006) : 1565-1567.

[138] Goldstein, Markus, and Seiichi Uchida. "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data." PloS one 11, no. 4 (2016) : e0152173.

[139] Goldstein, Markus, and Seiichi Uchida. "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data." PloS one 11, no. 4 (2016) : e0152173.

[140] Boussik, Amine, Wael Ben-Messaoud, Smail Niar, and Abdelmalik Taleb-Ahmed. "Railway obstacle detection using unsupervised learning : An exploratory study." In 2021 IEEE Intelligent Vehicles Symposium (IV), pp. 660-667. IEEE, 2021.

[141] Amine, Boussik, Antoine Plissonneau, Wael Ben Messaoud, Abdelmalik Taleb-Ahmed, Smail Niar, Abdelghani Bekrar, and Damien Trentesaux. "Vision-based railway track extraction and obstacle detection using deep learning for autonomous train." In The 2nd International Workshop on Artificial Intelligence for RAILwayS (AI4RAILS)., p. 190. 2021.

[142] Gasparini, Riccardo, Stefano Pini, Guido Borghi, Giuseppe Scaglione, Simone Calderara, Eugenio Fedeli, and Rita Cucchiara. "Anomaly detection for vision-based railway inspection." In Dependable Computing-EDCC 2020 Workshops : AI4RAILS, DREAMS, DSOGRI, SERENE 2020, Munich, Germany, September 7, 2020, Proceedings 16, pp. 56-67. Springer International Publishing, 2020.

[143] www.flir.it

[144] www.stereolabs.com

[145] www.baslerweb.com

[146] Gasparini, Riccardo, Andrea D'Eusanio, Guido Borghi, Stefano Pini, Giuseppe Scaglione, Simone Calderara, Eugenio Fedeli, and Rita Cucchiara. "Anomaly detection, localization and classification for railway inspection." In 2020 25th International Conference on Pattern Recognition (ICPR), pp. 3419-3426. IEEE, 2021.

[147] Chow, Jun Kang, Zhaoyu Su, Jimmy Wu, Pin Siang Tan, Xin Mao, and Yu-Hsing Wang. "Anomaly detection of defects on concrete structures with the convolutional autoencoder." Advanced Engineering Informatics 45 (2020) : 101105.

[148] Opricovic, Serafim, and Gwo-Hshiung Tzeng. "Compromise solution by MCDM methods : A comparative analysis of VIKOR and TOPSIS." European journal of operational research 156, no. 2 (2004) : 445-455.

[149] Wu, Huikai, Shuai Zheng, Junge Zhang, and Kaiqi Huang. "Gp-gan : Towards realistic high-resolution image blending." In Proceedings of the 27th ACM international conference on multimedia, pp. 2487-2495. 2019.

[150] Wang, Yao, Zujun Yu, and Liqiang Zhu. "Intrusion detection for high-speed railways based on unsupervised anomaly detection models." Applied Intelligence (2022) : 1-14.

[151] Brucker, M., Cramariuc, A., von Einem, C., Siegwart, R., & Cadena, C. (2023). Local and Global Information in Obstacle Detection on Railway Tracks. arXiv preprint arXiv :2307.15478.

[152] An, Jinwon, and Sungzoon Cho. "Variational autoencoder based anomaly detection using reconstruction probability." Special lecture on IE 2, no. 1 (2015) : 1-18.

[153] Tjandra, Andros, Berrak Sisman, Mingyang Zhang, Sakriani Sakti, Haizhou Li, and Satoshi Nakamura. "VQVAE unsupervised unit discovery and multi-scale code2spec inverter for zerospeech challenge 2019." arXiv preprint arXiv :1905.11449 (2019).

[154] Salimans, Tim, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. "Pixelcnn++ : Improving the pixelcnn with discretized logistic mixture likelihood and other modifications." arXiv preprint arXiv :1701.05517 (2017).

[155] Tian R, Shi H, Guo B, Zhu L (2022) Multi-scale object detection for high-speed railway clearance intrusion. Appl Intell 52(4) :3511–3526

[156] Wang, Tiange, Zijun Zhang, and Kwok-Leung Tsui. "A Deep Generative Approach for Rail Foreign Object Detections via Semisupervised Learning." IEEE Transactions on Industrial Informatics 19, no. 1 (2022) : 459-468.

[157] Pinaya, Walter HL, Mark S. Graham, Robert Gray, Pedro F. Da Costa, Petru-Daniel Tudosiu, Paul Wright, Yee H. Mah et al. "Fast unsupervised brain anomaly

detection and segmentation with diffusion models." In International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 705-714. Cham : Springer Nature Switzerland, 2022.

[158] Roth, Karsten, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. "Towards total recall in industrial anomaly detection." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14318-14328. 2022.

[159] Liu, Zhikang, Yiming Zhou, Yuansheng Xu, and Zilei Wang. "Simplenet : A simple network for image anomaly detection and localization." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 20402-20411. 2023.

[160] Zavrtanik, V., Kristan, M., & Skočaj, D. (2022, October). Dsr–a dual subspace re-projection network for surface anomaly detection. In European conference on computer vision (pp. 539-554). Cham : Springer Nature Switzerland.

[161] Deng, Hanqiu, Zhaoxiang Zhang, Jinan Bao, and Xingyu Li. "AnoVL : Adapting Vision-Language Models for Unified Zero-shot Anomaly Localization." arXiv preprint arXiv :2308.15939 (2023).

[162] Lei, Jiarui, Xiaobo Hu, Yue Wang, and Dong Liu. "PyramidFlow : High-Resolution Defect Contrastive Localization using Pyramid Normalizing Flow." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14143-14152. 2023.

[163] Zhang, Dongkai, Shibin Gao, Long Yu, Gaoqiang Kang, Xiaoguang Wei, and Dong Zhan. "DefGAN : Defect detection GANs with latent space pitting for high-speed railway insulator." IEEE Transactions on Instrumentation and Measurement 70 (2020) : 1-10.

[164] Çatal, Ozan, Sam Leroux, Cedric De Boom, Tim Verbelen, and Bart Dhoedt. "Anomaly detection for autonomous guided vehicles using bayesian surprise." In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 8148-8153. IEEE, 2020.

[165] Betechuoh, Brain Leke, Tshilidzi Marwala, and Thando Tettey. "Autoencoder networks for HIV classification." Current Science (00113891) 91, no. 11 (2006).

[166] Buckland, Michael, and Fredric Gey. "The relationship between recall and precision." Journal of the American society for information science 45, no. 1 (1994) : 12-19.

[167] Song, Weiwei, Shutao Li, Leyuan Fang, and Ting Lu. "Hyperspectral image classification with deep feature fusion network." IEEE Transactions on Geoscience and Remote Sensing 56, no. 6 (2018) : 3173-3184.

[168] Cai, Zhaowei, and Nuno Vasconcelos. "Cascade r-cnn : Delving into high quality object detection." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 6154-6162. 2018.

[169] Zhang, Shifeng, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z. Li. "Single-shot refinement neural network for object detection." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4203-4212. 2018.

[170] Ma, N., Zhang, X., Zheng, H.T. and Sun, J., 2018. Shufflenet v2 : Practical guidelines for efficient cnn architecture design. In Proceedings of the European conference on computer vision (ECCV) (pp. 116-131).

[171] Nie, Y., Hou, J., Han, X. and Nießner, M., 2021. Rfd-net : Point scene understanding by semantic instance reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 4608-4618).

[172] Chen, Yunqiang, Xiang Sean Zhou, and Thomas S. Huang. "One-class SVM for learning in image retrieval." In Proceedings 2001 international conference on image processing (Cat. No. 01CH37205), vol. 1, pp. 34-37. IEEE, 2001.

[173] Liu, F.T., Ting, K.M. and Zhou, Z.H., 2008, December. Isolation forest. In 2008 eighth ieee international conference on data mining (pp. 413-422). IEEE.

[174] Schlegl, Thomas, Philipp Seeböck, Sebastian M. Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery." In Information Processing in Medical Imaging : 25th International Conference, IPMI 2017, Boone, NC, USA, June 25-30, 2017, Proceedings, pp. 146-157. Cham : Springer International Publishing, 2017.

[175] Schlegl, Thomas, Philipp Seeböck, Sebastian M. Waldstein, Georg Langs, and Ursula Schmidt-Erfurth. "f-AnoGAN : Fast unsupervised anomaly detection with generative adversarial networks." Medical image analysis 54 (2019) : 30-44.

[176] Yurtkulu, Salih Can, Yusuf Hüseyin Şahin, and Gozde Unal. "Semantic segmentation with extended DeepLabv3 architecture." In 2019 27th Signal Processing and Communications Applications Conference (SIU), pp. 1-4. IEEE, 2019.

[177] Yang, Ziyi, Teng Zhang, Iman Soltani Bozchalooi, and Eric Darve. "Memory-augmented generative adversarial networks for anomaly detection." IEEE Transactions on Neural Networks and Learning Systems 33, no. 6 (2021) : 2324-2334.

[178] Mishra, Pankaj, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. "VT-ADL : A vision transformer network for image anomaly detection and localization." In 2021 IEEE 30th International Symposium on Industrial Electronics (ISIE), pp. 01-06. IEEE, 2021.

[179] Lee, Yunseung, and Pilsung Kang. "AnoViT : Unsupervised anomaly detection and localization with vision transformer-based encoder-decoder." IEEE Access 10 (2022) : 46717-46724.

[180] Prabhakar, Chinmay, Hongwei Bran Li, Jiancheng Yang, Suprosana Shit, Benedikt Wiestler, and Bjoern Menze. "ViT-AE++ : Improving Vision Transformer Autoencoder for Self-supervised Medical Image Representations." arXiv preprint arXiv :2301.07382 (2023).

[181] Mathian, Emilie, Huidong Liu, Lynnette Fernandez-Cuesta, Dimitris Samaras, Matthieu Foll, and Liming Chen. "HaloAE : An HaloNet based Local Transformer Auto-Encoder for Anomaly Detection and Localization." arXiv preprint arXiv :2208.03486 (2022).

[182] Zhang, Zhiwen, Teng Li, Xuebin Tang, Xiang Hu, and Yuanxi Peng. "CAEVT : Convolutional Autoencoder Meets Lightweight Vision Transformer for Hyperspectral Image Classification." Sensors 22, no. 10 (2022) : 3902.

[183] Murtagh, F., & Contreras, P. (2012). Algorithms for hierarchical clustering : an overview. Wiley Interdisciplinary Reviews : Data Mining and Knowledge Discovery, 2(1), 86-97.

[184] Amari, S. I. (1993). Backpropagation and stochastic gradient descent method. Neurocomputing, 5(4-5), 185-196.

[185] Zou, Fangyu, Li Shen, Zequn Jie, Weizhong Zhang, and Wei Liu. "A sufficient condition for convergences of adam and rmsprop." In Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, pp. 11127-11135. 2019.

[186] Qian, Ning. "On the momentum term in gradient descent learning algorithms." Neural networks 12, no. 1 (1999) : 145-151.

[187] Koonce, Brett, and Brett Koonce. "ResNet 50." Convolutional Neural Networks with Swift for Tensorflow : Image Recognition and Dataset Categorization (2021) : 63-72.

[188] Wu, Z., Shen, C., & Van Den Hengel, A. (2019). Wider or deeper : Revisiting the resnet model for visual recognition. Pattern Recognition, 90, 119-133.

[189] Sinwar, Deepak, and Rahul Kaushik. "Study of Euclidean and Manhattan distance metrics using simple k-means clustering." Int. J. Res. Appl. Sci. Eng. Technol 2, no. 5 (2014) : 270-274.

[190] Mascarenhas, Sheldon, and Mukul Agarwal. "A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification." In 2021 International conference on disruptive technologies for multi-disciplinary research and applications (CENTCON), vol. 1, pp. 96-99. IEEE, 2021.

[191] https://pjreddie.com/darknet/yolo/

[192] Jung, Sunguk, Hyeonbeom Heo, Sangheon Park, Sung-Uk Jung, and Kyungjae Lee. "Benchmarking Deep Learning Models for Instance Segmentation." Applied Sciences 12, no. 17 (2022) : 8856.

[193] Marzban, Caren. "The ROC curve and the area under it as performance measures." Weather and Forecasting 19, no. 6 (2004) : 1106-1114.

[194] Lipton, Z.C., Elkan, C. and Narayanaswamy, B., 2014. Thresholding classifiers to maximize F1 score. arXiv preprint arXiv :1402.1892.

[195] O'Mahony, Niall, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. "Deep learning vs. traditional computer vision." In Advances in Computer Vision : Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1, pp. 128-144. Springer International Publishing, 2020.

[196] KAGEYAMA, Ryo, Nozomi NAGAMINE, and Hiroki MUKOJIMA. "Train Frontal Obstacle Detection Method with Camera-LiDAR Fusion." Quarterly Report of RTRI 63, no. 3 (2022) : 181-186.

[197] Mukojima, Hiroki, Daisuke Deguchi, Yasutomo Kawanishi, Ichiro Ide, Hiroshi Murase, Masato Ukai, Nozomi Nagamine, and Ryuta Nakasone. "Moving camera background-subtraction for obstacle detection on railway tracks." In 2016 IEEE international conference on image processing (ICIP), pp. 3967-3971. IEEE, 2016.

[198] Weinzaepfel, Philippe, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. "DeepFlow : Large displacement optical flow with deep matching." In Proceedings of the IEEE international conference on computer vision, pp. 1385-1392. 2013.

[199] Coscia, Michele. "Generalized Euclidean Measure to Estimate Distances on Multilayer Networks." ACM Transactions on Knowledge Discovery from Data (TKDD) 16, no. 6 (2022) : 1-22.

[200] Satoh, Yutaka, Shun'ichi Kaneko, Yoshinori Niwa, and Kazuhiko Yamamoto. "Robust object detection using a Radial Reach Filter (RRF)." Systems and Computers in Japan 35, no. 10 (2004) : 63-73.

[201] Tianwen, Xiao, Xu Yongneng, and Yu Huimin. "Research on obstacle detection method of urban rail transit based on multisensor technology." Journal of Artificial Intelligence and Technology 1, no. 1 (2021) : 61-67.

[202] Illingworth, John, and Josef Kittler. "A survey of the Hough transform." Computer vision, graphics, and image processing 44, no. 1 (1988) : 87-116.

[203] Kanopoulos, Nick, Nagesh Vasanthavada, and Robert L. Baker. "Design of an image edge detection filter using the Sobel operator." IEEE Journal of solid-state circuits 23, no. 2 (1988) : 358-367.

[204] von Einem, C., Cramariuc, A., Tschopp, F., Cadena, C. and Siegwart, R., LROD : Long-range obstacle detection for railway driver assistance systems.

[205] Lin, J. and Peng, J., 2023. Adaptive inverse perspective mapping transformation method for ballasted railway based on differential edge detection and improved perspective mapping model. Digital Signal Processing, p.103944.

[206] Mallot, H.A., Bülthoff, H.H., Little, J.J. and Bohrer, S., 1991. Inverse perspective mapping simplifies optical flow computation and obstacle detection. Biological cybernetics, 64(3), pp.177-185.

[207] Otsu, Nobuyuki. "A threshold selection method from gray-level histograms." IEEE transactions on systems, man, and cybernetics 9, no. 1 (1979) : 62-66.

[208] Mohamad, Khairul Anuar, Ahmad Akmal Aziz, and Afishah Alias. "Obstacle Detection System for Railways using IoT Sensors." Evolution in Electrical and Electronic Engineering 1, no. 1 (2020) : 57-63.

[209] Khobragade, Shreyas, Akshata Kinage, Divyanshu Shambharkar, and Abhay Gandhi. "Real-time Track and Anomaly Detection in Complex Railway Environment." In 2022 1st International Conference on the Paradigm Shifts in Communication, Embedded Systems, Machine Learning and Signal Processing (PCEMS), pp. 93-97. IEEE, 2022.

[210] Canny, John. "A computational approach to edge detection." IEEE Transactions on pattern analysis and machine intelligence 6 (1986) : 679-698.

[211] Kiryati, Nahum, Yuval Eldar, and Alfred M. Bruckstein. "A probabilistic Hough transform." Pattern recognition 24, no. 4 (1991) : 303-316.

[212] Zhang, J., 1999. C-Bézier curves and surfaces. Graphical Models and Image Processing, 61(1), pp.2-15.

[213] Kyatsandra, Anand Kumar, R. K. Saket, Sachin Kumar, Kumari Sarita, Aanchal Singh S. Vardhan, and Akanksha Singh S. Vardhan. "Development of trinetra : a sensor based vision enhancement system for obstacle detection on railway tracks." IEEE Sensors Journal 22, no. 4 (2022) : 3147-3156.

[214] Sakurada, Mayu, and Takehisa Yairi. "Anomaly detection using autoencoders with nonlinear dimensionality reduction." Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis. 2014.

[215] Prabhakar, Gowdham, et al. "Obstacle detection and classification using deep learning for tracking in high-speed autonomous driving." 2017 IEEE region 10 symposium (TENSYMP). IEEE, 2017.

[216] Garnett, Noa, et al. "Real-time category-based and general obstacle detection for autonomous driving." Proceedings of the IEEE International Conference on Computer Vision Workshops. 2017.

[217] He, Kaiming, et al. "Mask r-cnn." Proceedings of the IEEE international conference on computer vision. 2017.

[218] Zendel, Oliver, et al. "RailSem19 : A dataset for semantic rail scene understanding." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2019.

[219] Ke, Muyuan, Chunyi Lin, and Qinghua Huang. "Anomaly detection of Logo images in the mobile phone using convolutional autoencoder." 2017 4th International Conference on Systems and Informatics (ICSAI). IEEE, 2017.

[220] Chow, Jun Kang, et al. "Anomaly detection of defects on concrete structures with the convolutional autoencoder." Advanced Engineering Informatics 45 (2020) : 101105.

[221] Goodfellow, Ian J., et al. "Generative adversarial networks." arXiv preprint arXiv :1406.2661 (2014).

[222] Donahue, Jeff, Philipp Krähenbühl, and Trevor Darrell. "Adversarial feature learning." arXiv preprint arXiv :1605.09782 (2016).

[223] Akcay, Samet, Amir Atapour-Abarghouei, and Toby P. Breckon. "Ganomaly : Semi-supervised anomaly detection via adversarial training." Asian conference on computer vision. Springer, Cham, 2018.

[224] Zenati, Houssam, et al. "Efficient gan-based anomaly detection." arXiv preprint arXiv :1802.06222 (2018).

[225] Pu, Yong-Ren, Li-Wei Chen, and Su-Hsing Lee. "Study of moving obstacle detection at railway crossing by machine vision." Y.-R. Pu. Informational Technology Journal 13.16 (2014) : 2611-2618.

[226] Pavlović, Milan, Nenad T. Pavlović, and Vukašin Pavlović. "Methods for detection of obstacles on the railway level crossing." Proc. 17th Scientific-Expert Conference on Railways RAILCON '16. 2016.

[227] Mukojima, Hiroki, et al. "Moving camera background-subtraction for obstacle detection on railway tracks." 2016 IEEE international conference on image processing (ICIP). IEEE, 2016.

[228] Rodriguez, LA Fonseca, Jonny Alexander Uribe, and JF Vargas Bonilla. "Obstacle detection over rails using hough transform." 2012 XVII Symposium of Image, Signal Processing, and Artificial Vision (STSIVA). IEEE, 2012.

[229] Yu, Mingyang, Peng Yang, and Sen Wei. "Railway obstacle detection algorithm using neural network." AIP Conference Proceedings. Vol. 1967. No. 1. AIP Publishing LLC, 2018.

[230] Wang, Zhangyu, et al. "Efficient rail area detection using convolutional neural network." IEEE Access 6 (2018) : 77656-77664.

[231] Botchkarev, Alexei. "Performance metrics (error measures) in machine learning regression, forecasting and prognostics : Properties and typology." arXiv preprint arXiv :1809.03006 (2018).

[232] Hore, Alain, and Djemel Ziou. "Image quality metrics : PSNR vs. SSIM." 2010 20th international conference on pattern recognition. IEEE, 2010.

[233] Wang, Zhou, et al. "Image quality assessment : from error visibility to structural similarity." IEEE transactions on image processing 13.4 (2004) : 600-612.

[234] Kingma, Diederik P., and Jimmy Ba. "Adam : A method for stochastic optimization." arXiv preprint arXiv :1412.6980 (2014).

[235] Liu, Liyuan, et al. "On the variance of the adaptive learning rate and beyond." arXiv preprint arXiv :1908.03265 (2019).

[236] Ginsburg, Boris, et al. "Training Deep Networks with Stochastic Gradient Normalized by Layerwise Adaptive Second Moments." (2019).

[237] Manessi, Franco, and Alessandro Rozza. "Learning combinations of activation functions." 2018 24th International Conference on Pattern Recognition (ICPR). IEEE, 2018.

[238] Nair, Vinod, and Geoffrey E. Hinton. "Rectified linear units improve restricted boltzmann machines." Icml. 2010.

[239] Clevert, Djork-Arné, Thomas Unterthiner, and Sepp Hochreiter. "Fast and accurate deep network learning by exponential linear units (elus)." arXiv preprint arXiv :1511.07289 (2015).

[240] Klambauer, Günter, et al. "Self-normalizing neural networks." arXiv preprint arXiv :1706.02515 (2017).

[241] Misra, Diganta. "Mish : A self regularized non-monotonic neural activation function." arXiv preprint arXiv :1908.08681 4 (2019).

[242] Ramachandran, Prajit, Barret Zoph, and Quoc V. Le. "Searching for activation functions." arXiv preprint arXiv :1710.05941 (2017).

[243] Misra, Diganta, Mila : Controlling Minima Concavity in Activation Function, https ://github.com/digantamisra98/Mila

[244] Misra, Diganta, SharkFin : A Modified Version of ReLU, https ://github.com/digantamisra98/SharkFin

[245] Glorot, Xavier, and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." Proceedings of the thirteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, 2010.

[246] Liono, pointcom, AB-VIEW Un cheval sur la voie sncf en Ariège, https ://www.youtube.com/watch ?v=N8DYcNRkauY

[247] Nwankpa, Chigozie, et al. "Activation functions : Comparison of trends in practice and research for deep learning." arXiv preprint arXiv :1811.03378 (2018).

[248] Link to the first segment's results : https://youtu.be/kXr2otmpkM8 .

[249] Link to the second segment's results : https://youtu.be/fv7p5uPyLds

[250] Borghesi, Andrea, Andrea Bartolini, Michele Lombardi, Michela Milano, and Luca Benini. "Anomaly detection using autoencoders in high performance computing systems." In Proceedings of the AAAI Conference on artificial intelligence, vol. 33, no. 01, pp. 9428-9433. 2019.

[251] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." Advances in neural information processing systems 30 (2017).

[252] Chowdhary, KR1442, and K. R. Chowdhary. "Natural language processing." Fundamentals of artificial intelligence (2020) : 603-649.

[253] Niu, Zhaoyang, Guoqiang Zhong, and Hui Yu. "A review on the attention mechanism of deep learning." Neurocomputing 452 (2021) : 48-62.

[254] Alexey Dosovitskiy et al. “An image is worth 16x16 words : Transformers for image recognition at scale”. In : arXiv preprint arXiv :2010.11929 (2020).

[255] Guo, Meng-Hao, Tian-Xing Xu, Jiang-Jiang Liu, Zheng-Ning Liu, Peng-Tao Jiang, Tai-Jiang Mu, Song-Hai Zhang, Ralph R. Martin, Ming-Ming Cheng, and Shi-Min Hu. "Attention mechanisms in computer vision : A survey." Computational visual media 8, no. 3 (2022) : 331-368.

[256] Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani et al. "An image is worth 16x16 words : Transformers for image recognition at scale." arXiv preprint arXiv :2010.11929 (2020).

Apprentissage profond non-supervisé : Application à la détection de situations anormales dans l’environnement du train autonome © 2023 by Amine Boussik is licensed under CC BY-NC-ND 4.0