HIGHER TECHNICAL SCHOOL OF ENGINEERING
(ICAI)

Bachelors Degree in Telecommunications Engineering

ANOMALY DETECTION IN RAILWAY INFRASTRUCTURE BASED ON 3D POINT CLOUD DATA USING NEURAL NETWORKS

Author
Lucía Hernández Fernández

Supervised by
Miguel Ángel Sanz Bobi

Madrid, Spain
August 2025

Lucía Hernández Fernández, declara bajo su responsabilidad, que el Proyecto con título **ANOMALY DETECTION IN RAILWAY INFRASTRUCTURE BASED ON 3D POINT CLOUD DATA USING NEURAL NETWORKS** presentado en la ETS de Ingeniería (ICAI) de la Universidad Pontificia Comillas en el curso académico 2020/21 es de su autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Firmado por SANZ BOBI MIGUEL ANGEL - ***6599** el día 26/08/2025 con un certificado emitido por AC FNMT Usuarios

Fdo.: ..... Fecha: ..... / ..... / .....

Autoriza la entrega:

El Director del Proyecto

Miguel Ángel Sanz Bobi

Fdo.: ..... Fecha: ..... / ..... / .....

V. B. del Coordinador de Proyectos

Nombre del Coordinador

Fdo.: ..... Fecha: ..... / ..... / .....

HIGHER TECHNICAL SCHOOL OF ENGINEERING
(ICAI)

Bachelors Degree in Telecommunications Engineering

ANOMALY DETECTION IN RAILWAY INFRASTRUCTURE BASED ON 3D POINT CLOUD DATA USING NEURAL NETWORKS

Author
Lucía Hernández Fernández

Supervised by
Miguel Ángel Sanz Bobi

Madrid, Spain
August 2025

ANOMALY DETECTION IN RAILWAY INFRASTRUCTURE BASED ON 3D POINT CLOUD DATA USING NEURAL NETWORKS

Author: Lucía Hernández Fernández
Supervisor: Miguel Ángel Sanz Bobi
Collaborating entity: ICAI - Universidad Pontificia de Comillas

ABSTRACT

Accurate analysis of railway infrastructure is very important for ensuring safety, reliability, and efficient operation of transportation systems, as it provides essential information for monitoring the condition of critical components, detecting changes over time, and anomalies. Aditionally, it allows engineers and operators to evaluate structural irregularities, assess the overall state of the infrastructure, and guide preventive interventions with precision.

This research paper presents a comprehensive study focused on the segmentation and classification of railway infrastructure components from LiDAR point clouds, and to automate the identification of structural elements, such as cables, posts, and tracks, exploring the potential of Deep Learning techniques to do so.

Three complementary approaches are investigated in this study. Firstly, a binary classification model is used to distinguish railway infrastructure from non-infrastructure elements within the LiDAR point clouds. Secondly, a semantic segmentation framework is developed to separate the main structural components. Finally, an anomaly detection module is designed to specifically identify irregularities in overhead cables. The study evaluates and compares the performance of these approaches in terms of accuracy, effectiveness, and practical applicability for intelligent infrastructure monitoring.

Beyond the specific application to the railway sector, the methodologies developed in this study can be used in other sectors involving LiDAR point cloud analysis, like different types of civil infrastructure, such as bridges, roads, or power lines, offering a broader range of applications.

Keywords: *LiDAR, railway infrastructure, Deep Learning, anomaly detection, semantic segmentation*

1. Introduction

Railway transport has always been important for industrial and social development, offering an efficient and sustainable alternative to other modes of transportation. Its lower environmental impact compared to road and air travel[20] has established the goal for expansion of high-speed and freight traffic across Europe. However, the ability to achieve these goals depends critically on the reliability and safety of the railway infrastructure, emerging technologies such as LiDAR mapping, drone-based surveys, and machine learning are transforming this traditional inspections by enabling faster, more accurate, and more cost-efficient monitoring of railway assets.

The main purpose of this project is to use deep learning techniques on 3D point cloud data for developing models capable of processing point clouds, classifying infrastructure components, and detect anomalies. The project also lays the groundwork for predictive maintenance applications, anticipating failures before they occur. Following these objectives, it contributes to the modernization of railway management and supports the transition toward a more digitalized sector.

Early works demonstrated the feasibility of automated classification of tracks, catenaries, and masts using mobile LiDAR data[5], while later studies introduced larger datasets and segmentation frameworks that improved scalability [16, 41, 14]. The adoption of neural network architectures such as PointNet++, KPConv, and SPVConv, consolidated deep learning as the state of the art for semantic segmentation in complex railway environments[30, 67]. Anomaly detection has followed a similar trajectory, integrating convolutional neural networks to detect fastener defects[13], while GANs and meta-learning have reduced inspection time and effort[11]. Together, these approaches show the growing role of AI for the continuous and reliable monitoring of railway systems.

2. Methodology

The methodology of this project is structured into four main stages. The first phase, **data exploration**, consists of analyzing the raw LiDAR dataset to assess its quality, identify irregularities, and evaluate class balance, producing a complete exploratory report. The second phase, **data processing**, applies normalization, filtering, splitting, and resampling to prepare reliable and balanced datasets for model training.

The third phase, **modelling**, integrates classification, segmentation, and anomaly detection. It begins with a binary classification distinguishing infrastructure from non-infrastructure, followed by a semantic segmentation of components using a PointNet-based deep learning model, and concludes with a dedicated anomaly detection system focused on overhead cables. Finally, the fourth phase, **testing and results**, evaluates model performance through quantitative metrics and visual 3D representations, allowing both accurate validation and intuitive inspection of detected anomalies.

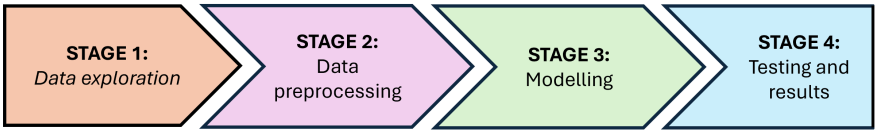


Figure 1: Project workflow

3. Description of the models

Railway presence classification

This stage changes the original multi-class dataset into a binary problem: distinguishing between **infrastructure points** (rails, poles, cables) and **non-infrastructure points** (ground, vegetation, other objects), to filter irrelevant data at an early stage and reducing complexity. The features include LiDAR coordinates, intensity, and colors.

XGBoost

XGBoost is a gradient-boosted decision tree algorithm that builds an additive ensemble of weak learners. At iteration M, the prediction is given by:

$$F_M(x) = \sum_{m=1}^{M} \eta f_m(x),$$

The main hyperparameters include *n_estimators*, *learning_rate*, *max_depth*, *min_child_weight*, *subsample*, *colsample_bytree*, and regularization terms α and λ. Class imbalance is corrected using *scale_pos_weight*. Models were compared 4.1 using Precision–Recall AUC as the primary metric, with F1-score and Balanced Accuracy as secondary criteria, and decision thresholds were optimized on the validation folds to maximize F1.

The final configuration balanced tree achieved the best trade-off across PR-AUC and F1-score among the XGBoost candidates. On the validation split, results were consistent with training, confirming good generalization.

Model	**depth**	**learning rate**	**estimators**	**Threshold**
MODEL 4	4	0.10	500	0.37

Table 1: Final model with all the parameters

LightGBM

LightGBM is also a gradient-boosted decision tree algorithm but employs histogram-based split finding and a leaf-wise growth strategy with depth constraints. The optimization problem is similar to XGBoost:

$$\mathcal{L} = \sum_{i=1}^{n} \ell(y_i, \hat{y}_i) + \Omega(f_m),$$

but training is accelerated by techniques such as Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB), which approximate gradient distributions and reduce feature dimensionality. This allows LightGBM to scale efficiently to large, sparse datasets.

Key hyperparameters include *num_ leaves*, *max_ depth*, *min_ data_ in_ leaf*, *feature_ fraction*, *bagging_ fraction*, *bagging_ freq*, and *learning_ rate*m and to handle imbalance,*scale_ pos_ weight* parameter was used. Validation with early stopping was applied, and models were selected according to PR-AUC and F1-score 4.5.

The selected LightGBM configuration produced the best PR-AUC and F1-score for the validation, confirming generalization and fast inference capability.

Model	**learning rate**	**depth**	**estimators**	**leaves**	**min child samples**	**Threshold**
Model 3	0.10	9	200	–	200	0.39

Table 2: Final LightGBM model and threshold applied

Railway component segmentation: PointNet

The goal of this stage is to segment the structural components of the railway infrastructure at point level, each point must now be assigned to one of three categories: **rails**, **poles**, or **wires**. PointNet was selected as the model because it directly processes raw point clouds without voxelization or projection, preserving geometric fidelity. The network applies shared multilayer perceptrons (MLPs) to each point independently, aggregates global features through symmetric functions, and outputs per-point class scores.

In this project, the original PointNet model was adapted to the specific characteristics of railway data. First, the input representation was extended from coordinates (x, y, z) to a seven-dimensional feature vector including spatial coordinates, RGB color, and intensity: $\mathbf{p}_i = [x_i, y_i, z_i, r_i, g_i, b_i, I_i], \quad i = 1, \ldots, N,$

Moreover, the architecture was designed to accept variable-size point clouds $(N, 7)$, making it fully point-wise and flexible across diverse railway scenes. A 3×3 T-Net was applied only to the coordinate subset to normalize spatial variations, while the feature-transform block of the original PointNet was omitted to avoid distortions when combining geometric and non-geometric attributes.

The model was trained using the *Adam optimizer*, which adapts learning rates for each parameter through first and second moment estimates of the gradients:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \qquad v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2,$$

with bias correction and parameter updates. A learning rate of 0.001 and L_2 weight decay of 10^{-4} were adopted, and to address class imbalance, class-specific weights were incorporated into the weighted cross-entropy loss, ensuring that minority classes such as poles and wires contributed proportionally to the optimization.

Training was organized into sessions of 100 epochs each. After every epoch, the validation mean Intersection-over-Union (mIoU) was computed:

$$\mathrm{IoU}_c = \frac{TP_c}{TP_c + FP_c + FN_c}, \qquad \mathrm{mIoU} = \frac{1}{m} \sum_{c=1}^{m} \mathrm{IoU}_c,$$

and the model with the highest validation mIoU was checkpointed. Two main training rounds (0–100 and 100–200 epochs) were performed, yielding models with comparable performance, with the best-performing configuration achieved a validation accuracy above 0.88, a balanced accuracy of 0.86, and a mean IoU of 0.77.

Best epoch	Train accuracy	Train mIoU	Validation mIoU
146	0.8631	0.7239	0.7703

Table 3: Training and validation performance of the best model from epoch 100 to 200.

Cable anomalies detection

This stage automatically identifies irregularities in the catenary system, one of the most critical elements of railway infrastructure. Detecting anomalies in cables is essential for a safe and reliable operation. This task is performed only on validation and test datasets to simulate real-world unseen data.

Before anomalies can be detected, it is necessary to isolate each cable from the overall set of points classified as *cable* in the segmentation step. To achieve this, DBSCAN clustering is applied to the 3D coordinates:

$$\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\} \subset \mathbb{R}^3, \quad \mathcal{P} = \bigcup_{k=1}^{K} \mathcal{C}_k, \quad \mathcal{C}_i \cap \mathcal{C}_j = \varnothing.$$

Each cluster $\mathcal{C}_k$ corresponds to a single cable, while noise points are labeled as -1. Parameters $\varepsilon = 0.045$ and $minPts = 10$ were tuned on validation data and fixed for testing, ensuring generalization.

Several anomaly detectors were designed, operating either at the global or point level of each cable:

Cable alignment: To check whether a cable follows a straight trajectory, PCA is applied to its points $\{\mathbf{p}_i\}$ to extract the dominant axis $\mathbf{v}_1$. The explained variance ratio

$$R = \frac{\sigma^2_{\text{axis}}}{\sigma^2_{\text{total}}}$$

measures alignment. If $R < \tau = 0.9$, the cable is considered anomalous.

Cable inclination: The orientation vector $\mathbf{e}_{cable}$ is compared to the reference axis or plane. An anomaly is detected if the inclination angle

$$\theta = \arccos(|\mathbf{e}_{cable} \cdot \mathbf{e}_{ref}|)$$

exceeds a threshold $\tau = 5$, indicating collapse or abnormal tilt.

Point deviation: Each point $\mathbf{p}_i$ is projected onto the cable's main axis, producing $\widehat{\mathbf{p}}_i$. The orthogonal distance

$$d_i = \|\mathbf{p}_i - \widehat{\mathbf{p}}_i\|_2$$

is computed, and the point is anomalous if $d_i > \delta = 0.05$.

Segment change in direction: Cables are split into segments $\mathcal{S}_k$. PCA gives each segment orientation $\mathbf{e}_k$. The angle between consecutive segments

$$\theta_k = \arccos(|\mathbf{e}_k \cdot \mathbf{e}_{k+1}|)$$

is compared to a threshold $\tau = 30$.

Space between points: After projecting all points onto the main axis, the gaps

$$\Delta_j = \pi_{j+1} - \pi_j$$

are computed. If $\Delta_j > \delta = 0.02$, the gap is marked anomalous, as it may correspond to a break or missing cable section.

4. Results

Railway presence classification

XGBoost

The XGBoost model achieved a **global accuracy** of *0.7579* and a **balanced accuracy** of *0.7304* on the test dataset, with ROC AUC and PR AUC scores of *0.8399* and *0.7962*, respectively. While Class 0 (non-infrastructure) maintained stable performance (F1-score = 0.8114), the Class 1 (railway presence) had a F1-score to 0.6621.

Class	**Precision**	**Recall**	**F1-score**	**IoU**
Class 0	0.7618	0.8678	0.8114	0.6826
Class 1	0.7494	0.5931	0.6621	0.4949

Table 4: Class-specific evaluation metrics on the test set for XGBoost model

LightGBM

The LightGBM model obtained higher overall accuracy on the test set (**0.7618**) but a similar balanced accuracy (*0.7298*), with ROC AUC and PR AUC values of *0.8300* and *0.7875*. Class 0 reached strong results (F1-score = 0.8177), while Class 1 had a F1-score of 0.6567.

Class	Precision	Recall	F1-score	IoU
Class 0	0.7562	0.8901	0.8177	0.6916
Class 1	0.7755	0.5694	0.6567	0.4889

Table 5: Class-specific evaluation metrics on the test set for the LightGBM model

Railway component segmentation: PointNet

Epochs 0-100: The PointNet model trained during the first 100 epochs achieved strong segmentation performance on the test set, with a **global accuracy** of *0.8687*, **balanced accuracy** of *0.8347*, and **mean IoU** of *0.7344*. At the class level, rails (Class 3) were segmented with excellent reliability (F1 = 0.9738, IoU = 0.9489), while cables (Class 5) also reached competitive values (F1 = 0.8299, IoU = 0.7093). In contrast, posts (Class 4) remained the most challenging, with F1 = 0.7055 and IoU = 0.5450 due to frequent confusion with cables.

Class	Precision	Recall	F1-score	IoU
Class 3	0.9564	0.9918	0.9738	0.9489
Class 4	0.8119	0.6237	0.7055	0.5450
Class 5	0.7784	0.8887	0.8299	0.7093

Table 6: Class-specific evaluation metrics for the PointNet model on the test set from epoch 0 to 100.

Epochs 100-200: In the second training phase, the best model achieved a **global accuracy** of *0.8242*, **balanced accuracy** of *0.8001*, and **mean IoU** of *0.6782*, showing a noticeable drop compared to the earlier epoch group. Rails (Class 3) remained the best-performing category (F1 = 0.9320, IoU = 0.8727), while cables (Class 5) maintained reasonable values (F1 = 0.8137, IoU = 0.6859). However, posts (Class 4) had F1 = 0.6449 and IoU = 0.4759, reflecting a high rate of misclassification into the cable class.

Class	Precision	Recall	F1-score	IoU
Class 3	0.9458	0.9992	0.9717	0.9450
Class 4	0.8455	0.6704	0.7478	0.5972
Class 5	0.8204	0.9034	0.8599	0.7543

Table 7: Class-specific evaluation metrics for the best model on the validation subset from epoch 0 to 100.

Cable anomalies detection

The evaluation of cable anomalies combined quantitative indicators with qualitative visual inspection. For each detector, numerical outputs were stored in structured dataframes, while visualizations with a green/red code confirmed the presence of an anomaly.

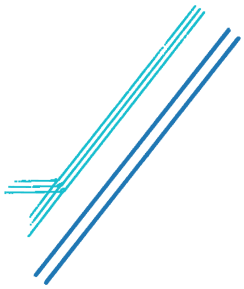
Cable Alignment: The dataframe records, for each cable, the file of origin, cable identifier, number of points, explained variance ratio of the first principal axis, and a binary flag indicating whether alignment falls below the threshold 0.90.

Cable Inclination: Each row includes the file, cable identifier, number of points, the measured inclination angle (in degrees) relative to a dominant axis or plane, and a binary anomaly flag. Angles above the 5° threshold are flagged as abnormal orientations.

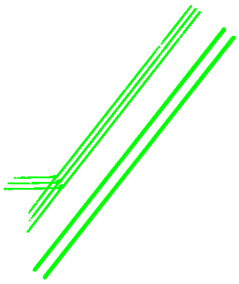
Point Deviation: The output dataframe contains the mean, maximum, and standard deviation of orthogonal deviations per cable, together with the number and indices of points exceeding the deviation threshold 0.05.

Segment Change in Direction: For each pair of consecutive segments, the results show the file, cable ID, point range of the segments, the computed angle between their dominant axes, and a flag if it exceeds 30°.

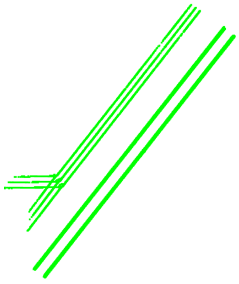
Space Between Points: This detector reports only anomalous cases, recording the file, cable ID, indices of the two points forming the gap, the gap size, and an anomaly flag if distance is greater than the threshold 0.02.



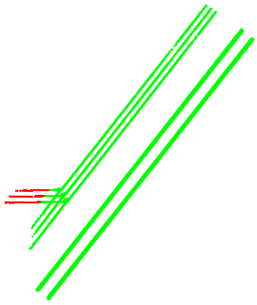
(a) Cable separation



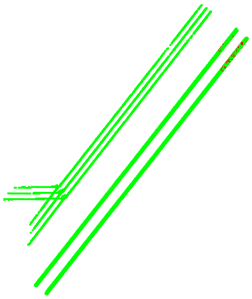
(b) Cable alignment



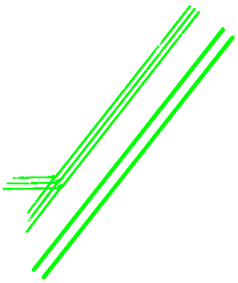
(c) Cable inclination



(d) Point per point deviation



(e) Segment change direction



(f) Distance between points

Figure 2: Cable anomalies detection results

5. Conclusions

The project combined PointNet-based segmentation, gradient-boosted tree classifiers, and custom anomaly detection models for analyzing railway infrastructure from LiDAR point clouds. Results confirmed consistent quantitative performance, supported by visual inspections that facilitated interpretability, making the methodology applicable even for non-specialized users in operational contexts.

Beyond the methodological contributions, the work shows the potential to strengthen predictive maintenance in the railway sector by reducing inspection costs, increasing safety, and supporting the digitalization of infrastructure management. Moreover, it is transferable to other industries where anomaly detection is critical, such as energy, civil engineering, telecommunications, and aerospace.

DETECCIÓN DE ANOMALÍAS EN INFRAESTRUCTURAS FERROVIARIAS A PARTIR DE NUBES DE PUNTOS 3D USANDO REDES NEURONALES

Autor: Lucía Hernández Fernández
Director: Miguel Ángel Sanz Bobi
Entidad colaboradora: ICAI - Universidad Pontificia de Comillas

RESUMEN

El análisis preciso de la infraestructura ferroviaria es de gran importancia para garantizar la seguridad, la fiabilidad y la eficiencia en los sistemas de transporte, ya que proporciona información para el monitoreo del estado de sus componentes, y la detección de cambios a lo largo del tiempo y anomalías. Además, permite a los ingenieros y operadores evaluar irregularidades estructurales para orientar intervenciones preventivas con precisión.

Este trabajo presenta un estudio exhaustivo centrado en la segmentación y clasificación de los componentes de la infraestructura ferroviaria a partir de nubes de puntos LiDAR, con el objetivo de automatizar la identificación de elementos estructurales, explorando las técnicas de Deep Learning.

En este estudio se investigan tres enfoques. Primero, se emplea un modelo de clasificación binaria para distinguir los elementos de infraestructura ferroviaria de aquellos que no lo son. En segundo lugar, se desarrolla una segmentación semántica para separar los principales componentes estructurales. Finalmente, se diseña un módulo de detección de anomalías para identificar irregularidades en los cables aéreos. El estudio evalúa y compara el rendimiento de estos enfoques en términos de precisión, eficacia y aplicabilidad para el monitoreo inteligente de la infraestructura.

Más allá de la aplicación en este sector, las metodologías desarrolladas en este estudio pueden aplicarse en los que se analicen nubes de puntos LiDAR, como diferentes infraestructuras civiles, como puentes o túneles, ofreciendo un abanico más amplio de aplicaciones.

Palabras clave: *LiDAR, infraestructura ferroviaria, Deep Learning, detección de anomalías, segmentación semántica*

1. Introducción

El transporte ferroviario siempre ha desempeñado un papel fundamental en el desarrollo industrial y social, al ser una alternativa eficiente y sostenible para transportarse. Su menor impacto ambiental en comparación con el transporte por carretera y aéreo[20] ha impulsado el objetivo de expandir las redes de alta velocidad y el tráfico de mercancías en toda Europa. No obstante, la consecución de estos objetivos depende de la fiabilidad y seguridad de la infraestructura ferroviaria. Tecnologías emergentes como el mapeo LiDAR, las inspecciones con drones y el aprendizaje automático están transformando las inspecciones tradicionales, al permitir una monitorización más rápida, precisa y rentable de los activos ferroviarios.

El propósito principal de este proyecto es aplicar técnicas de Deep Learning sobre datos de nubes de puntos 3D para desarrollar modelos capaces de procesar dichas nubes, clasificar componentes de la infraestructura y detectar anomalías. El proyecto también sienta las bases para aplicaciones de mantenimiento predictivo, anticipando fallos antes de que ocurran. En línea con estos objetivos, contribuye a la modernización de la gestión ferroviaria y respalda la transición hacia un sector más digitalizado.

Los primeros trabajos demostraron la viabilidad de la clasificación automática de vías, catenarias y postes empleando datos LiDAR móviles[5], mientras que estudios posteriores introdujeron conjuntos de datos más amplios y marcos de segmentación que mejoraron la escalabilidad [16, 41, 14]. Con la adopción de arquitecturas de redes neuronales como PointNet++, KPConv y SPVConv, el aprendizaje profundo se consolidó como el estado del arte en segmentación semántica de entornos ferroviarios complejos[30, 67]. La detección de anomalías ha seguido una trayectoria similar, integrando redes convolucionales para identificar defectos en fijaciones[13], mientras que técnicas como GANs y meta-learning han reducido los tiempos y esfuerzos de inspección[11]. En conjunto, estos enfoques evidencian el creciente papel de la inteligencia artificial en la monitorización continua y fiable de los sistemas ferroviarios.

2. Metodología

La metodología de este proyecto se estructura en cuatro etapas principales. La primera fase, **exploración de datos**, consiste en analizar el conjunto de datos LiDAR en bruto para evaluar su calidad, identificar posibles irregularidades y examinar el equilibrio de clases, generando un informe exploratorio completo. La

segunda fase, **procesamiento de datos**, aplica técnicas de normalización, filtrado, partición y remuestreo con el fin de preparar conjuntos de datos fiables y balanceados para el entrenamiento de los modelos.

La tercera fase, **modelado**, integra clasificación, segmentación y detección de anomalías. Comienza con una clasificación binaria que distingue entre infraestructura y no-infraestructura, continúa con una segmentación semántica de componentes mediante un modelo de aprendizaje profundo basado en PointNet, y finaliza con un sistema específico de detección de anomalías centrado en los cables aéreos. Finalmente, la cuarta fase, **pruebas y resultados**, evalúa el rendimiento de los modelos a través de métricas cuantitativas y representaciones visuales en 3D, lo que permite tanto una validación precisa como una inspección intuitiva de las anomalías detectadas.

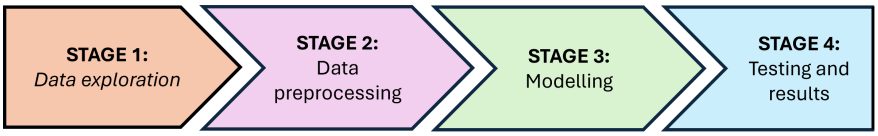


Figure 3: Flujo de trabajo del proyecto

3. Descripción de los modelos

Clasificación de la presencia de infraestructura ferroviaria

Esta etapa transforma el conjunto de datos original de múltiples clases en un problema binario: distinguir entre **puntos de infraestructura** (raíles, postes, cables) y **puntos de no-infraestructura** (suelo, vegetación, otros objetos). De esta manera, se filtran datos irrelevantes en una fase temprana, reduciendo la complejidad del problema. Las características empleadas incluyen las coordenadas LiDAR, la intensidad y los valores de color.

XGBoost

XGBoost es un algoritmo de árboles de decisión potenciados por gradiente que construye un ensamble aditivo de clasificadores débiles. En la iteración M, la predicción se define como:

$$F_M(x) = \sum_{m=1}^{M} \eta f_m(x),$$

Los principales hiperparámetros incluyen *n_ estimators*, *learning_ rate*, *max_ depth*, *min_ child_ weight*, *subsample*, *colsample_ bytree*, así como los términos de regularización α y λ. El desbalance de clases se corrige mediante el parámetro *scale_ pos_ weight*. Los modelos fueron comparados 4.1 utilizando el área bajo la curva Precisión–Recall (PR-AUC) como métrica principal, y el F1-score y la Precisión Balanceada como criterios secundarios. Los umbrales de decisión fueron optimizados en las particiones de validación para maximizar el F1.

La configuración final del árbol balanceado alcanzó el mejor compromiso entre PR-AUC y F1-score entre los candidatos de XGBoost. En el conjunto de validación, los resultados fueron consistentes con los de entrenamiento, confirmando una buena capacidad de generalización.

Modelo	**depth**	**learning rate**	**estimators**	**Threshold**
MODELO 4	4	0.10	500	0.37

Table 8: Modelo final con todos los parámetros

LightGBM

LightGBM es también un algoritmo de árboles de decisión potenciados por gradiente, pero emplea una estrategia de búsqueda de particiones basada en histogramas y un crecimiento por hojas con restricciones de profundidad. El problema de optimización es similar al de XGBoost:

$$\mathcal{L} = \sum_{i=1}^{n} \ell(y_i, \hat{y}_i) + \Omega(f_m),$$

pero el entrenamiento se acelera mediante técnicas como *Gradient-based One-Side Sampling* (GOSS) y *Exclusive Feature Bundling* (EFB), que aproximan las distribuciones de gradiente y reducen la dimensionalidad de las características. Esto permite que LightGBM escale de manera eficiente en conjuntos de datos grandes y dispersos.

Los hiperparámetros clave incluyen *num_ leaves*, *max_ depth*, *min_ data_ in_ leaf*, *feature_ fraction*, *bagging_ fraction*, *bagging_ freq* y *learning_ rate*. Para manejar el desbalance de clases se utilizó el parámetro *scale_ pos_ weight*. Se aplicó validación con *early stopping*, y los modelos fueron seleccionados según las métricas PR-AUC y F1-score 4.5.

La configuración seleccionada de LightGBM produjo el mejor PR-AUC y F1-score en la validación, confirmando su capacidad de generalización y rapidez de inferencia.

Modelo	**learning rate**	**depth**	**estimators**	**leaves**	**Mín. child samples**	**Threshold**
Modelo 3	0.10	9	200	–	200	0.39

Table 9: Modelo final de LightGBM y umbral aplicado

Segmentación de los componentes ferroviarios: PointNet

El objetivo de esta etapa es segmentar los componentes estructurales de la infraestructura ferroviaria a nivel de punto, de modo que cada punto de la nube de datos debe ser asignado a una de tres categorías: **raíles**, **postes** o **cables**. Se seleccionó PointNet como modelo porque procesa directamente nubes de puntos sin necesidad de voxelización ni proyección, preservando así la fidelidad geométrica. La red aplica perceptrones multicapa compartidos (MLPs) a cada punto de forma independiente, agrega características globales mediante funciones simétricas y genera puntuaciones de clase por punto.

En este proyecto, el modelo original PointNet fue adaptado a las características específicas de los datos ferroviarios. En primer lugar, la representación de entrada se amplió desde las coordenadas (x, y, z) a un vector de características de siete dimensiones que incluye coordenadas espaciales, color RGB e intensidad: $\mathbf{p}_i = [x_i, y_i, z_i, r_i, g_i, b_i, I_i], \quad i = 1, \ldots, N,$

Además, la arquitectura fue diseñada para aceptar nubes de puntos de tamaño variable $(N, 7)$, lo que la hace completamente punto a punto y flexible en diferentes escenas ferroviarias. Se aplicó un T-Net de 3×3 únicamente al subconjunto de coordenadas para normalizar las variaciones espaciales, mientras que el bloque de transformación de características del PointNet original fue omitido con el fin de evitar distorsiones al combinar atributos geométricos y no geométricos.

El modelo fue entrenado utilizando el *optimizador Adam*, que adapta las tasas de aprendizaje de cada parámetro mediante estimaciones de primer y segundo momento de los gradientes:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \qquad v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2,$$

con corrección de sesgo y actualización de parámetros. Se adoptó una tasa de aprendizaje de 0.001 y un decaimiento L_2 de 10^{-4}, y para abordar el desbalance de clases, se incorporaron pesos específicos en la función de pérdida de entropía cruzada ponderada, garantizando que las clases minoritarias como postes y cables contribuyeran proporcionalmente al proceso de optimización.

El entrenamiento se organizó en sesiones de 100 épocas cada una. Después de cada época, se calculó el valor de la métrica *Intersection-over-Union* media (mIoU) en validación:

$$\mathrm{IoU}_c = \frac{TP_c}{TP_c + FP_c + FN_c}, \qquad \mathrm{mIoU} = \frac{1}{m}\sum_{c=1}^{m} \mathrm{IoU}_c,$$

y se almacenó el modelo con mayor mIoU de validación. Se realizaron dos rondas principales de entrenamiento (0–100 y 100–200 épocas), obteniéndose modelos con desempeños comparables. La mejor configuración alcanzó una precisión en validación superior a 0.88, una precisión balanceada de 0.86 y un mIoU de 0.77.

Mejor época	**Precisión entrenamiento**	**mIoU entrenamiento**	**mIoU validación**
146	0.8631	0.7239	0.7703

Table 10: Rendimiento en entrenamiento y validación del mejor modelo entre las épocas 100 y 200.

Detección de anomalías en cables

Esta etapa identifica de manera automática las irregularidades en el sistema de catenarias, uno de los elementos más críticos de la infraestructura ferroviaria. La detección de anomalías en los cables es esencial para garantizar un funcionamiento seguro y fiable. Esta tarea se realiza únicamente sobre los conjuntos de datos de validación y prueba, con el objetivo de simular escenarios reales con datos no vistos previamente.

Antes de proceder con la detección de anomalías, es necesario aislar cada cable del conjunto total de puntos clasificados como *cable* en la etapa de segmentación. Para ello, se aplica el algoritmo de clustering DBSCAN a las coordenadas tridimensionales:

$$\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\} \subset \mathbb{R}^3, \quad \mathcal{P} = \bigcup_{k=1}^{K} \mathcal{C}_k, \quad \mathcal{C}_i \cap \mathcal{C}_j = \varnothing.$$

Cada clúster $\mathcal{C}_k$ corresponde a un cable individual, mientras que los puntos de ruido son etiquetados como -1. Los parámetros $\varepsilon = 0.045$ y $minPts = 10$ fueron ajustados en el conjunto de validación y posteriormente fijados en el de prueba, garantizando así la capacidad de generalización.

Se diseñaron varios detectores de anomalías, operando tanto a nivel global como a nivel puntual de cada cable:

Alineación del cable: Para verificar si un cable sigue una trayectoria recta, se aplica PCA a sus puntos $\{\mathbf{p}_i\}$ con el fin de extraer el eje dominante $\mathbf{v}_1$. El cociente de varianza explicada

$$R = \frac{\sigma^2_{\text{axis}}}{\sigma^2_{\text{total}}}$$

mide el grado de alineación. Si $R < \tau = 0.9$, el cable se considera anómalo.

Inclinación del cable: El vector de orientación $\mathbf{e}_{cable}$ se compara con el eje o plano de referencia. Se detecta una anomalía si el ángulo de inclinación

$$\theta = \arccos(|\mathbf{e}_{cable} \cdot \mathbf{e}_{ref}|)$$

supera un umbral $\tau = 5$, lo que indica colapso o inclinación anormal.

Desviación puntual: Cada punto $\mathbf{p}_i$ se proyecta sobre el eje principal del cable, generando $\widehat{\mathbf{p}}_i$. La distancia ortogonal

$$d_i = \|\mathbf{p}_i - \widehat{\mathbf{p}}_i\|_2$$

se calcula, y el punto se considera anómalo si $d_i > \delta = 0.05$.

Cambio de dirección por segmentos: Los cables se dividen en segmentos $\mathcal{S}_k$. PCA proporciona la orientación de cada segmento $\mathbf{e}_k$. El ángulo entre segmentos consecutivos

$$\theta_k = \arccos(|\mathbf{e}_k \cdot \mathbf{e}_{k+1}|)$$

se compara con un umbral $\tau = 30$.

Espacio entre puntos: Tras proyectar todos los puntos sobre el eje principal, se calculan las separaciones

$$\Delta_j = \pi_{j+1} - \pi_j$$

Si $\Delta_j > \delta = 0.02$, la separación se marca como anómala, ya que podría corresponder a una rotura o a una sección faltante del cable.

4. Resultados

Clasificación de la presencia de infraestructura ferroviaria

XGBoost

El modelo XGBoost alcanzó una **exactitud global** de *0.7579* y una **exactitud balanceada** de *0.7304* en el conjunto de prueba, con valores de ROC AUC y PR AUC de *0.8399* y *0.7962*, respectivamente. Mientras que la Clase 0 (no-infraestructura) mantuvo un rendimiento estable (F1-score = 0.8114), la Clase 1 (presencia ferroviaria) obtuvo un F1-score de 0.6621.

Clase	Precisión	Recall	F1-score	IoU
Clase 0	0.7618	0.8678	0.8114	0.6826
Clase 1	0.7494	0.5931	0.6621	0.4949

Table 11: Métricas de evaluación específicas por clase en el conjunto de prueba para el modelo XGBoost

LightGBM

El modelo LightGBM obtuvo una mayor exactitud global en el conjunto de prueba (**0.7618**) pero una exactitud balanceada similar (*0.7298*), con valores de ROC AUC y PR AUC de *0.8300* y *0.7875*, respectivamente. La Clase 0 alcanzó resultados sólidos (F1-score = 0.8177), mientras que la Clase 1 presentó un F1-score de 0.6567.

Clase	Precisión	Recall	F1-score	IoU
Clase 0	0.7562	0.8901	0.8177	0.6916
Clase 1	0.7755	0.5694	0.6567	0.4889

Table 12: Métricas de evaluación específicas por clase en el conjunto de prueba para el modelo LightGBM

Segmentación de los componentes ferroviarios: PointNet

Épocas 0-100: El modelo PointNet entrenado durante las primeras 100 épocas alcanzó un sólido rendimiento en la segmentación sobre el conjunto de prueba, con una **exactitud global** de *0.8687*, **exactitud balanceada** de *0.8347* y un

IoU medio de *0.7344*. A nivel de clase, los raíles (Clase 3) fueron segmentados con una fiabilidad excelente (F1 = 0.9738, IoU = 0.9489), mientras que los cables (Clase 5) también alcanzaron valores competitivos (F1 = 0.8299, IoU = 0.7093). En contraste, los postes (Clase 4) resultaron ser la categoría más desafiante, con F1 = 0.7055 e IoU = 0.5450, debido a la frecuente confusión con los cables.

Clase	**Precisión**	**Recall**	**F1-score**	**IoU**
Clase 3	0.9564	0.9918	0.9738	0.9489
Clase 4	0.8119	0.6237	0.7055	0.5450
Clase 5	0.7784	0.8887	0.8299	0.7093

Table 13: Métricas de evaluación específicas por clase para el modelo PointNet en el conjunto de prueba de la época 0 a 100.

Épocas 100-200: En la segunda fase de entrenamiento, el mejor modelo alcanzó una **exactitud global** de *0.8242*, **exactitud balanceada** de *0.8001* y un **IoU medio** de *0.6782*, mostrando una caída significativa en comparación con el grupo de épocas anterior. Los raíles (Clase 3) se mantuvieron como la categoría con mejor desempeño (F1 = 0.9320, IoU = 0.8727), mientras que los cables (Clase 5) conservaron valores razonables (F1 = 0.8137, IoU = 0.6859). Sin embargo, los postes (Clase 4) obtuvieron un F1 = 0.6449 e IoU = 0.4759, reflejando una elevada tasa de errores de clasificación en la categoría de cables.

Clase	**Precisión**	**Recall**	**F1-score**	**IoU**
Class 3	0.9458	0.9992	0.9717	0.9450
Class 4	0.8455	0.6704	0.7478	0.5972
Class 5	0.8204	0.9034	0.8599	0.7543

Table 14: Métricas de evaluación específicas por clase para el PointNet en el conjunto de pruebas de la época 100 a 200.

Detección de anomalías en cables

La evaluación de anomalías en los cables combinó indicadores cuantitativos con una inspección visual cualitativa. Para cada detector, los resultados numéricos fueron almacenados en *dataframes* estructurados, mientras que las visualizaciones con un código de color verde/rojo confirmaron la presencia de una anomalía.

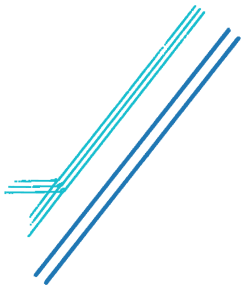
Alineación del cable: El *dataframe* registra, para cada cable, el archivo de origen, el identificador del cable, el número de puntos, el ratio de varianza explicada por el primer eje principal y una bandera binaria que indica si la alineación está por debajo del umbral 0.90.

Inclinación del cable: Cada fila incluye el archivo, el identificador del cable, el número de puntos, el ángulo de inclinación medido (en grados) con respecto a un eje o plano dominante, y una bandera binaria de anomalía. Los ángulos superiores al umbral de 5° se marcan como orientaciones anormales.

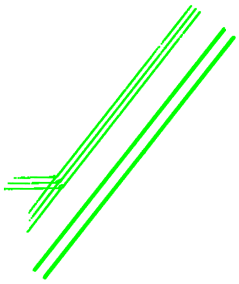
Desviación puntual: El *dataframe* de salida contiene la desviación ortogonal media, máxima y desviación estándar por cable, junto con el número e índices de puntos que superan el umbral de desviación 0.05.

Cambio de dirección por segmentos: Para cada par de segmentos consecutivos, los resultados muestran el archivo, el identificador del cable, el rango de puntos de los segmentos, el ángulo calculado entre sus ejes dominantes y una bandera que indica si supera los 30°.

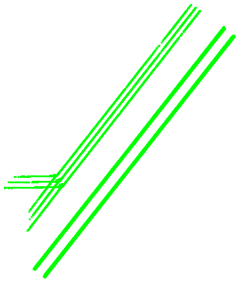
Espacio entre puntos: Este detector informa únicamente de los casos anómalos, registrando el archivo, el identificador del cable, los índices de los dos puntos que forman la discontinuidad, el tamaño del hueco y una bandera de anomalía si la distancia es mayor al umbral 0.02.



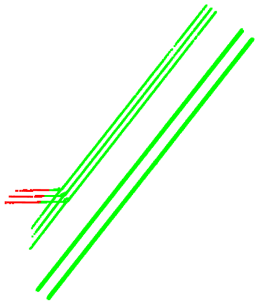
(a) Separación de cables



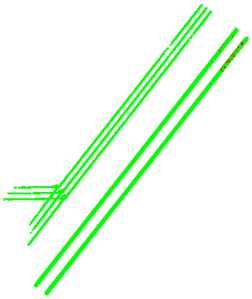
(b) Alineación del cable



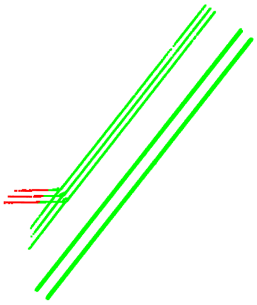
(c) Inclinación del cable



(d) Desviación punto a punto



(e) Cambio de dirección por segmentos



(f) Distancia entre puntos

Figure 4: Resultados de la detección de anomalías en cables

5. Conclusiones

El proyecto combina la segmentación basada en PointNet, clasificadores de árboles potenciados mediante gradiente y modelos personalizados de detección de anomalías para el análisis de infraestructuras ferroviarias a partir de nubes de puntos LiDAR. Los resultados confirmaron un rendimiento cuantitativo consistente, respaldado por inspecciones visuales que facilitaron la interpretabilidad, lo que hace que la metodología sea aplicable incluso para usuarios no especializados en contextos operativos.

Más allá de las contribuciones metodológicas, este trabajo demuestra el potencial para fortalecer el mantenimiento predictivo en el sector ferroviario, al reducir los costos de inspección, aumentar la seguridad y apoyar la digitalización de la gestión de infraestructuras. Asimismo, su aplicabilidad puede extenderse a otras industrias donde la detección de anomalías es crítica, como la energía, la ingeniería civil, las telecomunicaciones y la industria aeroespacial.

Abstract

Accurate analysis of railway infrastructure is very important for ensuring safety, reliability, and efficient operation of transportation systems, as it provides essential information for monitoring the condition of critical components, detecting changes over time, and anomalies. Aditionally, it allows engineers and operators to evaluate structural irregularities, assess the overall state of the infrastructure, and guide preventive interventions with precision.

This research paper presents a comprehensive study focused on the segmentation and classification of railway infrastructure components from LiDAR point clouds, and to automate the identification of structural elements, such as cables, posts, and tracks, exploring the potential of Deep Learning techniques to do so.

Three complementary approaches are investigated in this study. Firstly, a binary classification model is used to distinguish railway infrastructure from non-infrastructure elements within the LiDAR point clouds. Secondly, a semantic segmentation framework is developed to separate the main structural components. Finally, an anomaly detection module is designed to specifically identify irregularities in overhead cables. The study evaluates and compares the performance of these approaches in terms of accuracy, effectiveness, and practical applicability for intelligent infrastructure monitoring.

Beyond the specific application to the railway sector, the methodologies developed in this study can be used in other sectors involving LiDAR point cloud analysis, like different types of civil infrastructure, such as bridges, roads, or power lines, offering a broader range of applications.

Keywords: *LiDAR, railway infrastructure, Deep Learning, anomaly detection, semantic segmentation*

Resumen

El análisis preciso de la infraestructura ferroviaria es de gran importancia para garantizar la seguridad, la fiabilidad y la eficiencia en los sistemas de transporte, ya que proporciona información para el monitoreo del estado de sus componentes, y la detección de cambios a lo largo del tiempo y anomalías. Además, permite a los ingenieros y operadores evaluar irregularidades estructurales para orientar intervenciones preventivas con precisión.

Este trabajo presenta un estudio exhaustivo centrado en la segmentación y clasificación de los componentes de la infraestructura ferroviaria a partir de nubes de puntos LiDAR, con el objetivo de automatizar la identificación de elementos estructurales, explorando las técnicas de Deep Learning.

En este estudio se investigan tres enfoques. Primero, se emplea un modelo de clasificación binaria para distinguir los elementos de infraestructura ferroviaria de aquellos que no lo son. En segundo lugar, se desarrolla una segmentación semántica para separar los principales componentes estructurales. Finalmente, se diseña un módulo de detección de anomalías para identificar irregularidades en los cables aéreos. El estudio evalúa y compara el rendimiento de estos enfoques en términos de precisión, eficacia y aplicabilidad para el monitoreo inteligente de la infraestructura.

Más allá de la aplicación en este sector, las metodologías desarrolladas en este estudio pueden aplicarse en los que se analicen nubes de puntos LiDAR, como diferentes infraestructuras civiles, como puentes o túneles, ofreciendo un abanico más amplio de aplicaciones.

Palabras clave: *LiDAR, infraestructura ferroviaria, Deep Learning, detección de anomalías, segmentación semántica*

To my grandparents, my parents, and my sisters, for accompanying me throughout these four years.

Thank you for always believing in me, even when I doubted myself, and for being the best example I could ever have, not only in the professional field but, above all, in the personal one.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Rail transport has been, since its beginnings, a key factor in economic and social development, promoting the growth of the industrial sector by making it easier to move goods and people from one place to another. However, its usage is still far from reaching its full potential, accounting for around 7% of all passenger transport and approximately 18% of goods transport in the European Unión, according to Eurostat and the European Commission [37].

Unlike other transport systems, the train has a significantly lower environmental footprint: data from the European Environment Agency dated back to 2018 [20], show that it emits only 33 g CO_2 per passenger-kilometer, compared to 143 g for cars and 160 g for planes. In freight, the advantage is even greater: the rail generates roughly 24 g of CO_2 compared to 1036 g for air cargo.

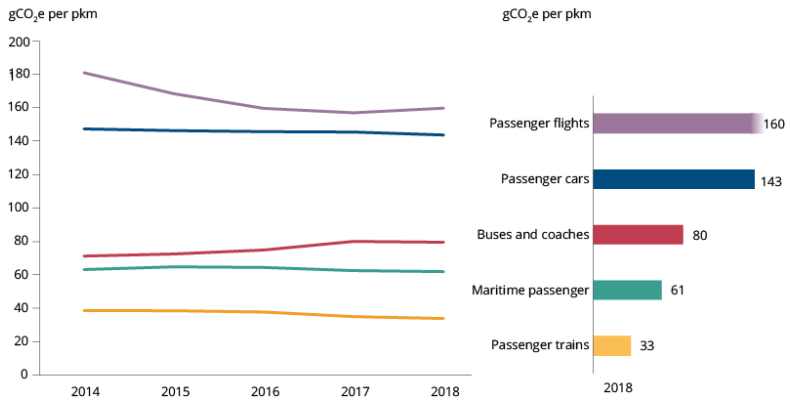


Figure 1.1: Average GHG emissions by motorised mode of passenger transport, EU-27, 2014-2018

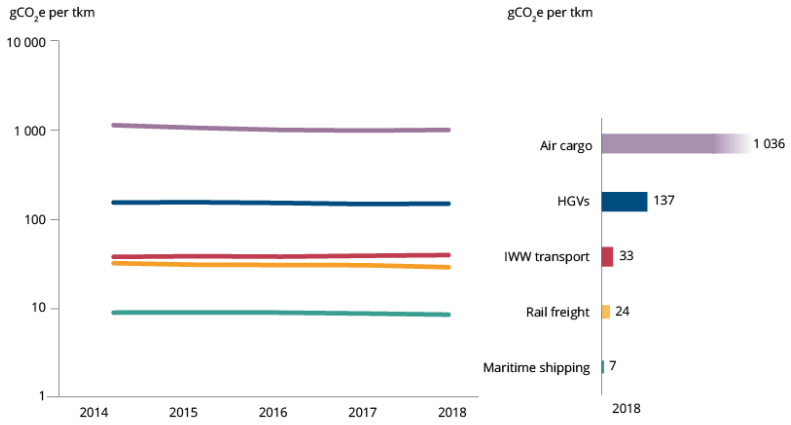


Figure 1.2: Average GHG emissions by motorised mode of freight transport, EU-27, 2014-2018

Consequently, the train has been established as one of the key factors in achieving climate neutrality, according to the European transport policy, doubling the traffic of high-speed trains by 2030 and triple it by 2050[50]. In order to be able to triple the traffic, it is essential to maintain rail safety and reliability,which are highly dependent on the quality of its infrastructure and the regularity of inspections, performed on the thousands of kilometers of Europe's railroads. Failures in the tracks, switches, poles and cables, can result in high economic losses, operational disruptions, and, in the most severe cases, deadly accidents.

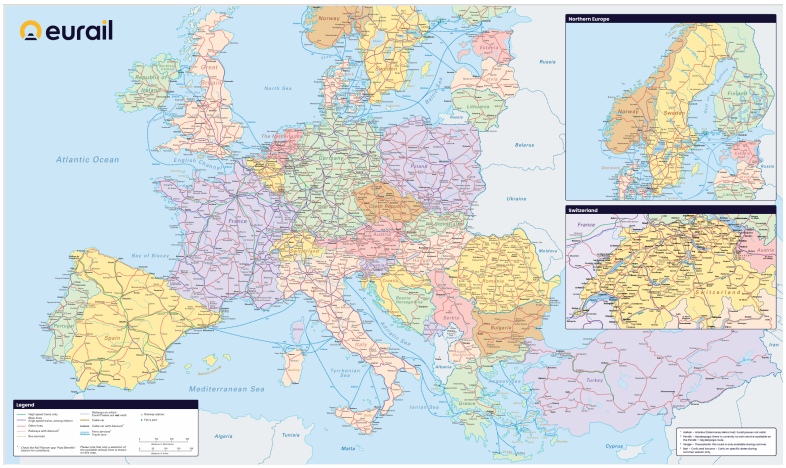


Figure 1.3: Eurail network map of Europe

For decades, traditional inspections such as visual assesments done by trained staff and inspection trains, have been used to monitor the condition of the railway infrastructure. While these methods have been proven reliable, they require significant resources and highly trained professionals. Recent innovations such as non-destructive testing (NDT), drone-based aerial surveys, high-resolution LiDAR mapping, and computer vision systems make it posible to reduce the dependance on manual field inspections, increase the frequency of inspections and improve the early detection of defects by gathering precise and high-quality data [8].

Additionaly, the latest advances in predictive manintenance are transforming how railway infrastructure is inspected, combining real-time sensor networks with 3D point cloud analysis and machine learning algorithms to detect faults that require further inspection, before they become more severe. Studies [26, 60, 86] have shown that integrating these technologies can enable a continuous, data-driven analysis that improves the accuracy and speed of defect detection. Their use also contribute to cost reduction by prioritising inspections where they are most needed.

1.1 Motivation for the study

The main motivation of this project is to address the current challenges arising from the application of Artificial Intelligence techniques to the railway sector, a field that is expected to undergo a big digital transformation in the future,

especially considering the European Union's goal of tripling rail traffic by the year 2050 [50]. This goal needs to be achieved not only with the expansion of railway infrastructure, but also with the introduction of advanced technologies that optimize the inspections of the already installed infrastructure.

In this context, research work becomes crucial to automatize and digitalize how and when inspections take place. This project aims to create automatic models that require minimal human intervention, making use of deep learning algorithms on multidimensional point cloud data, adapting to datasets collected from different countries and different surface types.

In addition, this project has a big ethical and social component, contributing to several of the Sustainable Development Goals (SDGs) established by the United Nations (UN) (see Appendix A) in 2015 as target goals for the year 2030:

- **Industry, Innovation and Infrastructure (9).** Addressed in this project by promoting the use of artificial intelligence to optimize inspection, maintenance, and repairing tasks in critical infrastructure. The main goal is to reduce reliance on manual interventions while advancing the digitalization of this industry. This vision is in line with initiatives led by the European Union Agency for Railways (ERA) and the Europe's Rail Joint Undertaking [17], which encourage the development of more resilient, intelligent, and sustainable infrastructures, as outlined in the EU's common transport policy.
- **Reduced inequalities (10).** This project develops tools that can be adopted by countries regardless of their level of technological resources. Because the methodology is based on reproducible models and open-access datasets, it can be adapted to different national contexts, from highly industrialized railway networks to less developed regional systems. This promotes equal access to advanced digital solutions, reducing dependence on costly technologies and enabling wider adoption, contributing to narrow the technological gap within and among countries, while fostering more inclusive and sustainable mobility on a global scale.
- **Sustainable cities and communities (11).** Promoting rail transport as an efficient, safe, and low-emission type of mobility directly supports urban sustainability. Strengthening the role of railways helps reduce dependence on private cars, alleviate congestion in cities, and lowers both noise and air pollution levels. Beyond these immediate benefits, it also contributes to encouraging more integrated and sustainable urban models.

- **Climate Action (13).** Sustainability is closely connected to the railway sector, as trains are one of the most sustainable modes of transport available today. Compared to road and air travel, the train has significantly lower greenhouse gas emissions, consumes less energy per passenger or ton of goods transported. This project reinforces the long-term sustainability of rail transport, helping it to remain a safe, efficient, and environmentally friendly alternative.

- **Partnerships for the goals (17).** This project is framed within the context of international cooperation to build a common, interoperable, and digital railway network. This goal is to create the Trans-European Transport Network (TEN-T)[21], which aims to connect rail systems across European Union's member countries and to ensure seamless cross-border mobility. European transport policy [17] emphasizes the importance of interoperability, standardization, and shared digital platforms so that national networks can operate as part of a single system. In doing so, it reinforces the idea that the modernization of the railway sector is a collective effort, where shared knowledge, common standards, and joint strategies are essential to achieving sustainable, efficient, and resilient transport across Europe.

1.1.1 Main objectives

- **Development of models capable of processing 3D point clouds.** A central objective of this project is the design of models able to process 3D point cloud data, which is the format in which LiDAR captures information about the railway environment. Building robust tools to handle this type of data is essential, as point clouds provide a highly detailed and accurate digital representation of infrastructure elements, enabling advanced analysis and automation.

- **Automatic classification of railway infrastructure elements** Another key goal is to achieve the automatic classification of different components of the railway infrastructure, such as tracks, masts, vegetation, and signaling devices. The purpose is to create a pipeline in which the model learns to distinguish these elements without manual intervention, ensuring that the workflow extends seamlessly from the acquisition of raw point cloud data to the final analysis of anomalies.

- **Automatic anomaly detection in railway infrastructure** This project also seeks to contribute to the automation of failure and anomaly detec-

tion in railway networks. By reducing reliance on manual inspections, the proposed methods support the development of safer and more efficient infrastructures. This objective is strongly aligned with European initiatives that aim to modernize railway management by lowering maintenance costs and increasing reliability through digital technologies.

- **Creation of a foundation for predictive maintenance applications** Finally, the project is intended to lay the groundwork for future predictive maintenance systems. The models developed here could be integrated into decision-support tools capable of issuing early warnings about potential infrastructure problems. Establishing this foundation is an important step toward more proactive and sustainable asset management, ultimately contributing to the resilience and long-term sustainability of railway transport.

1.1.2 Secondary objectives

- **Comparison of different classification methodologies** A complementary objective of this project is to compare different classification approaches, ranging from traditional machine learning methods to advanced deep learning techniques. This comparison allows for a better understanding of the strengths and limitations of each methodology when applied to diverse railway environments.

- **Evaluation of the generalization capability of the models** Another important secondary objective is to assess the ability of the developed models to generalize beyond the specific environments in which they were trained. This involves testing whether models trained with point cloud data from one country or type of railway system can also perform effectively in other regions with different infrastructure characteristics. Such evaluation is crucial to determine the robustness and transferability of the proposed methods, paving the way for their adoption in a wider range of real-world applications.

1.2 Methodology

This project develops a complete workflow for the automatic analysis of railway infrastructure based on 3D point cloud data. The methodology follows four main phases: first, an exploration of the raw dataset to understand its structure and

quality; second, a processing stage where the data is normalized, filtered, resampled, and split to create balanced subsets; third, the modelling phase, which includes a classification, segmentation, and anomaly detection tasks using both traditional machine learning and deep learning techniques; and finally, a testing and visualization stage, where the results are evaluated and interpreted.

STAGE 1: DATA EXPLORATION

STAGE 2: DATA PROCESSING

STAGE 3: MODELLING

STAGE 4: TESTING AND RESULTS

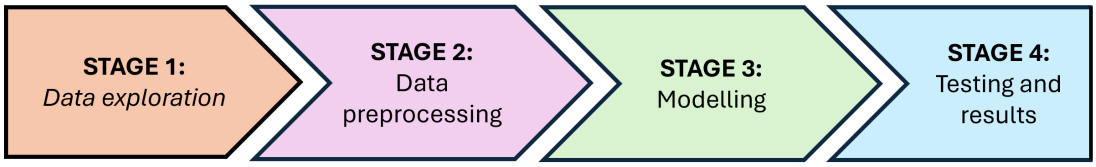


Figure 1.4: Project workflow

STAGE 1: Data exploration

The first phase consists of an exploratory analysis of the raw LiDAR dataset. This stage is very important because raw point cloud data usually presents a high level of complexity: it contains millions of points with diverse attributes, irregular densities, and heterogeneous distributions depending on the scene. The main objective of this phase is to detect irregularities that may affect the processing or modelling stages. Moreover, the EDA allows us to evaluate class balance across the infrastructure categories, a key factor for avoiding biased models in subsequent training.

The output of this phase is a comprehensive Exploratory Data Analysis (EDA) report that includes summary statistics, graphical visualizations, and descriptive insights.

STAGE 2: Data processing

The second phase applies a transformation to the raw data which is crucial to

guarantee that the models trained in the subsequent phase are reliable and robust.

Data normalization, ensures that all numerical features are scaled to a consistent range, preventing certain variables with naturally larger ranges, such as coordinates to dominate. Next, the data split divides the dataset into three subsets: train, validation, and test. The training subset is used to fit the parameters of the models, the validation subset allows for tuning hyperparameters and preventing overfitting, and the testing subset provides a final and independent evaluation of model performance. In parallel, data resampling techniques are applied to correct imbalances in the dataset, ensuring that all classes are represented in a balanced way, which improves the ability of the models to correctly recognize minority infrastructure elements. Finally, the data filtering stage removes irrelevant points that do not belong to the target classes of interest. Since the focus of this project is on railway infrastructure, points outside the desired categories (cables, poles, and rails) are excluded.

The result of this processing phase is a collection of normalized, balanced, and filtered datasets, organized into train, validation, and test subsets.

STAGE 3: Modelling

The modelling phase represents the core of this project, as it integrates all the previous steps into a series of predictive tasks that enable the automatic classification and analysis of railway infrastructure. This stage is divided into three consecutive subphases, each addressing a specific task in the interpretation of the point cloud data.

The first subphase, railway presence classification, simplifies the original multiclass dataset into a binary problem: infrastructure points (classes corresponding to poles, rails, and cables) versus non-infrastructure points (such as vegetation, ground, or other objects).

The second subphase, railway components segmentation, separates the different elements of the infrastructure. At this stage, a custom deep learning model based on the PointNet architecture is employed. This model classifies each point into one of three categories: rails, poles, or cables.

The final subphase, cable anomalies detection, focuses exclusively on the points classified as cables in the previous step. Cables play a critical role in railway systems, as they are responsible for power transmission and signaling, and their failure

can lead to significant disruptions in railway operation. For this reason, a dedicated anomaly detection system is developed to identify five types of anomalies.

From the initial binary filtering to the anomaly detection system, each step is designed to build upon the previous one. The outcome is a structured and automated methodology capable of detecting not only the presence of railway infrastructure but also its detailed components and potential defects.

STAGE 4: Testing and results

The final phase focuses on testing, visualization, and interpretation of results. The trained models are applied to validation and test subsets, generating color-coded 3D visualizations. These visual outputs highlight infrastructure elements classified as normal or anomalous. This stage provides not only a quantitative evaluation of the models but also an intuitive and visual representation of the anomalies detected, supporting decision-making in railway maintenance workflows.

Chapter 2

State of art

2.1 Evolution of railway transport

The origins of rail transport [51] date back much earlier than commonly assumed. Since ancient times, humans have sought ways to make the movement of heavy loads easier, and one of the earliest examples of guided transport can be traced to the Diolkos in Ancient Greece during the 6th century BC. This construction consisted of a stone-paved track with carved grooves that enabled carts to carry ships over land. Although it was far from what we now recognize as a railway, it already introduced the fundamental concept of vehicles following a fixed path, which is one of the key principles of the modern railway systems.

Later on, during the 18th century, the real foundations of modern railways were established with the invention of the steam engine by James Watt between 1763 and 1775. At the beginning, rail-like systems were mainly used in the mining industry, where wagons carrying minerals were placed on wooden rails to reduce friction and make transportation easier. This was a very simple idea, but it showed how combining rails with mechanical power could completely revolutionize the way big loads were transported from one place to another reducing the amount of effort needed.

The first major breakthrough took place in 1802, when Richard Trevithick built the very first prototype of a steam-powered locomotive. Although the machine was still limited and mainly experimental, it demonstrated that steam engines could successfully move vehicles along rails. A little more than twenty

years later, in 1825, George Stephenson inaugurated the world's first commercial railway line, connecting Stockton and Darlington in England. This milestone was a turning point, proving that railways were not only technically possible but also a practical solution for transporting goods across the country. Just five years later, in 1830, Stephenson expanded this vision with the Liverpool–Manchester line, the first railway specifically designed to carry both passengers and freight, opening the door to the modern railway era.

Figure 2.1: A steam locomotive from the mid-20th century.

After its success in the United Kingdom, railways quickly spread across Europe. Belgium and Germany built their first lines in 1835, France followed in 1837, Spain in 1848 with the Mataró–Barcelona route, and Italy in 1861. This expansion was a clear sign that railways were becoming one of the most important technological innovations of the 19th century, fundamentally transforming transport and commerce. By the end of that century, steam locomotives were already being replaced by electric ones, which were cleaner and more efficient.

The 20th century brought another revolution with the development of high-speed trains [62]. Japan was the pioneer in this field with the launch of the Shinkansen in 1964, capable of reaching speeds of 300 kilometers per hour. This idea quickly inspired other countries: France introduced the TGV (Train à Grande Vitesse) in 1981, and Spain followed with the AVE (Alta Velocidad Española) in 1992, connecting Madrid and Seville just in time for the Universal Exposition. High-speed trains completely changed the way people traveled across countries, making these trains a real competitor to road and even air transport.

Figure 2.2: The inauguration of the first Shinkansen high-speed train in Japan (1964)

Today, Europe has become a world leader in railway transport, with extensive national and international networks that offer strong connectivity across the continent. High-speed rail has been a key factor in this development, positioning Europe as a leader in sustainable and efficient mobility. Spain, for example, stands

out as the country with the second largest high-speed rail network in the world, covering 4,327 kilometers, only behind China. France follows closely in third place, with Germany and Italy also ranking among the countries with the most developed high-speed infrastructures.

2.2 Railway infrastructure inspections

The railway infrastructure (see Figure 2.3 [61]) is formed by several interconnected subsystems, each of which plays an important role in guaranteeing operational safety and efficiency. The track system, made up of rails, sleepers, ballast, and fastening elements, provides the physical guidance for trains, but, it is subject to deterioration mechanisms such as rail head wear, rolling contact fatigue, cracks, and misalignment caused by big loads and environmental factors. The supporting structures, such as poles and masts, ensure the mechanical stability of the electrification system, but over time they may experience corrosion, loss of verticality, or structural weakening due to material fatigue.

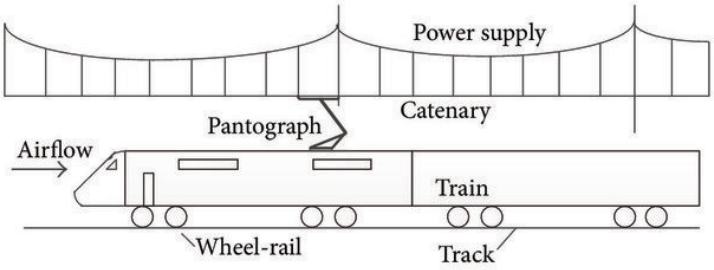


Figure 2.3: Schematic representation of the main components of an electrified railway system

A particularly sensitive component the catenary system, which is the overhead contact line (OCL), supplies electrical energy to the trains through continuous contact with the pantograph. Failures in this subsystem [31] can have a very severe impact, as they can lead to service disruptions or even accidents. The most common defects include loss of tension, wire wear at the pantograph contact points, mechanical oscillations, displacement or sagging of the contact wire, and, in the most extreme cases, wire breakage. For this reason, inspection protocols place special emphasis on the verification of catenary geometry (height and stagger), measurement of contact wire wear, and detection of anomalies in the mechanical tension systems.

Incidents like the one in the Netherlands in September of 2023 [36], where a freight train caught fire after a defective temporary repair on the overhead line, demonstrates the risks of inadequate infrastructure interventions. More recently, in June of 2025 [64], Eurostar services between France and the United Kingdom were severely disrupted following the theft of power supply cables in northern France, demonstrating how both technical failures and external factors can have a very big impact in the maintenance of the infrastructure. Shortly after, on June 30, a major breakdown occurred on the Spanish high-speed network [68] between Madrid and Andalucia, when a contact wire melted due to an electrical overload caused by multiple trains accumulating on the line. This incident led to the suspension of AVE and Avant services, leaving hundreds of passengers overnight and emphasizing the vulnerability of high-speed networks to failures in their catenary systems

2.2.1 Visual inspection and specialized equipment

Visual inspections[22] remain one of the most traditional and widely applied methods for assessing the condition of railway infrastructure. These inspections are generally carried out either on foot or using hi-rail vehicles.

- **On-foot inspections.** Specialized personnel walk along the track to identify surface defects such as cracks in the rail head, misalignments, deteriorated sleepers, or loose fastenings. It usually requires one or two inspectors and is highly labor-intensive, which increases its operational costs. The frequency of such inspections depends on the type of line and its usage: high-speed and heavily trafficked lines may require weekly examinations, whereas secondary lines are inspected monthly.
- **Hi-rail vehicles.** More efficient alternative, consisting on road vehicles adapted with retractable rail wheels that allow inspectors to travel along the tracks while conducting visual assessments. This reduces the physical demands of walking long stretches of railway and enables faster coverage of larger distances, although it still requires one or two trained operators. The costs of hi-rail inspections are lower than those of on-foot inspections in terms of labor, but they continue to depend on track access schedules and the availability of specialized vehicles, and are usually conducted weekly.

Specialized inspection trains [52] represent a more advanced alternative to manual visual inspections, providing continuous and automated monitoring of railway

infrastructure across long distances, and can be divided into two main categories depending on their function:

- **Track geometry measurement trains.** Dedicated to the evaluation of rail and track geometry parameters such as alignment, gauge, curvature, and surface wear. These trains are typically equipped with laser sensors, inertial systems, and high-speed cameras that allow precise detection of deviations that could compromise safety. The frequency of these inspections depends on the traffic of the tracks, going from biweekly inspections in high-speed lines to annual in low-used rails.
- **Overhead line monitoring trains.** Specifically designed to inspect the state of catenaries and electrical supply systems. Using thermal cameras, laser scanners, and high-resolution video equipment, they are able to identify problems such as irregular contact wire height, excessive inclination, or wear in the pantograph–catenary interface. The inspections are usually conducted every four weeks, but could have a higher or lower frequency depending on the traffic of the rails inspected.

Both types of trains cover long distances in a single run, which makes them highly efficient for national and international networks. However, they require a very high investment to produce and use them, additionally needing teams of specialized engineers and technicians to operate the onboard instruments and interpret the data collected.

2.2.2 Non-destructive testing (NDT) methods

Non-Destructive Testing (NDT) [2] methods represent one of the most reliable approaches for ensuring the safety and durability of railway infrastructure, as they allow the early detection of defects without causing any physical damage to rails or components.

- **Ultrasonic Testing (UT).** This is the most widely applied in railway inspection and relies on the transmission of high-frequency sound waves into the rail material to identify internal flaws. A transducer emits ultrasonic pulses that travel through the steel, and any discontinuities, such as cracks, voids, or inclusions, cause reflections that are captured and analyzed by the device. The method is typically carried out at regular intervals ranging from

every few weeks to several months, depending on traffic density and rail category. This technique generally requires a team of two to three trained technicians and involves moderate costs associated with both equipment and specialized labor.

- **Eddy Current Testing (ET).** This method uses electromagnetic induction to detect surface and near-surface flaws in rails and metallic components. A probe induces alternating currents (eddy currents) in the conductive material, and any disruption caused by cracks, corrosion, or material loss modifies the flow, which is detected by the instrument. ET is highly effective for identifying head checks and surface defects that are often missed by ultrasonic techniques. Usually performed in shorter cycles and can also be deployed on inspection trains equipped with automated sensors, reducing the need for manual operators.

- **Electromagnetic Acoustic Transducers (EMAT).** This technique is a variation of ultrasonic testing that does not require direct contact with the material or the use of coupling gels, making it highly practical for railway applications. They generate ultrasonic waves within the material through electromagnetic induction, which makes them particularly effective for inspecting rails that are dirty, oxidized, or coated, where traditional ultrasonic probes lose efficiency.

METHOD	PERSONNEL	COST	FREQUENCY
On-Foot Visual Inspection	*1-2 inspectors*	*Medium–High*	*Weekly/Monthly*
Hi-Rail Vehicle	*1-2 inspectors*	*Medium*	*Weekly/Monthly*
Track Geometry Car	*3 operators*	*Very High*	*Annual/Biweekly*
Overhead Line Inspect Trains	*3 operators*	*Very High*	*Every 4 Weeks*
Ultrasonic Testing (UT)	*2-3 trained technicians*	*Moderate*	*Few weeks/Months*
Electromagnetic Testing (ET)	*2 technicians*	*Moderate*	*1-3 months*
Electromagnetic Acoustic Transducers (EMAT)	*2-3 skilled technicians*	*High*	*2-6 months*

Table 2.1: Combined overview of traditional and NDT inspection methods.

2.3 Technological innovations in railway monitoring

2.3.1 LiDAR and point cloud technologies

LiDAR technology and point cloud data has become one of the most advanced approaches for the inspection and digitalization of railway infrastructures [76], including airborne scanners, vehicle-mounted systems, and UAV-based platforms, and has been very important for the efficient acquisition of three-dimensional data in railway infrastructure, enabling applications that range from the creation of digital twins to high-precision inspections.

Building on this line of research, Dekker et al.[19] demonstrated how the integration of LiDAR with Mobile Laser Scanning (MLS) makes it easier to create a digital representation of railway assets, called digital twins, allowing both detailed modeling and near real-time monitoring, thus constituting a decisive step toward full digitalization of the sector. From an aerial perspective, Yarroudh et al. [80] showed that UAV platforms equipped with LiDAR can effectively monitor catenary systems in areas that are otherwise difficult to access, offering a safer and more ef-

ficient alternative to traditional methods. In addition, non-intrusive systems have been developed to measure track geometry irregularities (such as gauge, curvature, and profile)[4] using UAV-LiDAR platforms capable of operating without service disruption, achieving sub-inch accuracy.

In addition, techniques for the automatic extraction of overhead contact lines from MLS data have also been explored: Zhang et al. [29] successfully extracted support structures with an accuracy exceeding 97% by employing algorithms such as RANSAC and DBSCAN. These contributions, together with recent reviews on UAV-LiDAR applications for railway monitoring [78], illustrate the transition of LiDAR from its initial use in basic recognition and classification tasks to its integration into advanced solutions for inspection, digital modeling, and automated asset analysis, consolidating its role as a central technology in the modern management of railway infrastructure.

A recent study titled *A Railway LiDAR Point Cloud Reconstruction Based on Target Detection and Trajectory Filtering* [46] presents an advanced method for reconstructing railway point clouds based on mobile LiDAR. In this work, odometer information is corrected through the automatic identification of characteristic points in the railway environment using deep learning techniques, followed by trajectory optimization with a Rauch–Tung–Striebel (RTS) filter. Applied to experimental railway data, the method successfully constrained the maximum East–North coordinate difference to within 7 cm and achieved a mean altitude error of only 2.39 cm. This approach represents a big step forward, improving the geometric precision of the generated 3D models and enhancing the spatial fidelity of representations derived from mobile platforms.

2.3.2 Deep learning applications in railway inspection

The classification of railway infrastructure elements in LiDAR point clouds has changed a lot over the past decade, going from heuristic and geometric methods to advanced data-driven frameworks. In a pioneering study, Arastounia[5] introduced automated approaches for recognizing tracks, catenary and return wires, masts, and cantilevers using mobile mapping LiDAR data, achieving 100% precision and accuracy at the object level, with mean per-point accuracy and precision of 96.4% and 97.1%, respectively.Two years later, this author made refinements of the initial study[6], demonstrating >95% accuracy and precision even in complex urban environments.

Similarly, the study *Classification of railway assets in mobile mapping point*

clouds[16] advanced the automated processing of point clouds through Mobile Mapping by developing algorithms for the classification of railway assets directly from LiDAR data, thereby demonstrating the potential of these technologies for automated inventory generation. Building on this, Lamas et al. [41] designed a heuristic segmentation model for a 90 km dataset, successfully classifying rails, droppers, wiring, masts, and signals, and reporting F1-scores above 85% overall and greater than 99% for rails, confirming the scalability of such approaches.

Multi-Scale Hierarchical CRF for Railway Electrification Asset Classification from Mobile Laser Scanning Data[14] introduced a step forward in classification by proposing a multi-scale Hierarchical Conditional Random Field (HiCRF) model for railway electrification asset classification from Mobile Laser Scanning data, reaching an overall accuracy of 99.67%, which outperformed previously used local methods.

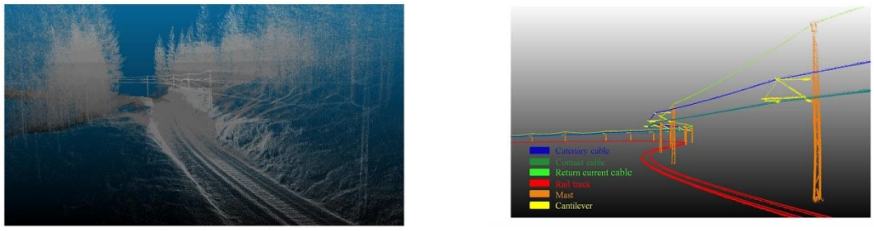


Figure 2.4: Classification of railway infrastructure from LiDAR data. *Automated recognition of railroad infrastructure in rural areas from LiDAR data. Remote Sensing*[6]

Since then, deep learning models have been used to classify the different parts of the infrastructure, like in the work of Grandío et al.[30], who applied PointNet++ and KPConv to semantic segmentation of complex railway environments, obtaining 90% accuracy, a mean IoU of 74.89%, and F1-scores above 90%. In a complementary study, Ton et al[67]compared PointNet++, SuperPoint Graph, and Point Transformer for the semantic segmentation of catenary arches using terrestrial laser scans, where PointNet++ achieved the best results with an IoU exceeding 71%.

More recently, in 2024, *UAV-Based LiDAR and Semantic Segmentation for Railway Catenary Infrastructure Reconstruction*[81] combined KPConv-based semantic segmentation with parametric modeling for reconstructing elements such as rails, wires, and poles, achieving a mean IoU of 84%.

Benchmarking efforts culminated in the introduction of the Rail3D dataset by Kharroubi et al[39], which comprises 288 million annotated points across nine

classes gathered in Hungary, France, and Belgium. Their evaluation demonstrated that KPConv achieved approximately an 86% mIoU, while LightGBM reached nearly a 71%, demonstrating the superior performance of deep learning architectures and the value of multi-context datasets for generalizable segmentation. Complementing these works, *Learning Behaviour of SPVConv Models for Semantic Segmentation of Railway Infrastructure*[70] analyzed the learning behavior of SPVConv-based models, providing insights into their scalability when trained on limited data.

In addition, Yu et al.[82] introduced a voxel-based, multi-scale neural network for real-time rail recognition, effectively handling straight, curved, and complex geometries, and the *Active Learning for Semantic Segmentation of Railway Point Clouds*[3] study applied active learning techniques and reported a mean IoU of 71.48% across nine classes, confirming the feasibility of scalable and automated semantic segmentation frameworks for railway infrastructure.

Taken together, these studies demonstrate a clear evolution from early heuristic and probabilistic approaches, such as HiCRF, to deep learning-based architectures, consistently improving performance metrics and consolidating semantic segmentation as a possible technology for the accurate and efficient management of modern railway infrastructure.

2.3.3 Anomaly detection in catenaries systems

Research on anomaly detection in railway infrastructure can be done in several parts of the infrastructure, each with distinct sensing and algorithmic approaches. For example, Ghiasi et al.[28] explored in their study *Unsupervised anomaly detection in railway track geometry using One-Class SVMs* unsupervised methods who applied One-Class SVMs to vibration data collected on SNCF networks, improving defect detection accuracy by 12% compared with raw accelerometer signals, and outperforming baselines like Isolation Forest and LOF.

Expanding this line, Cao et al.[11] introduced a meta-learning framework combined with GANs to detect anomalies in high-speed railway inspections, reporting a 99.7% reduction in inspection effort and 96.7% reduction in inspection time. UAV-based measurement has also been applied[59], with Qiu et al. (UNLV) developing a multi-rotor platform integrating LiDAR and vision for real-time track geometry monitoring, achieving high accuracy in gauge and curvature without service disruption.

In catenary systems, deep learning and sensing methods dominate. Chen et al.[13] employed convolutional neural networks to detect fastener defects on support devices, achieving high detection accuracy across large image datasets. *Inspection of railway catenary systems using machine learning with domain knowledge integration*[84] further advanced this by integrating domain knowledge into machine learning models, improving interpretability and defect classification, and Cao et al.[10] created a systematic mapping reviewing recent sensing and monitoring approaches for catenary condition assessment.

Complementary to this, Zhou et al.[87] proposed a vibration-based defect diagnosis method for rigid catenary systems, analyzing pantograph response signals to detect issues such as wear and poor contact with high reliability. Unsupervised strategies have also been used for catenary anomalies detection like in *Defect Diagnosis of Rigid Catenary System Based on Pantograph Vibration Performance Actuators* where the authors Wu et al.[77] developed an anomaly detection system for split pins in catenary assemblies using clustering-based methods without extensive labeled data. Beyond tracks and catenaries, anomaly detection has extended

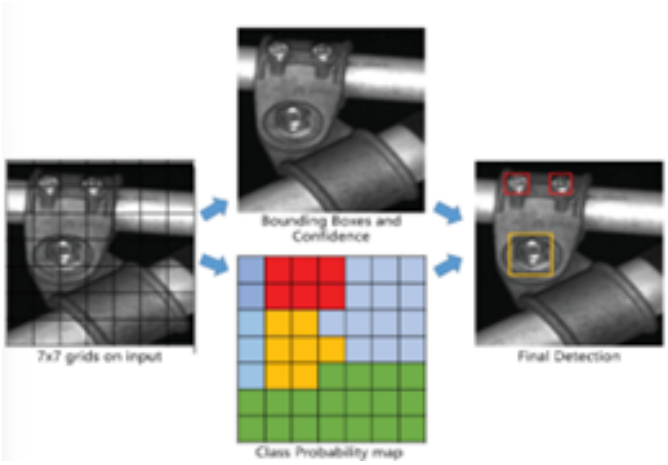


Figure 2.5: Catenary anomaly detection. *Defect Diagnosis of Rigid Catenary System Based on Pantograph Vibration Performance Actuators*[77]

to signaling cables, where Wang et al.[72] proposed a measurement-based monitoring framework to detect early degradation of signaling cables through electrical and operational parameter analysis. Meanwhile, computer vision continues to expand the scope of inspection: a systematic review in *A systematic literature review of defect detection in railways using machine vision-based inspection methods*[27] established that deep learning methods consistently outperform traditional rule-based image analysis for detecting defects in rails, sleepers, and fasteners, although

progress is constrained by the limited availability of annotated datasets.

Finally, advances in multi-modal and unsupervised methods broaden anomaly detection. *DHT-CL: Multi-modal Contrastive Learning for Obstacle Detection in Railways under Adverse Weather Electronics*[75] introduced the DHT-CL model, combining LiDAR and camera data via multi-modal contrastive learning for obstacle detection under adverse weather, achieving mIoU of 87.38%.

Self-supervised strategies have also been tasted: Li et al.[44] proposed MAESTRO, a reconstruction-only anomaly detection framework for 3D point clouds that outperformed previous benchmarks in industrial settings, while Li[43] released Anomaly-ShapeNet and introduced IMRNet, reaching 66.1% I-AUC and 72.5% on Real3D-AD benchmarks, confirming the value of large-scale synthetic datasets for anomaly detection research.

2.4 Datasets for railway infrastructure

A growing number of datasets have been developed to support research in railway infrastructure monitoring and semantic segmentation, each addressing specific components of the system. The RailSem19 dataset[83] provides over 8,500 annotated sequences of railway and tramway scenes from a train's perspective, enabling semantic understanding of rail environments including crossings and urban interactions.

Focusing on railway signaling, the FRSign dataset[34] compiles more than 100,000 annotated images of French railway traffic lights, capturing temporal, spatial, and sensor-related metadata to support signal recognition tasks. In terms of safety, the RAWPED dataset[48] introduces annotated data of pedestrians in railway environments, while a complementary study by the same authors employed a proprietary dataset to evaluate ensemble methods for pedestrian detection in driver support systems. For signalization in different contexts, the GERALD dataset[42] focuses on German mainline railway signals, providing over 5,000 images annotated for signal detection.

In addition to 2D images, several high-resolution 3D datasets have been released. Ton[66] introduced a labeled terrestrial laser scanning dataset covering 15 catenary arches in the Netherlands, annotated into 14 distinct classes, which can be used as a reference benchmark for overhead line segmentation. Expanding in scale, the WHU-Railway3D dataset[58] offers over 4 billion points covering

30 km of railway infrastructure across rural, urban, and plateau environments in China, divided into 11 annotated classes, and has become one of the largest and most diverse benchmarks for point cloud segmentation. Similarly, the OSDaR23

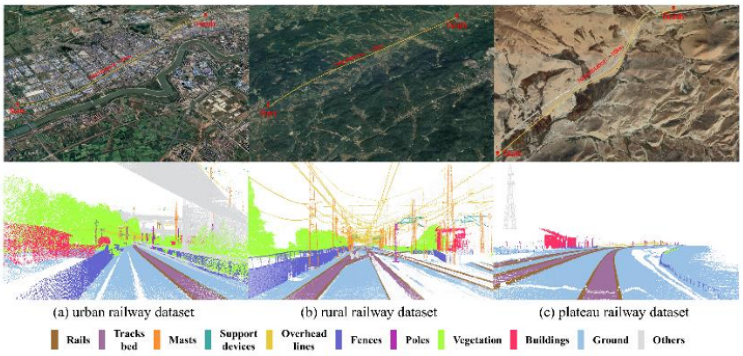


Figure 2.6: WHU Dataset.[58]

dataset[65] provides a multi-sensor dataset collected in Hamburg, Germany, combining LiDAR, RGB, infrared cameras, and radar, with annotations spanning 20 semantic classes including trains, catenary poles, and vegetation. For localization and mapping, Wang[74] released the Railway SLAM dataset, which consists of LiDAR and inertial data supporting the development of simultaneous localization and mapping algorithms for railway environments.

Finally, Kharroubi et al.[40] introduced the Rail3D dataset, a multi-context annotated point cloud dataset with 288 million points across nine universal classes (rails, poles, wires, vegetation, signals, fences, installations, buildings, and ground), collected in Hungary, France, and Belgium. This dataset represents a key step toward generalizable models, as it provides cross-context benchmarks for semantic segmentation under varying acquisition conditions.

Collectively, these datasets constitute the foundation for advancing perception and monitoring in railway research, offering a range of resources from annotated 2D images to large-scale 3D point clouds that enable both semantic understanding and the development of deep learning models for railway infrastructure management.

Study	Elements Classified	Model Used	Performance Metrics
Ton et al. (2023) *Semantic Segmentation of Terrestrial Laser Scans of Railway Catenary Arches*	*14 catenary arch components*	*PointNet++, SuperPoint Graph, Point Transformer*	*PointNet++: IoU (as reported)*
Yarroudh et al. (2024) *UAV–Based LiDAR for Railway Catenary Reconstruction*	*Rails, wires, poles*	*KPConv + parametric modeling*	*mIoU = 84%*
Kharroubi et al. (2024) *Rail3D Dataset and Benchmarks*	*9 universal classes*	*KPConv, LightGBM, Random Forest*	*KPConv: ~86% mIoU; LightGBM: lower mIoU*
Werf (2023) *Learning Behaviour of SPVConv Models*	*Multiple railway asset classes*	*SPVConv*	*Scalability under limited data*
Yu et al. (2022) *Real-Time Rail Recognition*	*Rails*	*Voxel-based, multi-scale neural network*	*Robust detection across diverse geometries*
arXiv (2024) *Active Learning for Semantic Segmentation of Railway Point Clouds*	*9 semantic railway classes*	*Active learning-based 3D segmentation*	*mIoU = 71.48%*
Arastounia (2015) *Automated Recognition of Railroad Infrastructure in Rural Areas from LiDAR Data*	*Tracks, catenary wires, return wires, masts, cantilevers*	*Rule-based segmentation with geometric & spatial relations (KD-tree, FLANN)*	*100% object-level accuracy & precision; 96.4% point-level accuracy*

Table 2.2: State of the art studies for infrastructure classification (part I).

Study	**Elements Classified**	**Model Used**	**Performance Metrics**
Corongiu et al. (2020) *Classification of Railway Assets in Mobile Mapping Point Clouds*	*Rails, poles, cables, signals*	*Handcrafted geometric descriptors + classifiers*	*Reliable asset classification (no exact % reported)*
Lamas et al. (2021) *Automatic Point Cloud Semantic Segmentation of Complex Railway Environments*	*Rails, droppers, wiring, masts, signals*	*Heuristic pipeline segmentation*	*F1 > 85% overall; >99% for rails*
Chen et al. (2019) *Multi-Scale Hierarchical CRF for Railway Electrification Asset Classification*	*Electrification assets (10 classes)*	*Multi-scale Hierarchical CRFs (HiCRF)*	*99.67% overall accuracy*
Grandío et al. (2022) *Point Cloud Semantic Segmentation of Complex Railway Environments Using Deep Learning*	*Rails, signals, poles, vegetation, lighting, etc.*	*PointNet++ and KPConv*	*Accuracy = 90%; mIoU = 74.89%; F1 ~ 90%*

Table 2.3: State of the art studies for infrastructure classification (part II).

Chapter 3

Theoretical foundations

3.1 Traditional Machine Learning techniques

3.1.1 XGBoost

The **XGBoost algorithm**, short for **Extreme Gradient Boosting**, was introduced in the study *XGBoost: A Scalable Tree Boosting System* by Tianqi Chen and Carlos Guestrin in 2016[15]. Its name is a reflection of both, the family of boosting algorithms and the fact that it is an optimized and extended version of gradient boosting.

Boosting

Boosting[23] is an ensemble learning technique. Its fundamental principle lies in sequential training: each new model is fitted to correct the errors of its predecessors, so that misclassified observations receive greater weight in subsequent iterations. Boosting focuses on reducing bias by progressively correcting mistakes.

Gradient boosting

Friedman[24] and Mason et al.[49] extended this framework by introducing gradient boosting. Instead of merely reweighting or reclassifying mispredicted examples, each new model $f_t(x)$ is trained to approximate the negative gradient of a differentiable loss function. The resulting model takes an additive form:

$$\hat{F}_t(x) = \hat{F}_{t-1}(x) + \nu f_t(x)$$

ν denotes the learning rate. In this way, boosting becomes a flexible method that can adapt to a variety of tasks simply by choosing the appropriate loss function.

XGBoost

Although XGBoost is grounded in the principles of gradient boosting, it introduces several innovations that distinguish it from traditional boosting frameworks and standard gradient boosting implementations. The most relevant properties[15] include:

1 **Explicit regularization.** Incorporates both L1 and L2 penalties in the objective function, directly controlling tree complexity and reducing overfitting.

2 **Second-order optimization.** Uses not only gradients but also Hessians (second derivatives), improving convergence speed and stability.

3 **Histogram-based split finding.** Accelerates the process of identifying optimal split points in trees, reducing computational cost.

4 **Cache-aware and parallel computation.** Employs memory-efficient strategies and supports parallelization, enabling fast training even on large-scale datasets.

5 **Distributed learning.** Provides support for training on distributed systems, making the algorithm highly scalable.

6 **Sparsity-aware handling of missing data.** Automatically learns default split directions for missing values, eliminating explicit imputation.

7 **Shrinkage (learning rate scaling).** Reduces the contribution of each tree, improving generalization and preventing overfitting.

8 **Row and column subsampling.** Randomly samples data and features, further enhancing robustness and reducing variance.

MATHEMATICAL FORMULATION

XGBoost is based on an objective function[15], which defines the optimization problem to be solved during training:

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

This objective function consists of two components. The first term: $\sum_{i=1}^{n} l(y_i, \hat{y}_i)$, corresponds to the loss function, which measures the discrepancy between the predicted values $haty_i$ and the true labels y_i. Depending on the task, different loss functions can be employed, such as the mean squared error for regression or the logistic loss for binary classification.

The second term: $\sum_{k=1}^{K} \Omega(f_k)$ represents the regularization component, which penalizes the complexity of the individual trees. This penalty is determined by factors such as the number of leaves or the magnitude of the leaf weights, ensuring that the model does not become unnecessarily complex.

The objective function is used as the guiding criterion for optimization. At each boosting iteration, a new tree is added only if it contributes to minimizing this function. Trees that explain the data very well but are overly complex receive a penalty, while simpler trees that achieve a good trade-off between accuracy and complexity are favored.

TREE CONSTRUCTION

XGBoost trees are specifically designed to reduce the residual errors of the previous ones.

The split criterion quantifies the improvement in the objective function that would result from dividing a node into two branches, left L and right R:

$$Gain = \frac{1}{2}\left(\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + \lambda}\right) - \gamma \tag{3.1}$$

Here, G_L and G_R represent the sums of the gradients of the loss function for the samples assigned to the left and right branches, while H_L and H_R are the corresponding sums of the Hessians (second derivatives). The parameter λ provides L2 regularization, and γ imposes a penalty for introducing a new leaf. A split is only performed if the Gain is positive, avoiding unnecessary splits.

The quality of the entire tree is measured using the tree structure score, defined

as:

$$\mathcal{L}_{tree} = -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j+\lambda} + \gamma T \tag{3.2}$$

where T denotes the total number of leaves, G_j and H_j are the sums of gradients and Hessians for the samples in leaf j, and λ and γ are the regularization parameters. A higher score indicates a better trade-off between accuracy and complexity: the first term analyzes accurate splits, while the second penalizes trees with excessive leaves.

REGULARIZATION AND PRUNING

A distinctive feature of XGBoost compared to earlier boosting implementations is the integration of an explicit regularization framework that directly influences the growth and pruning of decision trees to prevent overfitting and to improve generalization.

The regularization term penalizes both the number of leaves and the magnitude of their weights:

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2 \tag{3.3}$$

where T is the number of leaves in the tree, w_j represents the weight assigned to the leaf j, γ is a complexity penalty for each additional leaf, and λ is the L2 regularization parameter. This formulation ensures that trees do not grow excessively deep and that leaf predictions remain bounded, promoting simpler and more interpretable structures.

Building upon this, XGBoost determines the optimal weight for each leaf by minimizing the regularized objective. The optimal value is obtained in closed form as:

$$w_j^* = -\frac{G_j}{H_j+\lambda} \tag{3.4}$$

where $G_j = \sum_{i\in I_j} g_i, \quad H_j = \sum_{i\in I_j} h_i$ denote the sum of gradients and Hessians for the samples in leaf j respectively. Nodes are split only if the gain in objective reduction exceeds the penalty γ; otherwise, the split is discarded. This means that tree growth is not limited by a predefined depth but by the trade-off between predictive improvement and structural complexity.

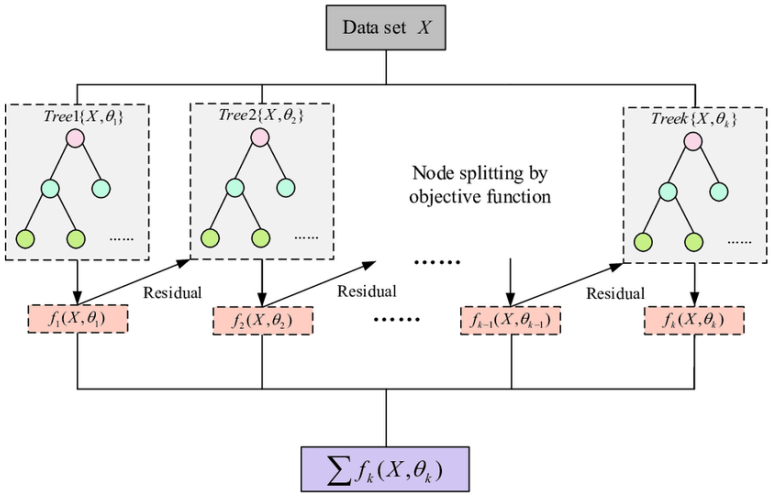


Figure 3.1: XGBoost training workflow

3.1.2 LightGBM

LightGBM, short for **Light Gradient Boosting Machine**, is an open-source framework introduced in *LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Advances in Neural Information Processing Systems* by Microsoft in 2017[38] as part of the gradient boosting family of algorithms. Like other models such as XGBoost, it builds ensembles of decision trees using the principle of gradient boosting, where trees are sequentially added to correct the residual errors of their predecessors.

This algorithm is particularly useful for handling large-scale datasets with high-dimensional features. Unlike traditional gradient boosting methods that grow trees level-wise, LightGBM employs a leaf-wise growth strategy, which splits the leaf with the largest loss reduction at each step.

The most relevant properties[38] include:

1 **Histogram-based split finding.** Continuous features are discretized into bins, reducing memory usage and accelerating split calculations without loss of accuracy.

2 **Leaf-wise growth strategy with depth constraints.** Trees grow by splitting the leaf with the largest potential loss reduction, often yielding

higher accuracy than level-wise growth, while depth constraints mitigate overfitting.

3 **Gradient-based One-Side Sampling (GOSS).** LightGBM maintains instances with large gradients and randomly samples those with smaller gradients, reducing training data size.

4 **Exclusive Feature Bundling (EFB).** High-dimensional sparse features that rarely take non-zero values simultaneously are bundled together, effectively reducing dimensionality.

5 **Efficient handling of categorical features.** The algorithm supports categorical variables natively by finding optimal split points without needing one-hot encoding.

6 **Parallel and distributed learning.** LightGBM can scale across CPUs and GPUs, and supports distributed training for very large datasets.

7 **GPU acceleration.** A dedicated GPU implementation speeds up histogram construction and split finding, making LightGBM particularly suitable for high-dimensional data.

8 **Optimization for sparse data.** The framework is designed to skip zero entries directly, making it efficient for tasks with sparse input matrices (e.g., text mining, click prediction).

9 **Early stopping.** Built-in mechanisms allow training to end once validation error stops improving, reducing computational cost and preventing overfitting.

10 **Compatibility with multiple loss functions.** LightGBM supports a wide range of differentiable loss functions (regression, classification, ranking, etc.) and allows users to define custom objectives.

MATHEMATICAL FORMULATION

LightGBM, as other gradient boosting frameworks, is based on an objective function[38], which combines the prediction loss with a regularization term to balance accuracy and model complexity. The general form is expressed as:

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

Here, $l(y_i, \hat{y}_i)$ represents a differentiable loss function that measures the discrepancy between the true label y_i and the predicted value $haty_i$, while $Omega(f_k)$ is a regularization term applied to each tree in the ensemble. This formulation allows LightGBM to adapt to a wide range of tasks—using squared error for regression, log-loss for classification, or pairwise ranking losses for learning-to-rank problems.

The regularization component ensures that model complexity is penalized, preventing overfitting and encouraging generalizable solutions. Each new tree added to the model is constructed with the objective of minimizing this function, thereby reducing prediction error while keeping the overall structure compact.

TREE CONSTRUCTION

The construction of decision trees in LightGBM is guided by the principle of functional gradient boosting. The optimization relies on the **second-order Taylor expansion of the loss function**, which incorporates both gradients and Hessians to approximate the objective more accurately and ensure stable convergence:

$$\mathcal{L}^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \tfrac{1}{2} h_i f_t(x_i)^2 \right] + \Omega(f_t)$$

Similarly, the **tree structure score** (see Equation 3.2) and the **split gain** (see Equation 3.1) formulas are equivalent to those in XGBoost, quantifying respectively the overall quality of the tree and the improvement obtained from each potential split. These ensure that splits are performed only when they reduce the objective, balancing predictive power and structural complexity.

Where LightGBM innovates is in the way it accelerates split finding and reduces the cost of tree construction. The first major contribution is the **Histogram-based approximation**. Instead of evaluating continuous feature values directly when searching for splits, LightGBM discretizes them into a finite number of bins:

$$x_i \;\rightarrow\; \text{bin}(x_i)$$

This transformation allows the algorithm to accumulate gradient and Hessian statistics per bin rather than per sample. The computational complexity is thereby reduced from $O(num_samplesxnum_features)$ to $O(num_binsxnum_features)$, accelerating training while maintaining nearly identical predictive performance, making LightGBM particularly effective for large datasets.

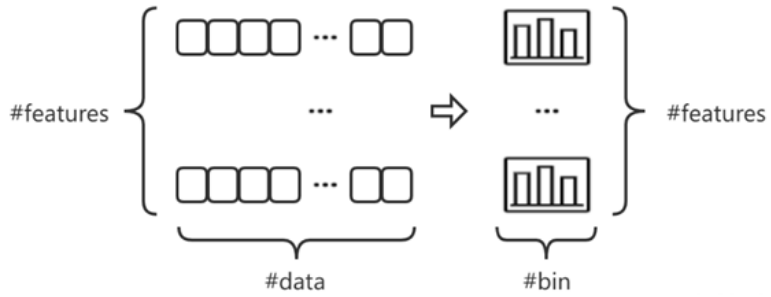


Figure 3.2: LightGBM histogram based approximation

The second key innovation is **Gradient-based One-Side Sampling (GOSS)**. The idea is that not all samples contribute equally to gradient-based optimization: instances with large gradients correspond to poorly predicted examples and thus contain more information for reducing loss. GOSS retains all instances with large gradients and only samples a proportion of those with smaller gradients:

$$\tilde{D} = D_{large} \cup \text{Sample}(D_{small},\, a)$$

where D_{large} are the high-gradient instances, D_{small} the low-gradient ones, and a is the sampling ratio applied. By emphasizing informative samples while discarding redundant ones, GOSS reduces the dataset's size considered during split evaluation without compromising accuracy. This innovation also accelerates training and reduces computational overhead.

Together, these two innovations improve the efficiency in tree construction, which is the most important feature for LightGBM, enabling this model to scale to much larger datasets then XGBoost, making it highly suitable for industrial applications.

REGULARIZATION

An essential component of LightGBM is the introduction of regularization to balance predictive accuracy with model complexity. This regularization strategy operates on two levels: explicit, through penalties on tree complexity and leaf weights, and implicit, via dimensionality reduction with EFB. This dual mechanism ensures that models remain compact, generalizable, and efficient, setting LightGBM apart from earlier gradient boosting implementations.

The **explicit regularization** is done using the **regularization term** (see

Equation 3.3) and the **optimal leaf weight** (see Equation 3.4), that follow the same formulas as other algorithms like XGBoost, encouraging compact tree structures by explicitly constraining the number of leaves and their output values and penalizing large leaf weights.

The implicit regularization, a distinctive innovation of LightGBM is **Exclusive Feature Bundling (EFB)**, which acts as a form of implicit regularization by reducing the dimensionality of sparse, high-dimensional feature spaces. Features that are mutually exclusive are combined into a single bundled feature:

$$X' = \text{Bundle}(X)$$

This decreases the effective number of features $d \rightarrow d'$, where $d' \ll d$, while maintaining the complete predictive information. By simplifying the input space, EFB reduces the risk of overfitting in high-dimensional datasets and improves computational efficiency without loss of accuracy.

3.2 3D Deep learning models: PointNet

PointNet is a machine learning algorithm developed in the study *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*[56] that belongs to the family of deep learning architectures for 3D data processing, specifically designed to operate directly on point clouds.

A major challenge addressed by PointNet is the limitations associated to applying traditional convolutional neural networks (CNNs) to 3D data[33], because CNNs rely on regular grid-like structures like pixels in images, but point clouds are inherently irregular, unordered, and sparse. PointNet directly operates on the raw point set, without intermediate representations, obtaining a strong performance in applications related to 3D objects.

INPUT REPRESENTATION AND PERMUTATION INVARIANCE

A 3D object or scene in PointNet is represented as an unordered set of points: $P = \{p_1, p_2, \ldots, p_n\}, \quad p_i \in \mathbb{R}^d$ where each p_i is a point in d-dimensional space (normally d=3 for XYZ coordinates, but it may include other features). Permuting the order of points in P should not change the representation or prediction of the model. For any permutation π: $f(\{p_1, p_2, \ldots, p_n\}) = f(\{p_{\pi(1)}, p_{\pi(2)}, \ldots, p_{\pi(n)}\})$

To guarantee this permutation invariance, PointNet applies shared functions to each point and then aggregates them using a symmetric function, the most common one being max pooling. This process can be expressed as:

$$f(P) = \gamma\Big(\underset{p_i \in P}{\mathrm{SYM}}\, h(p_i)\Big)$$

where $h(p_i)$ is a shared transformation function applied to each point, SYM is a symmetric aggregation function such as max pooling, and *gamma* is a final function that maps the aggregated feature vector to the desired output.

NETWORK ARQUITECTURE

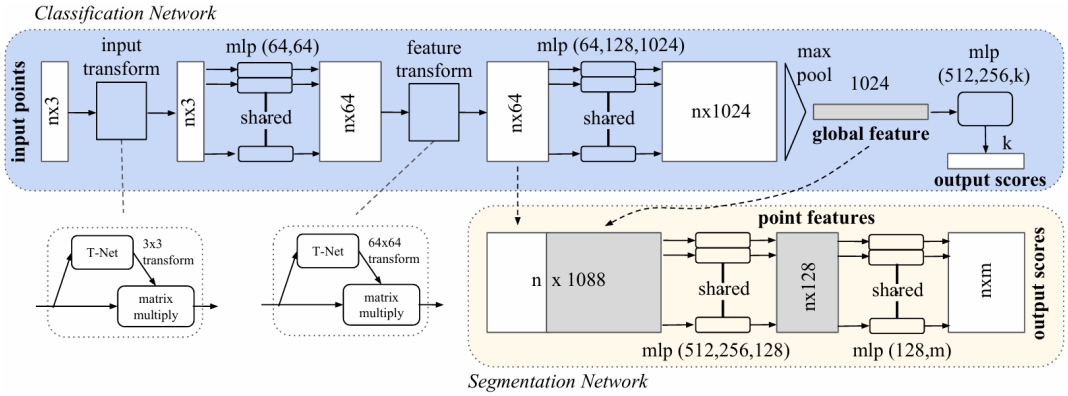


Figure 3.3: Point Net network arquitecture

The input to PointNet is a set of n points in 3D space, each described by its coordinates (xyz), but could also include other features available in the point clouds: $P = \{p_i\}_{i=1}^{n}, \qquad p_i \in \mathbb{R}^3$

$$X = \begin{bmatrix} p_1^\top \\ \vdots \\ p_n^\top \end{bmatrix} \in \mathbb{R}^{n \times 3}$$

INPUT TRANSFORM (T-NET): A T-Net predicts a $3x3$ transformation matrix that aligns the input point cloud to a canonical space. This enables rotations and translations in the data, ensuring the network learns features that are invariant to pose.

$$T_{\text{in}} \in \mathbb{R}^{3\times 3}, \qquad X^{(0)} = X\, T_{\text{in}}$$

MLP(64,64): Each point is independently passed through two fully connected layers (Multi-Layer Perceptrons) with 64 units. These shared MLPs extract features for each point, like local geometric descriptors, while maintaining permutation invariance.

$$\phi_{64,64} : \mathbb{R}^3 \to \mathbb{R}^{64}, \qquad H^{(1)} = \left[\ \phi_{64,64}(x_i^{(0)})\ \right]_{i=1}^{n} \in \mathbb{R}^{n\times 64}$$

FEATURE TRANSFORM (T-NET): A second T-Net predicts a $64x64$ transformation matrix, which aligns the feature space instead of the raw coordinates, making it invariance to different geometric configurations and normalizes feature embeddings across points.

$$T_{\text{feat}} \in \mathbb{R}^{64\times 64}, \qquad H^{(1')} = H^{(1)}\, T_{\text{feat}}$$

MLP(64, 128, 1024): Points are further processed with deeper shared MLPs, increasing the feature dimension up to 1024. This allows the network to capture complex and high-level geometric information at the individual point level

$$\phi_{64,128,1024} : \mathbb{R}^{64} \to \mathbb{R}^{1024}, \qquad H^{(2)} = \left[\ \phi_{64,128,1024}(h_i^{(1')})\ \right]_{i=1}^{n} \in \mathbb{R}^{n\times 1024}$$

MAX POOL & GLOBAL FEATURE (1024): A symmetric max pooling operation aggregates point-wise features into a single global feature vector of dimension 1024. This vector is a compact, order-invariant representation of the entire point cloud, summarizing its most discriminative features.

$$g = \text{MAX}_{i=1,\dots,n}\ H_{i:}^{(2)}\ \in \mathbb{R}^{1024}$$

This algorithm is designed to perform two separated networks: classification network and segmentation network. Depending on the desired result of the model, the next steps are different:

CLASSIFICATION NETWORK:

MLP(512, 256, k) & OUTPUT SCORES: The global feature is passed through fully connected layers of size 512 and 256, before reaching a final layer of dimension k corresponding to the number of object categories. A softmax function is then applied to produce class probabilities for object classification.

$$\hat{y} = \text{Softmax}(W_3\, \sigma(W_2\, \sigma(W_1 g + b_1) + b_2) + b_3)\,, \quad \hat{y} \in \mathbb{R}^k$$

SEGMENTATION NETWORK:

POINT FEATURES n × 1088: The global feature vector (1024) is concatenated with the original per-point features (64), yielding a per-point descriptor of size 1088, so each point prediction is created by both its local properties and the global context of the cloud.

$$Z_i = \left[h_i^{(1)} \parallel g\right] \in \mathbb{R}^{64+1024=1088}, \qquad Z = \left[Z_i^\top\right]_{i=1}^{n} \in \mathbb{R}^{n\times 1088}$$

MLP(512, 256, 128): These shared MLPs refine the concatenated per-point features, progressively reducing dimensionality while extracting relevant representations for segmentation.

$$U = \left[\ \psi_{512,256,128}(Z_i)\ \right]_{i=1}^{n} \in \mathbb{R}^{n\times 128}$$

MLP(128, m) & OUTPUT SCORES: Finally, a per-point classifier outputs predictions of dimension m, which is the number of segmentation classes. A softmax is applied independently to each point, assigning each point to a specific class

$$\hat{Y} = \left[\ \mathrm{Softmax}\big(W_s\, U_{i:} + b_s\big)\ \right]_{i=1}^{n} \in \mathbb{R}^{n\times m}$$

REGULARIZATION

A critical component of PointNet's design is the introduction of explicit mechanisms to ensure regularization and stability during training[56]. Since the architecture relies on T-Nets to predict transformation matrices, an additional loss term is incorporated to constrain these matrices to be close to orthogonal. This avoids degenerate transformations and encourages geometrically meaningful mappings.

The **penalty** is defined as:

$$L_{\text{reg}} = \left\| I - TT^\top \right\|_F^2,$$

where T is the predicted transformation matrix, I is the identity matrix of the appropiate size and applying the Frobenius norm. By minimizing this term, the network enforces approximate orthogonality, stabilizing learning.

PointNet also integrates standard deep learning techniques to improve generalization, like **Dropout**, which is applied in all the layers to prevent overfitting by randomly deactivating neurons during training, which can be expressed as:

$$\tilde{h}_i = m_i \cdot h_i, \quad m_i \sim \text{Bernoulli}(p),$$

where p is the keep probability.

Similarly, **Batch Normalization (BN)** is used after MLP layers to stabilize the distribution of activations and accelerate convergence, normalizing inputs to zero mean and unit variance:

$$\hat{x} = \frac{x - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \quad y = \gamma \hat{x} + \beta,$$

Chapter 4

Development of the railway anomaly detection algorithm

4.1 Data exploration

The dataset employed in this project originates from the Rail3D benchmark dataset, an open-access repository specifically designed for the semantic segmentation and analysis of railway infrastructure in 3D point clouds. Rail3D[25] compiles several sub-datasets acquired through LiDAR scanning technologies in different railway environments.

Each file in the dataset is provided in *.PLY* format, containing a 3D point cloud representation of the railway environment. Each point is characterized by a set of attributes:

- **Spatial coordinates (x, y, z)**. Defining the 3D geometry of the environment.
- **Color information (R, G, B)**. capturing the visual appearance of the scanned scene.
- **Intensity values.** reflecting the return strength of the LiDAR signal, which is correlated with the material properties of the surface.
- **Classification labels.** categorical annotations that assign each point to a specific element of the railway infrastructure.

The Rail3D dataset defines a set of nine semantic classes that cover the most relevant components of railway infrastructure:

1 **Ground.** Track bed and surrounding terrain not belonging to above-ground structures.

2 **Vegetation.** Trees, shrubs, grass, and other foliage in the scene.

3 **Rail.** Steel tracks and directly attached metallic rail components.

4 **Poles.** Vertical support structures unrelated to buildings (e.g., masts, posts).

5 **Wires.** Overhead and side wires, including catenary and auxiliary wiring.

6 **Signaling.** Railway signaling devices and supports (e.g., signal heads, indicators).

7 **Fences.** Barriers and guardrails delimiting the railway right-of-way.

8 **Installation.** Technical installations and equipment cabinets near the track.

9 **Building.** Architectural structures such as stations, sheds, or nearby buildings.

In this project, only the SNCF and HMLS sub-datasets were used, and they will be analyzed separately to have a better understanding of each of the datasets.

SNCF

The SNCF subset was collected on operational railway tracks managed by the French National Railway Company (Société Nationale des Chemins de fer Français), and acquired using terrestrial LiDAR scanning technologies, enabling a dense and highly accurate 3D representation of the track environment. This subset covers approximately 1,6 kilometers of railway infrastructure and is composed of 16 individual point cloud files, amounting to a total of 143.313.588 3D points.

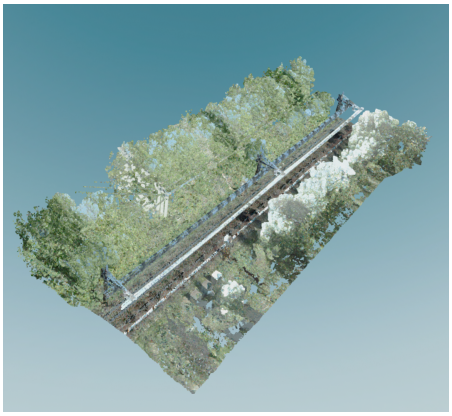


Figure 4.1: Visual representation of the original data *(sncf_ 05)*

FEATURE ANALYSIS

The descriptive statistics of the SNCF dataset show the scale and variability of the features(see Figure 4.2).

	x	y	z	red	green	blue	intensity	classification
count	143313588.000000	143313588.000000	143313588.000000	143313588.000000	143313588.000000	143313588.000000	143313588.000000	143313588.000000
mean	1701994.179032	3112133.662065	9.047610	55.419612	66.174563	52.028624	13.489364	1.740503
std	275.237262	351.890116	2.699474	35.635228	42.365972	40.104309	11.131396	1.023435
min	1701501.466003	3111510.298004	2.274000	0.000000	0.000000	0.000000	0.000000	1.000000
25%	1701764.276978	3111845.849976	7.537000	30.000000	36.000000	27.000000	6.000000	1.000000
50%	1701976.596008	3112112.880005	7.716000	50.000000	59.000000	44.000000	10.000000	2.000000
75%	1702227.198975	3112430.187012	10.253000	71.000000	84.000000	63.000000	19.000000	2.000000
max	1702511.680054	3112792.452026	27.650000	255.000000	255.000000	255.000000	255.000000	8.000000

Figure 4.2: Descriptive statistics for the SNCF database

The **x and y coordinates** reflect the longitudinal and lateral coverage of the scans, with a very high range of values, while **z** values are much more limited, ranging from 2.27 to 27.65, which matches the expected elevation of railway tracks.

The **RGB channels** remain within the expected 0–255 range, with averages around red = 55, green = 66, and blue = 52. These values confirm the predominance of darker tones in the data, while the relatively high standard deviations point to the variability introduced by different elements of the infrastructure. **Intensity** values also range from 0 to 255 but show a low mean (13.49) and median (10.00), indicating a strong skew toward weaker reflections, and the presence of occasional high values suggests the influence of reflective materials.

The distribution plots(see Figure 4.3) complement the descriptive statistics by illustrating the shape of each feature's variability. The **spatial coordinates x and y** follow almost uniform distributions, reflecting the continuous coverage of the LiDAR scans along the track. By contrast, **z** shows a highly concentrated peak at ground level decreasing towards higher values, corresponding to high structures such as poles and wires. The **RGB** histograms are right-skewed, maintaining the prevalence of darker tones. Similarly, the **intensity** distribution is heavily skewed toward the lower end, with most points concentrated below 50, but with some outliers. These patterns show the heterogeneity of the dataset and justify feature preprocessing before modelling.

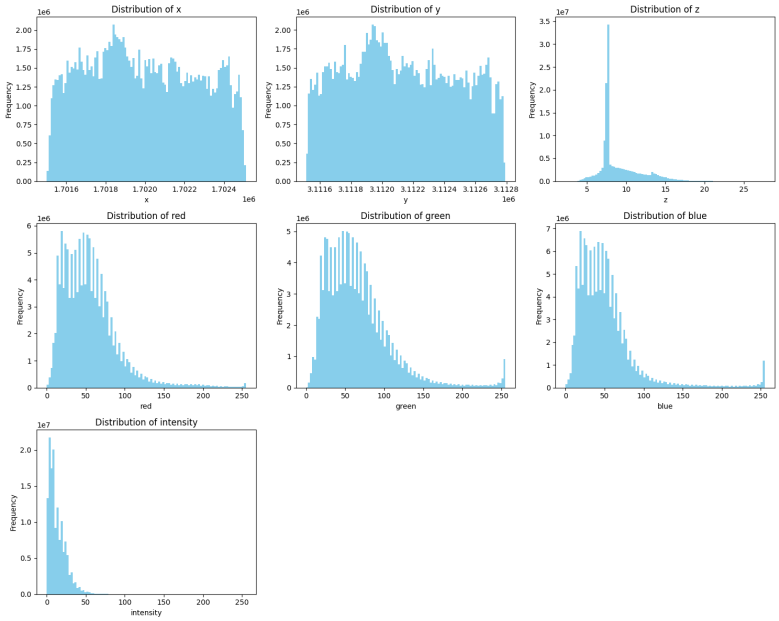


Figure 4.3: Distribution of the features from the SNCF database

The correlation matrix (see Figure 4.4) confirms that the spatial coordinates are largely independent from both color and intensity features, with near-zero correlations between **x, y** and the remaining variables. This reinforces their role as purely geometric descriptors of the scene.

Among the **color channels**, very strong positive correlations are observed (red–green = 0.94, green–blue = 0.92), which reflects the natural coupling of RGB values in real-world surfaces, making the color features valuable to detect if a point is part of infrastructure materials or vegetation. The **z** coordinate shows moderate positive correlations with color values (0.33–0.42) and with the classification label (0.41), which suggests that elevation helps distinguish classes such as poles, wires, and ground. **Intensity** exhibits very very low correlation with all other features, underlining its importance as an independent attribute that complements both geometry and color in the dataset.

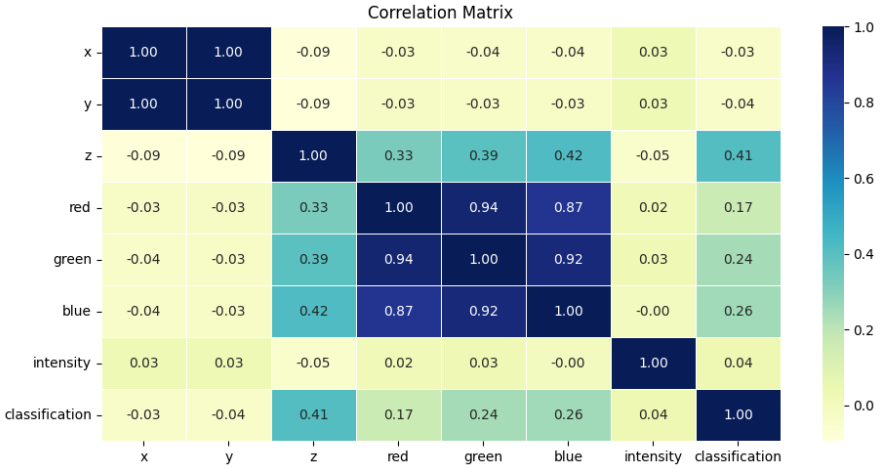


Figure 4.4: Correlation features from the SNCF database

CLASSIFICATION ANALYSIS

The **global class distribution** of the entire database reveals a strong imbalance among the nine semantic categories defined in Rail3D. The two dominant classes are Class 1 (Ground) with 51.2 million points (35.74%) and Class 2 (Vegetation) with 85.5 million points (59.66%), together, these categories account for more than 95% of the dataset.

Infrastructure-related elements are far less represented: Class 3 (Rail) contributes only 2.59 million points (1.81%), while Class 4 (Poles) and Class 5 (Wires) account for 0.97% and 1.53% of the dataset, respectively. The remaining categories appear only marginally. Class 6 (Signaling), Class 7 (Fences), and Class 8 (Installation) represent 0.08%, 0.16%, and 0.06% of all points, respectively. Notably, Class 9 (Building) is entirely absent from the SNCF subset, despite being part of the Rail3D taxonomy.

The global class distribution shows the dominance of environmental features over railway infrastructure components, so class imbalance must be addressed.

When the class distribution is examined at individual **per-file level** rather than the full dataset, the imbalance becomes even more pronounced. While Ground (Class 1) and Vegetation (Class 2) are consistently present and remain the dominant categories in every scan, many infrastructure-related classes are either absent

or appear in very small proportions. For example, Rail (Class 3), Poles (Class 4), and Wires (Class 5) are present across all files but never exceed a few percent of the total points. Minority classes such as Signaling (Class 6), Fences (Class 7), and Installation (Class 8) are often missing altogether, specifically, Class 6 is absent in 69% of the files, Class 7 in 31%, and Class 8 in 50%. Finally, Buildings (Class 9) are completely absent from the SNCF subset.

The per-file analysis reports the difficulty of modeling minority classes: not only are they underrepresented globally, but they are also missing in a substantial fraction of the files.

HMLS:

The HMLS subset was generated through mobile LiDAR acquisitions conducted at high speed along railway corridors in Hungary, obtaining wide coverage and diversity of operating conditions, capturing long stretches of track with varying surrounding environments. This subset represents approximately 2,14 kilometers of railway lines and consists of 29 point cloud files, with a total of 104.958.796 points.

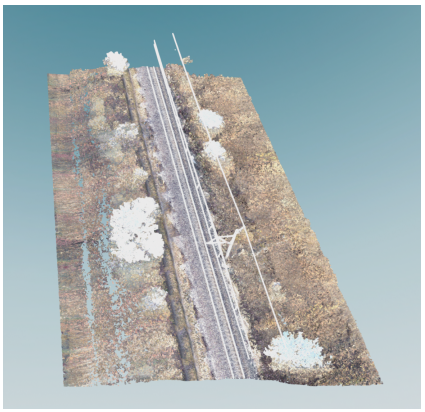


Figure 4.5: Visual representation of the original data *(hmls_ 01)*

FEATURE ANALYSIS

The statistical examination of the HMLS dataset provides a comprehensive analysis of the characteristics of its features (see Figure 4.6).

	x	y	z	red	green	blue	intensity	classification
count	104958796.000000	104958796.000000	104958796.000000	104958796.000000	104958796.000000	104958796.000000	104958796.000000	104958796.000000
mean	543037.918757	173978.778468	164.022542	85.988699	82.936384	73.649465	8968.754883	1.414304
std	112676.219672	10670.773434	61.785419	61.991550	60.524784	62.290590	10716.478516	1.025413
min	439040.000000	159100.000000	93.031502	0.000000	0.000000	0.000000	0.000000	1.000000
25%	439606.407227	162888.603760	96.266998	45.000000	46.000000	37.000000	70.000000	1.000000
50%	440257.913086	183532.873047	218.649002	67.000000	65.000000	55.000000	116.000000	1.000000
75%	665803.299072	183800.725098	220.162003	104.000000	92.000000	83.000000	17399.000000	2.000000
max	666390.666504	183862.999023	237.899994	255.000000	255.000000	255.000000	65535.000000	9.000000

Figure 4.6: Descriptive statistics for the HMLS database

The **x and y coordinates** cover very large ranges, with mean values around 543,038 and 173,978 respectively, and standard deviations in the order of tens of thousands, reflecting the broad horizontal and lateral coverage of the LiDAR acquisition. However, the **z** dimension ranges from approximately 93 to 238, with a mean of 164, which is consistent with the representation of the elevation of railway elements, a ot of them being at ground level.

The **RGB channels** are all three within the 0–255 range, with average values of red = 86, green = 83, and blue = 74, indicating a predominance of darker or mid-range tones across the dataset. The **intensity** feature is particularly distinctive. While values theoretically extend up to 65,535, the median intensity is only 116, far below the mean of = 8,969, which can cause a strong skew in the distribution, and confirm the presence of outliers.

The histograms (see Figure 4.7) provide a more detailed perspective on these trends.

Both **x and y** show clear peaks and gaps, reflecting the scanning trajectory and the segmentation of the LiDAR acquisitions into separate files. The **z** distribution shows two strong groups: one that could be interpreted as near ground level, and another at higher values, associated with higher elements like poles, wires, and signaling infrastructure. The **RGB** histograms are right-skewed, with the majority of points concentrated below 100, also showing that their are a lot of darker colors in the images. The **intensity** histogram is the most extreme, with a massive peak at very low values and a long, thin tail extending towards the maximum.

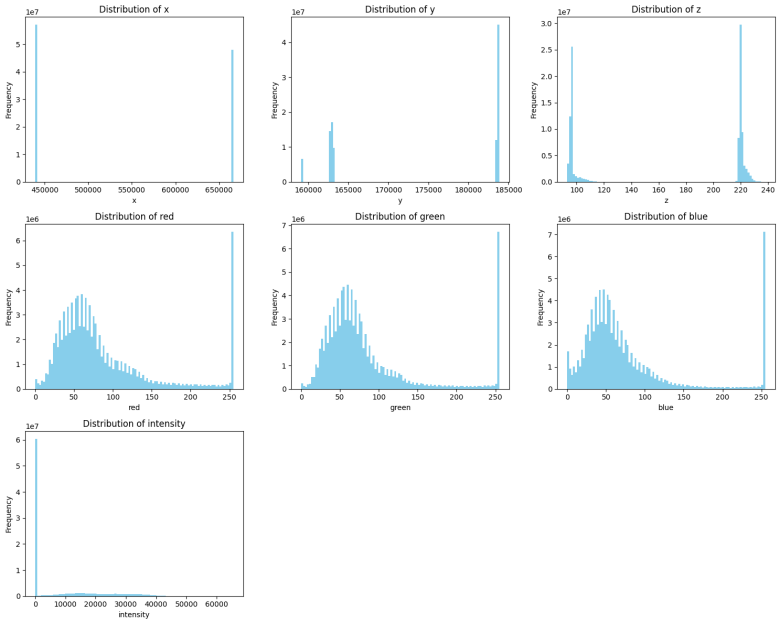


Figure 4.7: Distribution of features in the HMLS database

The correlation matrix (see Figure 4.8) adds another dimension to this analysis. The **xyz** coordinates are very independent from both color and intensity, showing that they represent the spacial distribution. By contrast, the **RGB** channels are very strongly correlated (red–green = 0.96, green–blue = 0.95), which is typical because colors tend to be a combination of the three features. **Intensity** remains weakly correlated with other features.

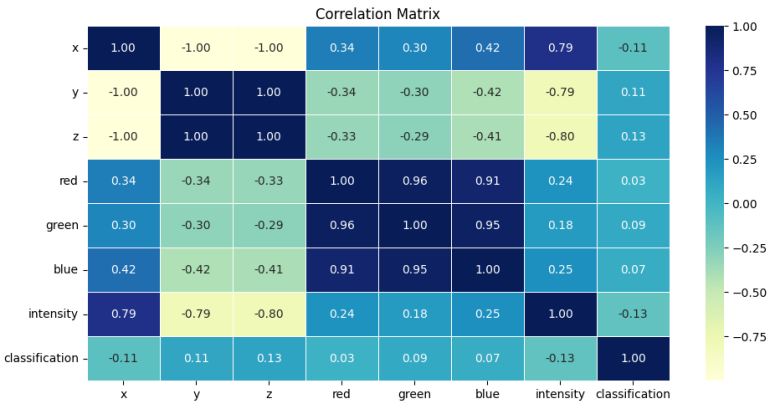


Figure 4.8: Correlation between features in the HMLS database

CLASSIFICATION ANALYSIS

The **global distribution** of semantic classes in the HMLS dataset is strongly imbalanced. The two dominant categories are Ground (Class 1) and Vegetation (Class 2), which account for 66.42% and 30.69% of all points respectively. Together, these classes represent more than 97% of the dataset, showing that the images obtained of the infrastructure contain more of other elements.

Infrastructure-related elements are represented only marginally. Rail (Class 3) contributes 1.10%, Poles (Class 4) 0.64%, and Wires (Class 5) 0.41%, and while these classes are critical for the semantic understanding of railway infrastructure, the low percentage of points from these classes makes it very difficult for models to train properly. Even more problematic are the minority categories: Signaling (Class 6) accounts for only 0.07%, Fences (Class 7) for 0.34%, and Installation (Class 8) for 0.07%. Buildings (Class 9) represent only 0.26%, and are present in very few files, making them particularly challenging for segmentation.

A **per-file analysis** has a much more severe class imbalance, where only Classes 1, 2, 3, and 5 appear consistently across all 29 files. Many other categories are missing in a big fraction of the dataset: Class 4 (Poles) is absent in one file (3.45%), Class 6 (Signaling) is missing in 41% of the files, Class 7 (Fences) in 72%, and Class 8 (Installation) in 21%. Most critically, Class 9 (Buildings) is missing in nearly 90% of the files, which leaves only 3 files with all the classes.

The abundance of points that are part of Classes 1 and 2 make it difficult to obtain images with the classes that are really needed for our algorithm, which are the points from railway infrastructure (class 3, 4, 5), limiting the creation of models and risking those models to only interpret everything as part of the majority classes.

4.2 Data processing

After the exploratory analysis carried out in the previous section, important differences were observed in the scale and range of the variables, and an unbalanced distribution of the classes within the dataset. This is the main reason for applying a data processing in the raw data, in order to reduce the bias towards the majority classes, and dividing the whole dataset into different subsets: train, validation and test, to design and train the best possible model. To this end, the data processing stage includes normalization procedures, dataset splitting, resampling strategies to

handle class imbalance, and filtering techniques aimed at refining the information before its use in the modeling phase.

4.2.1 Data normalization

In contrast to the exploratory analysis, where both datasets were studied independently in order to identify their specific statistical characteristics, in this stage the two datasets Will be processed together. This choice is motivated by the fact that both sources will be joined in the modelling stage, so the normalization of features needs to be consistent between both databases. Since each variable represents a different characteristic of the data, with distinct numerical ranges and distributions,it is necessary to apply separate normalization techniques to each of them:

SPATIAL COORDINATES (x, y, z): The chosen normalization method for the spatial coordinates is centroid normalization, also referred to as unit sphere normalization. This approach was applied because the spatial ranges of the data are unbalanced: the longitudinal and lateral dimensions (x, y) have much larger values compared to the vertical axis (z). Without this adjustment, the horizontal dimensions would dominate the learning process, reducing the contribution of the height information.

Centroid normalization ensures that all points are fit within a unit sphere, consisting of two consecutive steps: centering and scaling. First, each point cloud is translated so that its centroid lies at the origin, which given a set of N points $p_i = (x_i, y_i, z_i)$, the centroid is computed as:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} p_i$$

Each point is then shifted by the centroid and rescaled using the maximum Euclidean distance of any point to the centroid:

$$\tilde{p}_i = \frac{p_i - \mu}{\max_j \|p_j - \mu\|_2}$$

This transformation maps the point cloud into a unit sphere centered at the origin, constraining the new coordinate range approximately to $[-1, 1]$.

This type of normalization is particularly suitable for spatial coordinates since it removes variations due to translation and scale, while preserving the relative

geometry of the structures. Its effectiveness has been demonstrated in point cloud deep learning literature, starting from the seminal PointNet architecture[55] and continuing in more recent benchmark studies [69, 85]. These works consistently recommend centroid-based unit sphere normalization as a reliable preprocessing step for 3D point clouds.

In the statistical summary obtained, it can be observed that the normalized coordinates (x,y,z) are ranged between $[-1, 1]$, however, this does not imply that the minimum and maximum values of each coordinate exactly reach -1 and 1. The normalization procedure ensures that the Euclidean distance of all points to the centroid remains smaller than or equal to one, expressed as:

$$\sqrt{x^2 + y^2 + z^2} \leq 1$$

COLORS (red, green, blue): The RGB channels show the original color captured by the scan for each point, originally stored as integer values within the range $[0, 255]$. If left unprocessed, these magnitudes would not be directly comparable to the rest of the variables. To address this, a min-max scaling[2] was applied, standardizing the three color dimensions of both databases to a $[0, 1]$ range:

$$\tilde{c}_i = \frac{c_i}{255}, \quad c_i \in \{\text{red}, \text{green}, \text{blue}\}$$

INTENSITY: The intensity attribute encodes the reflectance of each laser return, and unlike the spatial coordinates or the RGB values, its numerical range strongly depends on the characteristics of the acquisition system and the scanning conditions. In the SNCF dataset, the raw intensity values range from $0to255$, but in the HMLS dataset the observed values go from $0to65535$. These differences are substantial: while SNCF intensities are confined to a single byte representation $2^8 = 255$, HMLS intensities are stored in two bytes $2^{16} = 65535$.

If the two datasets were directly merged without normalization, the disparity in ranges would cause the HMLS values to dominate the learning process, making the contribution of SNCF intensities negligible. To avoid this, a min–max normalization was applied independently to each dataset, so that both are rescaled into a common range [0,1].

For the SNCF dataset:

$$\tilde{I}_i^{\text{SNCF}} = \frac{I_i - I_{\text{min}}^{\text{SNCF}}}{I_{\text{max}}^{\text{SNCF}} - I_{\text{min}}^{\text{SNCF}}}, \quad I_{\text{min}}^{\text{SNCF}} = 0,\ I_{\text{max}}^{\text{SNCF}} = 255$$

For the HMLS dataset:

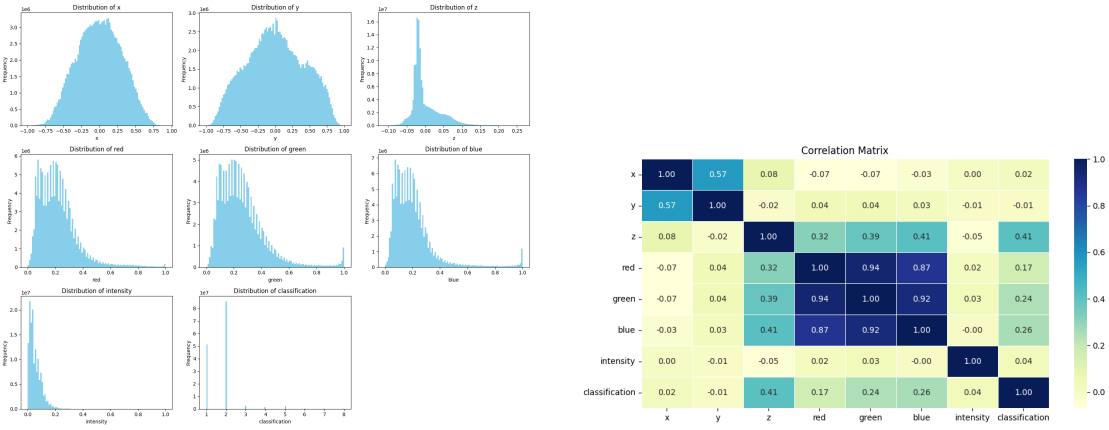
$$\tilde{I}_i^{\text{HMLS}} = \frac{I_i - I_{\text{min}}^{\text{HMLS}}}{I_{\text{max}}^{\text{HMLS}} - I_{\text{min}}^{\text{HMLS}}}, \quad I_{\text{min}}^{\text{HMLS}} = 46,\ I_{\text{max}}^{\text{HMLS}} = 32767$$

The adoption of dataset-specific normalization is particularly justified in multi-modal or multi-source scenarios such as this one, where acquisition protocols are different. Comparable strategies have been reported in LiDAR processing literature, where min–max scaling per sensor ensures consistent feature ranges and prevents model bias towards datasets with broader raw distributions[5].

CLASSIFICATION: This variable will not be normalized because it is our target variable, and the classes represented are consistent between both databases.

The analysis of the **histograms** (see Figures 4.9a,4.10a) before and after normalization shows that, with the exception of the spatial coordinates, the distributions of the variables remain unchanged apart from a rescaling of their minimum and maximum values, except the xyz coordinates that follow the shape of a normal distribution,

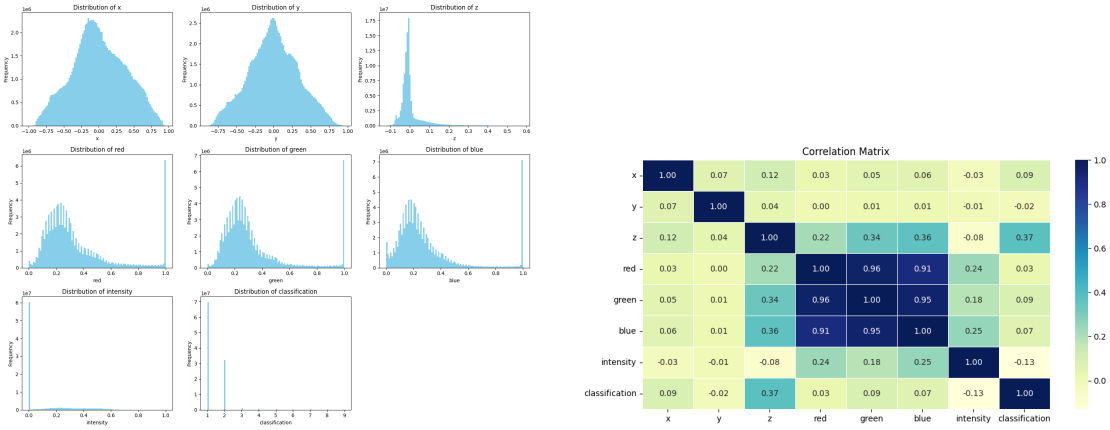
Regarding the **correlation matrix** (see Figures 4.9b,4.10b) the normalization of the spatial coordinates has an impact in the final result. In the xyz coordinates, the centroid normalization is a non-linear transformation, but the correlation only measures linear dependencies, so their values are affected by the normalization. For the rest of the features, the normalization min-max is a linear transformation so it maintains the same values as before.



(a) Distribution of variables

(b) Correlation between variables

Figure 4.9: Analysis of features after normalization from the SNCF database



(a) Distribution of variables

(b) Correlation between variables

Figure 4.10: Analysis of features after normalization from the HMLS database

4.2.2 Data splitting

For the training of the models, the dataset must be divided into three different subsets: a training set, used for fitting the parameters of the model, a validation set, used for hyperparameter tuning and early stopping, and a test set, reserved exclusively for the final evaluation of generalization performance. In this project, the data was divided according to an 80/10/10 split[6]. For the SNCF database:

$$\text{TRAIN: } 0.8 \cdot 16 = 12.8 \approx 12 \text{ files}$$

$$\text{VALIDATION: } 0.1 \cdot 16 = 1.6 \approx 2 \text{ files}$$

$$\text{TEST: } 0.1 \cdot 16 = 1.6 \approx 2 \text{ files}$$

For the HMLS database:

$$\text{TRAIN: } 0.8 \cdot 29 = 23.2 \approx 23 \text{ files}$$

$$\text{VALIDATION: } 0.1 \cdot 29 = 2.9 \approx 3 \text{ files}$$

$$\text{TEST: } 0.1 \cdot 29 = 2.9 \approx 3 \text{ files}$$

Due to the significant class imbalance present in the point clouds, a random allocation of files could result in subsets without the presence of some classes, or with highly skewed distributions, to address this, a custom algorithm was implemented to optimize the allocation of files across the three sets.

The algorithm generates multiple candidate splits by randomly shuffling the files and evaluates each split by computing the class distribution in terms of point counts, an assigning a score to each candidate based on the imbalance between sets, defined as the deviation of each class count from the average across train, validation, and test. The split with the lowest imbalance score is selected. Importantly, the algorithm ensures that all classes are represented in each subset, which would not be guaranteed in a purely random distribution.

Finally, the normalized data from both SNCF and HMLS were merged into the same directories (see Appendix B), allowing the model to be trained jointly on both datasets. The resulting class distributions in terms of point counts are shown below:

```
Train:                                  Validation:                             Test:
- Class 1: 90,411,317 pts (50.59%)      - Class 1: 14,047,774 pts (43.78%)      - Class 1: 16,467,934 pts (43.94%)
- Class 2: 81,278,904 pts (45.48%)      - Class 2: 16,845,382 pts (52.50%)      - Class 2: 19,591,092 pts (52.28%)
- Class 3: 2,845,499 pts (1.59%)        - Class 3: 398,290 pts (1.24%)          - Class 3: 505,866 pts (1.35%)
- Class 4: 1,495,879 pts (0.84%)        - Class 4: 281,273 pts (0.88%)          - Class 4: 280,724 pts (0.75%)
- Class 5: 1,937,580 pts (1.08%)        - Class 5: 353,436 pts (1.10%)          - Class 5: 327,232 pts (0.87%)
- Class 6: 108,056 pts (0.06%)          - Class 6: 12,176 pts (0.04%)           - Class 6: 64,922 pts (0.17%)
- Class 7: 428,718 pts (0.24%)          - Class 7: 41,886 pts (0.13%)           - Class 7: 113,879 pts (0.30%)
- Class 8: 87,233 pts (0.05%)           - Class 8: 4,725 pts (0.01%)            - Class 8: 69,670 pts (0.19%)
- Class 9: 116,435 pts (0.07%)          - Class 9: 102,596 pts (0.32%)          - Class 9: 53,906 pts (0.14%)
```

Figure 4.11: Final distribution of classes in each subset

The final distribution is relatively balanced across the three subsets, with each class represented in all partitions. Minor differences remain, which are explained by the file-based splitting procedure: since files are not subdivided at the point level, some subsets contain files with slightly higher prevalence of certain classes, but the resulting distribution achieves the necessary balance to train and evaluate the model reliably.

4.2.3 Data resampling

One of the main challenges encountered during the preparation of the datasets is the class imbalance. Such imbalance would inevitably bias the learning process towards the majority classes, limiting the model's capacity to correctly recognize underrepresented categories. To mitigate this issue, a resampling strategy was applied, aimed both at amplifying the representation of minority classes and at reducing the dominance of the majority ones.

The absolute number of points in the final distribution was chosen like the original dataset sizes, thereby preserving the overall scale of the data while modifying

only the relative proportions among classes.

The resampling was designed to enforce a target distribution of 60/40 between classes representing railway infrastructure (Classes 3, 4, and 5) and classes not directly associated with the train (Classes 1, 2, 6, 7, 8, and 9). After resampling, the non-train classes account for a total of 84,765,934 points (60.00%), while the train-related classes amount to 56,510,622 points (40.00%).

Within the non-train group, an additional adjustment was introduced in order to further balance the contribution of different infrastructure elements. Specifically, a 95/5 split was applied between Classes 1–2, which dominate the dataset, and Classes 6–9, which are less frequent. As a result, Classes 1 and 2 jointly contain 80,323,282 points (94.76%), while Classes 6–9 together comprise 4,442,652 points (5.24%).

In order to obtain this distribution, it is necessary to apply sampling techniques to each of the classes:

- Class 1,2: Undersampling
- Class 3,4,5: Oversampling
- Class 6,7,8,9: Oversampling

UNDERSAMPLING:

Random undersampling[7] is a common approach to address class imbalance by reducing the number of samples in overrepresented classes. Instead of replicating minority classes, this method randomly removes samples from majority classes until the desired distribution is achieved. In this way, the dataset size is reduced while having the class proportions match the predefined target.

Formally, let $N = \sum_{k=1}^{C} n_k$ be the total number of points in the dataset. The goal of undersampling is to transform n_k into $\tilde{n}_k$, such that

$$\tilde{n}_k = \alpha_k \, n_k, \quad 0 < \alpha_k \leq 1$$

where $alpha_k$ is the undersampling multiplier applied to the class. For majority classes, the multiplier is chosen smaller than 1, while for minority classes, the multiplier is equal to 1.

The multipliers are determined such that

$$\frac{\tilde{n}_k}{\sum_{j=1}^{C} \tilde{n}_j} \approx p_k$$

where p_k is the target proportion of class. It is calculated using the following equation: The multiplier is then

$$\alpha_k = \frac{n_k^{\text{target}}}{n_k}$$

The main drawback is the loss of information from discarded samples, which may negatively affect the model if the removed points contained relevant variability.

For this project, class 1 and class 2 need to be reduced, so it is necessary to obtain the multiplier needed to reduce them:

$$\alpha_{1,2} = \frac{111{,}600{,}000}{238{,}642{,}403} \approx 0.4678$$

OVERSAMPLING:

To mitigate the imbalance of minority classes, an oversampling strategy based on the **Synthetic Minority Over-sampling Technique (SMOTE)**[12] was implemented. Unlike random duplication of samples, SMOTE generates synthetic data points by interpolating between existing samples of the same class. This approach increases the representativeness of the minority classes without artificially inflating the dataset with repeated instances, thereby improving generalization.

Let be the feature vector of a minority-class sample. For each x_i its k-nearest neighbors from the same class are first identified using the Euclidean distance:

$$\|x_i - x_j\|_2 = \sqrt{\sum_{\ell=1}^{d} \big(x_{i,\ell} - x_{j,\ell}\big)^2}.$$

From these neighbors, one sample is randomly selected, and a new synthetic point is then generated as a combination of both points:

$$x_i^* = x_i + \lambda\big(x_i^{(r)} - x_i\big) = (1-\lambda)\, x_i + \lambda\, x_i^{(r)}.$$

This interpolation guarantees that the new point lies on the line segment between the original point and its neighbor, preserving the structure of the feature space after normalization. Repeating this process for multiple neighbors and values of λ allows for generating as many synthetic points as required.

The number of synthetic points to be created can be defined in two ways. The first option is to specify a multiplicative factor

$$N_k^{\text{tgt}} = \alpha_k \, N_k^{\text{orig}},$$

where N^orig is the original number of points in class and N^tgt is the desired number after oversampling. The number of new synthetic samples is then:

$$N_k^{\text{new}} = N_k^{\text{tgt}} - N_k^{\text{orig}} = (\alpha_k - 1) \, N_k^{\text{orig}}.$$

Alternatively, it is possible to define the absolute number of desired points per class, which gives:

$$N_k^{\text{new}} = \max\bigl(0,\, N_k^{\text{tgt}} - N_k^{\text{orig}}\bigr).$$

Both methods are equivalent in practice. For this project, the multiplicative factor approach was chosen, establishing the multiplier as:

For classes 3,4,5: $\alpha_{3,4,5} = \frac{N_{3,4,5}^{\text{tgt}}}{N_{3,4,5}^{\text{orig}}} = \frac{53{,}612{,}886}{5{,}953{,}922} \approx 9$

For classes 6,7,8,9: $\alpha_{6,7,8,9} = \frac{N_{6,7,8,9}^{\text{tgt}}}{N_{6,7,8,9}^{\text{orig}}} = \frac{4{,}238{,}297}{747{,}157} \approx 6$

As we can see, for this method the multiplier needs to be a number greater than 1 in order to create more points than the original ones.

The final distribution of the classes after applying both methods for the different elements:

```
Original Distribution:                    Resampled Distribution:
- Class 1: 90,411,317 pts (50.59%)        - Class 1: 42,297,886 pts (29.94%)
- Class 2: 81,278,904 pts (45.48%)        - Class 2: 38,025,396 pts (26.92%)
- Class 3: 2,845,499 pts (1.59%)          - Class 3: 25,609,491 pts (18.13%)
- Class 4: 1,495,879 pts (0.84%)          - Class 4: 13,462,911 pts (9.53%)
- Class 5: 1,937,580 pts (1.08%)          - Class 5: 17,438,220 pts (12.34%)
- Class 6: 108,056 pts (0.06%)            - Class 6: 648,336 pts (0.46%)
- Class 7: 428,718 pts (0.24%)            - Class 7: 2,572,308 pts (1.82%)
- Class 8: 87,233 pts (0.05%)             - Class 8: 523,398 pts (0.37%)
- Class 9: 116,435 pts (0.07%)            - Class 9: 698,610 pts (0.49%)
```

Figure 4.12: Final distribution of classes after resampling

4.2.4 Data filtering

As a final step in the preprocessing process, a filtering procedure was implemented to limit the dataset to the structural elements of the railway infrastructure, in particular, only the points associated with Classes 3, 4, and 5 were maintained, since these correspond to rails, poles, and catenary cables.

This selection enables the construction of a database that will be used as input for the segmentation model, that will use this information to classify the different elements in the rail infrastructure, so the models can be trained more efficiently and without interference from non-infrastructure elements.

The filtering can be formally expressed as the following subset of the original point cloud:

$$P_{\text{infra}} = \{\, p_i \in P \mid y_i \in \{3, 4, 5\} \,\}$$

where P denotes the full set of points, p_i represents each point and y_i its corresponding class label.

4.3 Modelling

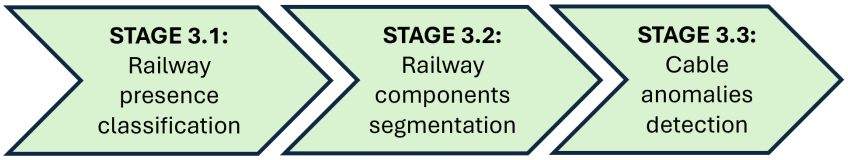


Figure 4.13: Stages of the modelling

4.3.1 Railway presence classification

The first stage of the modelling process (see Figure 4.13) focuses on **railway presence classification**, which aims to separate points belonging to the railway infrastructure and those corresponding to external or irrelevant elements. This step changes the original multi-class classification task into a binary problem, simplifying the data while preserving the essential distinction required for subsequent analysis.

The **input** to this stage consists of preprocessed point cloud data, where each point is characterized by its spatial coordinates, color channels, and intensity values. The **output**, in contrast, is a binary label that indicates whether a point belongs to the railway infrastructure (such as rails, poles, or catenary cables) or to non-infrastructure elements (including vegetation, ground, and surrounding objects).

This reduction to a binary classification problem is very important in the pipeline because, it creates a first filter for the 3D point clouds obtained in the raw data, making it possible to use the algorithm for new, non-classified data. It also improves the computational effort required in the complete model by reducing the size of the original database, eliminating more than half the points, which are not relevant in the anomaly detection.

MULTI-CLASS TO BINARY CLASS CONVERSION

The dataset initially contains nine semantic classes, so in order to adapt the data to the binary setting required for railway presence classification, a custom conversion was implemented, performed through a dedicated function, which systematically maps the original class labels into two categories: infrastructure points (Positive class), corresponding to rails, poles, and cables, and non-infrastructure points (Negative class), containing the rest of the classes. The mapping is achieved by verifying whether the label of each point belongs to a user-defined set of target classes, allowing for a flexible and reusable model.

The conversion starts by extracting the features (coordinates, color channels, and intensity), and reassigning the labels into the binary format by applying a logical test:

$$y_i = \begin{cases} 1, & \text{if } c_i \in \mathcal{C}_{\text{target}} \\ 0, & \text{if } c_i \notin \mathcal{C}_{\text{target}} \end{cases}$$

Following the conversion, the binary dataset was analyzed in terms of class distribution. As expected after the resampling done in the preprocessing stage, approximately 60% of the points belong to Class 0 (non-infrastructure) and 40% to Class 1 (infrastructure). This distribution avoids the severe imbalance that the raw dataset had, reducing the bias in the modelling.

The definition of infrastructure in the design is not restricted to a fixed subset of classes, so the target classes parameter can be easily modified, making the

framework suitable for alternative configurations or future datasets. By allowing the redefinition of the target classes, the method extends beyond the current application to rails, poles, and cables, and could be readily adapted to other domains.

XGBoost

The XGBoost model was trained using the Google Colab environment with an NVIDIA A100 GPU, which can handle large-scale point cloud datasets. The model was implemented with the XGBClassifier interface, and its hyperparameters were carefully selected to balance predictive accuracy with generalization capacity.

The hyperparameters used in the training of the model were the following:

- ***objective='binary:logistic':*** Specifies that it is a binary classification problem, with logistic regression applied at the output layer to produce probabilities between 0 and 1.
- ***tree_method='gpu_hist'*** and ***predictor='gpu_predictor'***: Uses GPU-accelerated histogram-based algorithms for tree construction and prediction, reducing training time and allows the model to scale to millions of points efficiently.
- ***eval_metric='aucpr':*** Sets the area under the precision–recall curve as the evaluation metric during training. This choice is particularly appropriate for imbalanced datasets, as it emphasizes the ability to correctly identify the positive (infrastructure) class.
- ***scale_pos_weight:*** Compensates for class imbalance by scaling the contribution of positive samples in the loss function, so that misclassifications of the minority class (infrastructure) are penalized more heavily, improving sensitivity. It is defined as the ratio between the number of negative and positive samples in the validation set:

$$\text{scale\_pos\_weight} = \frac{N_{\text{class 0}}}{N_{\text{class 1}}}$$

- ***max_depth:*** Limits the maximum depth of each tree. A relatively small value prevents the model from overfitting by restricting overly complex tree structures, favoring generalization instead.

- ***learning_rate:*** Controls the contribution of each new tree to the overall ensemble. A smaller learning rate stabilizes training but requires more trees to achieve optimal performance.
- ***n_estimators:*** Defines the maximum number of boosting rounds.
- ***subsample=0.8:*** Randomly samples a percentage of the training data for each boosting round, reducing overfitting, established in 0.8 for all of our models.
- ***n_jobs=-1:*** Uses all available CPU threads for parallelization of non-GPU tasks, accelerating data preprocessing and auxiliary computations.
- ***random_state=42:*** Fixes the random seed to ensure reproducibility of results.

During training, a validation subset was used to monitor the model's performance on unseen data. This separation is essential to detect overfitting: while the training accuracy may continue to improve, if the validation performance is not improved it may indicate that the model is the losing generalization ability.

- ***early_stopping=50:*** If no improvement in the evaluation metric (aucpr) was observed over a number of consecutive boosting rounds, 50 for our model, the training process is stopped prematurely, preventing unnecessary computations and choosing the final model that achieved the best validation performance, rather than the maximum number of estimators.

  This parameter ensures that the maximum number of estimators is equal to the n_estimators parameter, but could be less than this number if the early stopping is activated:

$$n_{\text{final}} \leq n_{\text{estimators}}$$

HYPERPARAMETER SELECTION:

To identify the best configuration for the model, experimental models were trained using different sets of hyperparameters, to see how variations in depth, learning rate and number of estimators affect the performance of the classifier.

Model	depth	learning rate	estimators	Best iteration	AUC PR Train	AUC PR Validation
MODEL 1	3	0.07	700	697	0.9515	0.8692
MODEL 2	5	0.10	500	459	0.9772	0.8692
MODEL 3	8	0.10	500	177	0.9828	0.8394
MODEL 4	4	0.10	500	338	0.9607	0.8734
MODEL 5	2	0.10	500	490	0.9232	0.8504
MODEL 6	3	0.10	500	489	0.9519	0.8695

Table 4.1: Comparison of different XGBoost model configurations and their performance on training and validation sets.

Out of these combinations, Model 4 was the best model for the binary classification, based on the model's balance between training and validation performance:

Model	depth	learning rate	estimators	Best iteration	AUC PR Train	AUC PR Validation
MODEL 4	4	0.10	500	338	0.9607	0.8734

Table 4.2: Selected model with the best performance

While deeper models such as Model 3 achieved slightly higher training scores (*AUC PR Train = 0.9828*), they exhibited a notable drop in validation accuracy (*AUC PR Train =0.8394*), suggesting signs of overfitting. In contrast, Model 4 reached a competitive training performance (*AUC PR Train = 0.9607*) while also obtaining the highest validation score across all models (*AUC PR Train =0.8734*).

The best iteration was 338, a low value compared to the rest of the models, indicating that this model converged earlier and needed fewer rounds to reach the best performance, reducing the risk of overfitting and improving computational efficiency.

THRESHOLD SELECTION

In binary classification, the decision threshold is the value that establishes how the probabilities turn into discrete class labels. By default, a threshold of 0.5 is often used, meaning that points with a predicted probability greater than or equal to 0.5 are assigned to the positive class, while the rest are assigned to the negative class.

The choice of threshold is critical because it directly influences the trade-off between precision and recall, and consequently affects the resulting values of metrics such as the F1-score or Intersection over Union (IoU). A higher threshold reduces false positives, increasing precision but often lowering recall, while a lower threshold increases recall at the expense of precision.

After obtaining predicted probabilities for the validation set, a range of thresholds between 0.01 and 0.99 is tested, and for each one, evaluation metrics including F1-score, recall, precision, and IoU are computed to determine the best threshold.

The optimal threshold selected depends on the metric used, because different metrics have different optimal thresholds. The threshold that maximizes the F1-score may not be the same as the one that maximizes recall or IoU. This means that the optimal threshold needs to be chosen based on the metric that will be used to evaluate the final model, obtaining the best results for the desired metric.

EVALUATION METRICS

During training, the evaluation metric employed was the Area Under the Precision–Recall Curve (AUC–PR). This metric summarizes the trade-off between precision and recall across all possible threshold values. Unlike the more common ROC–AUC, which can sometimes provide overly optimistic results in the presence of imbalanced datasets, AUC–PR is more informative when the positive class represents a minority, because it quantifies the model's ability to maintain high precision and high recall.

For this project, the metric that was chosen for the evaluation of the model and the threshold selection is the F1-score. The F1-score is a widely used evaluation metric in binary classification tasks, particularly when class imbalance is present, as it defines the harmonic mean of precision and recall, which minimises false positives (precision) and increases the correctly identification of positive samples

(recall):

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$

For railway presence classification, it is important that the model identifies infrastructure points reliably, which requires maintaining a sufficiently high recall. On the other hand, excessively increasing recall without considering precision would result in a large number of false positives, which undermines the objective of filtering non-infrastructure data. If too many points from Class 0 are misclassified as infrastructure, the dataset remains contaminated with irrelevant elements, negating the benefits of the binary reduction.

By evaluating based on the F1-score, it balances the two requirements: ensuring a strong detection rate of infrastructure points while simultaneously effectively filtering non-infrastructure points. This balance is the best way of obtaining an accurate and clean dataset.

EVALUATION OF THE MODEL WITH THE VALIDATION SUBSET:

The **selected threshold** based on the F1-score for the Class 1 in the validation subset is ***0.37***. A higher threshold would have improved precision at the expense of recall, leading to a greater loss of infrastructure points, whereas the chosen value maintains a more equitable trade-off.

The evaluation of the model on the validation data has a **global accuracy** of ***0.8270*** and a **balanced accuracy** of ***0.8183***, indicating that the classifier performs consistently across both classes. The **ROC AUC** of ***0.8946*** confirms the model's ability to discriminate effectively between infrastructure and non-infrastructure points, while the **PR AUC** of ***0.8734*** confirms the balance between precision and recall for all the different thresholds.

Class	**Precision**	**Recall**	**F1-score**	**IoU**
Class 0	0.8516	0.8618	0.8567	0.7493
Class 1	0.7889	0.7748	0.7818	0.6418

Table 4.3: Class-specific evaluation metrics on the validation set.

For Class 0 (non-infrastructure), the model achieved high precision (0.8516)

and recall (0.8618), resulting in a F1-score of 0.8567 and an IoU of 0.7493, filtering irrelevant points effectively. However, the performance for Class 1 (infrastructure) was slightly lower, with precision = 0.7889, recall = 0.7748, and an F1-score of 0.7818, showing that some misclassifications persist, particularly false negatives, which reduce recall. Even with these misclassifications, the results are very good, filtering the non-infrastructure elements but also correctly identifying the infrastructure points.

True \ Predicted	**Class 0**	**Class 1**
Class 0	12,017,740	1,927,745
Class 1	2,093,460	7,203,531

Table 4.4: Confusion matrix on the validation set.

The confusion matrix shows that out of more than 14 million Class 0 points, only 1.9 million were misclassified as infrastructure, while for Class 1 points, approximately 2.1 million were incorrectly labeled as non-infrastructure.

LightGBM

The LightGBM model was trained in the Google Colab environment using an NVIDIA A100 GPU, which ensured efficient handling of the large-scale point cloud dataset. The model was implemented with the `LGBMClassifier` interface, and its hyperparameters were carefully selected to provide a trade-off between predictive accuracy, generalization, and computational efficiency.

The configuration of the model was defined as follows:

- ***objective='binary'***: Specifies that the task is a binary classification problem, returning the probability of each point belonging to the positive class.
- ***class_weight='balanced'***: Adjusts the weight of each class based on its frequency, compensating for the imbalance.
- ***learning_rate***: Regulates the contribution of each new tree to the ensemble

- ***max_ depth***: Maximum depth of the trees, prevents excessive complexity and reducing overfitting risk.
- ***n_ estimators***: Maximum number of boosting rounds
- ***random_ state=42***: Ensures reproducibility by fixing the random seed.
- ***n_ jobs=-1***: Uses all available CPU cores for parallel execution of non-GPU operations.
- ***num_ leaves***: Restricts the maximum number of terminal leaves per tree, controlling model complexity and balancing between capturing non-linear patterns and avoiding overfitting.
- ***device_ type='gpu'***: Enforces GPU acceleration for faster training on large datasets.
- ***min_ child_ samples***: Number of samples required to form a new leaf node, reducing the risk of very small data partitions.

As in XGBoost, a validation subset was used to monitor the model's performance on unseen data, but in the LightGBM, this parameter is added during the fitting of the model, not in the model creation.

- ***callbacks=[early_ stopping(stopping_ rounds=20)]***: If no improvement in the model was observed after 20 rounds, the training is interrupted, so the number of estimators for the model is:

$$n_{\text{final}} \leq n_{\text{estimators}}$$

HYPERPARAMETER SELECTION:

It is important to select the parameters to design and create the model, using different sets of them to see how variations affect the performance of the model:

Model	learning rate	depth	estimators	leaves	min child samples	Best iteration	Train Log Loss	Validation Log Loss
Model 1	0.10	7	200	20	–	138	0.231168	0.4175
Model 2	0.10	7	200	–	–	168	0.197668	0.4095
Model 3	0.10	9	200	–	200	200	0.174528	0.414523
Model 4	0.15	9	200	–	200	49	0.251793	0.446101
Model 5	0.07	9	200	–	200	200	0.199436	0.427404
Model 6	0.07	10	200	–	200	200	0.197803	0.415143

Table 4.5: Comparison of different LightGBM model configurations and their performance on training and validation sets.

For LightGBM, using binary log loss as the training metric includes certain limitations when selecting the final model. Although log loss provides valuable information about how well the predicted probabilities align with the true labels, it is not the metric used to evaluate the effectiveness of the classifier in this project.

This is the reason why the selection of the best LightGBM configuration cannot be based solely on the reported log loss values. Instead, the comparative results obtained on the validation set provide a more reliable criteria, as they directly reflect the model's capacity to generalize and to perform for new unseen data. Nevertheless, the log loss values remain useful as a preliminary indicator, offering an overview of which configurations may perform better or worse, but the decisive factor in the selection of the model will be the validation performance under the evaluation metric used in the project.

THRESHOLD SELECTION:

The LightGBM classifier generates probabilistic outputs that must be converted into binary predictions using a decision threshold. Rather than fixing this value at 0.5, the same function as in the XGBoost model was used: for each candidate threshold, the corresponding precision, recall, F1-score, and IoU were computed, and the optimal value was chosen after deciding the best evaluation metric for this project.

EVALUATION METRICS:

During the training of the LightGBM model, the objective was based on the binary log loss metric, also known as logistic loss or cross-entropy loss. This metric measures the difference between the predicted probabilities generated by the classifier and the true binary class labels:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\Big[y_i \log(p_i) + (1-y_i)\log(1-p_i)\Big]$$

where N is the number of samples, $y_i \in \{0,1\}$ is the true label of sample i, and p_i is the predicted probability that sample i belongs to the positive class.

This penalizes incorrect predictions in a non-linear manner: confident misclassifications, which consists on assigning a high probability to the wrong class, have a much higher penalty than less confident ones. As a result, the metric encourages the model to assign probabilities that are both accurate and well calibrated, unlike metrics based on hard class assignments like accuracy, log loss works directly on the probabilities of the model.

For LightGBM, binary log loss is particularly useful because each new tree is constructed to minimize the overall uncertainty of the model's predictions.

On the other hand, for the evaluation of the model the same metric used in XGBoost Will be maintained,making sure that both models are being evaluated using the same metrics, which is essential to compare the performance of both models.

EVALUATION OF THE MODEL WITH THE VALIDATION SET:

As explained before, the LightGBM best model needs to be selected comparing the F1-score metric for all the models created.

In order to do so, each model will have a different optimal threshold, which will also be selected along with the best model:

Model	Threshold	Train PR AUC	Validation PR AUC
MODEL 1	0.43	0.9595	0.8669
MODEL 2	0.39	0.9714	0.8746
MODEL 3	0.35	0.9777	0.8816
MODEL 4	0.44	0.9512	0.8506
MODEL 5	0.41	0.9710	0.8706
MODEL 6	0.39	0.9716	0.8744

Table 4.6: Threshold values and corresponding PR AUC scores for training and validation across different LightGBM models.

Among the models created, **Model 3** is the best performing LightGBM model for the railway presence classification task, because it achieved the best validation score (0.8816) while also maintaining the best training result (0.9777). In comparison, other models such as Model 2 and Model 6 achieved similar validation scores (0.8746 and 0.8744, respectively), but it was slightly worse than Model 3. Meanwhile, models like Model 4 and Model 5 had much lower validation results, proving that the model does not work well with new, unseen data.

The optimal threshold for this model is 0.35 which maximizes the F1-score for the class 1 (positive outcome).

Model	learning rate	depth	estimators	leaves	min child samples	Best iteration	Train Log Loss	Validation Log Loss
Model 3	0.10	9	200	–	200	200	0.174528	0.414523

Table 4.7: Best performing LightGBM model

The evaluation of this model on the validation set had a **global accuracy** of ***0.8368*** and a **balanced accuracy** of ***0.8228***, indicating that the classifier maintains a consistent behavior across both classes despite their imbalance. The **ROC AUC** of ***0.8993*** shows the model's ability to distinguish reliably between infrastructure and non-infrastructure points, while the **PR AUC** of ***0.8816*** reflects a strong balance between precision and recall for different thresholds.

Class	Precision	Recall	F1-score	IoU
Class 0	0.8442	0.8928	0.8678	0.7665
Class 1	0.8240	0.7528	0.7868	0.6485

Table 4.8: Class-specific evaluation metrics for the best model on the validation set.

For Class 0 (non-infrastructure), the model achieved strong results, with precision of 0.8442 and recall of 0.8928, obtaining an F1-score of 0.8678 and an IoU of 0.7665, confirming the model filters irrelevant elements. In contrast, the performance for Class 1 (infrastructure) was slightly lower, with precision = 0.8240, recall = 0.7528, and an F1-score of 0.7868, some false negatives persist, slightly reducing recall.

True \ Predicted	Class 0	Class 1
Class 0	12,450,154	1,495,331
Class 1	2,298,202	6,998,78

Table 4.9: Confusion matrix of Model 3 on the validation set.

The confusion matrix shows that, out of more than 13.9 million Class 0 points, approximately 1.5 million were misclassified as infrastructure, and for Class 1 points, about 2.3 million were incorrectly labeled as non-infrastructure. This confirms that while some misclassifications remain, the model provides a reliable classification between infrastructure and non-infrastructure elements.

4.3.2 Railway components segmentation: PointNet

The goal of this stage is to design and implement a deep learning architecture capable of segmenting the different components of the railway infrastructure at point level (see Figure 4.13). In this context, each point of the input cloud must be assigned to one of the predefined semantic classes, allowing the separation of structural elements such as tracks, masts, and overhead wires.

The PointNet model was selected as the backbone of the segmentation framework because it is a pioneering architecture for directly processing raw point clouds

without intermediate voxelization or projection. The choice of this type of model is also motivated by its efficiency and adaptability, unlike more complex hierarchical methods, it provides a balance between computational cost and segmentation accuracy, which is particularly relevant when processing large-scale railway scenes with millions of points.

The model implemented in this project is based on the original PointNet model[] but some changes have been made in order to create a better model for this specific tasks, using all of the information available in the database. First of all, we want to obtain all the points classified as part of the infrastructure from the previous model, as segment them as part of three possible categories:

- **Class 3:** Rails
- **Class 4:** Poles
- **Class 5:** Wires

MODEL INPUT:

The main difference between this model and the original PointNet lies in the definition of the input representation. In the original formulation, each point of the cloud is characterized solely by its spatial coordinates (x, y, z), but the model developed extends the representation by including additional per-point attributes: the RGB color channels (r, g, b) and the intensity value. This results in a better description of the scene, which is particularly useful in the railway context, where many components may share similar geometric profiles but differ in visual or reflectance properties.

The integration of these attributes required adapting the input for the model. For each scene, the information stored in the LiDAR dataset is combined into a single feature vector per point of the form:

$$\mathbf{p}_i = \big[x_i, y_i, z_i, r_i, g_i, b_i, I_i\big], \quad i = 1, \ldots, N$$

where N denotes the number of points in the scene and I_i corresponds to the intensity value of point i.

Another important difference with the original model is the number of points per scene N: in the original model it is a fixed number, but as was analyzed in the

data exploration stage, this database has different number of points per scene, so N can not be a fixed number. To work with this, the model is designed to operate in a fully point-wise manner, applying shared multilayer perceptrons to each point independently,so that the architecture remains invariant to the number of input points.

As a result of this preprocessing, each scene is represented as a tensor of dimension $(N, 7)$, where N is the number of points and 7 corresponds to the concatenated features described above. This flexible representation preserves all relevant information from the dataset while designing the model to process point clouds of different size without requiring additional processing.

INPUT TRANSFORM BLOCK:

Unlike the original PointNet, which operates on per-point coordinates (x, y, z), this model works on $(N, 7)$ containing spatial coordinates, color (r, g, b) and intensity.

The purpose of the this block is to learn a distribution of the points that reduces variations, like rotations around the track axis, which is why $\mathbf{T}$ needs to be exclusively applied to (x, y, z). Mixing color or intensity into this would use non-geometrical attributes to create the geometrical representation, causing the model to incorrectly detect the real geometry of each image.

In this part of the model, two groups of features are created:

$$\mathbf{X}_{\text{xyz}} = \mathbf{X}[:, 1{:}3] \in \mathbb{R}^{N\times 3}, \qquad \mathbf{X}_{\text{aux}} = \mathbf{X}[:, 4{:}7] \in \mathbb{R}^{N\times 4}.$$

A learnable 3×3 matrix $\mathbf{T}$, predicted by a T-Net, is then applied to the coordinates,

$$\mathbf{X}'_{\text{xyz}} \;=\; \mathbf{X}_{\text{xyz}}\,\mathbf{T} \;\in \mathbb{R}^{N\times 3},$$

and the modified coordinates are concatenated back with the untouched additional features to recover a scene tensor of shape $(N, 7)$:

$$\mathbf{X}' \;=\; \left[\mathbf{X}'_{\text{xyz}} \parallel \mathbf{X}_{\text{aux}}\right] \;\in \mathbb{R}^{N\times 7}.$$

This design preserves the complete $(N, 7)$ representation at the model level while maintaining geometrical steps based only on the coordinates of each point.

The T-Net is created for k=3 (see Appendix C) and the original coordinates are multiplied by this matrix to obtained the new outputs.

MLP(64,64): At this stage, the network processes the input tensor of size (N,7), which encodes the spatial coordinates together with the radiometric attributes (RGB and intensity). These features go through a shared multilayer perceptron (see Appendix C) with two fully connected layers with ReLU activations.

FEATURE TRANSFORM:

In the original PointNet formulation, a second transformation network (T-Net) is applied at the feature level, after projecting the input into a 64-dimensional space. However, in this project the feature transform block is deliberately omitted.

The reason behind is related to the additional features available in this model, which are non-geometrical, so this addition makes the result from the previous step not entirely geometrical. The resulting 64-dimensional space is no longer separated between purely geometric and non-geometric components, so once the features are combined, it is not possible to separate whether a given dimension has spatial structure, color, or intensity. Applying a transformation at this stage could change meaningful correlations,creating a worse model.

Although the absence of the feature transform block might lead to a slightly worse performance compared to the original PointNet, it is a necessary design choice for this project, that prioritizes the integration of all available attributes over the potential loss in model performance.

The original PointNet framework is capable of creating two different models: object classification, where a single label is predicted for the entire point cloud, and point-wise segmentation, where a class is assigned to each individual point. For this problem, only the segmentation branch is implemented, because the objective of this study is not to classify entire railway scenes into categories, but to identify and separate the structural components that form the infrastructure within each scene. For this purpose, point-wise segmentation is the only relevant output, since the classification of entire scenes into a single label would not provide meaningful information for detecting anomalies in cables.

OUTPUT SCORES: The result of the model is a tensor of size (N,3) for each scene where N corresponds to the number of points in the input file, which is a variable number depending on the file.

The dimension 3 corresponds to the number of target classes:

- **Class 3:** Rails
- **Class 4:** Poles
- **Class 5:** Wires

Each row of this matrix contains the raw output scores (logits) for the three classes associated with a single point. After applying a softmax activation, these scores are converted into class probabilities. hus, the element (i, j) of the tensor represents the probability that point i of the scene belongs to class $j \in \{\text{rails}, \text{poles}, \text{wires}\}$. The final predicted segmentation is obtained by assigning to each point the class with maximum probability.

OPTIMIZER SELECTION:

The training of the model was performed using the **Adam optimizer**, a widely adopted method in deep learning due to its ability to adapt the learning rate individually for each parameter. Adam combines the advantages of two optimization techniques: momentum and RMSProp. Specifically, it computes exponential moving averages of both the gradients (first moment estimate) and the squared gradients (second moment estimate). Formally, for a parameter θ_t at iteration t with gradient g_t, Adam maintains:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \quad v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2,$$

where m_t and v_t are the biased first and second moment estimates, and β_1, β_2 are exponential decay rates (commonly set to 0.9 and 0.999, respectively). Bias-corrected estimates $\hat{m}_t$ and $\hat{v}_t$ are then computed, and the parameters are updated as:

$$\theta_{t+1} = \theta_t - \eta \, \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon},$$

where η is the global learning rate and ϵ a small constant for numerical stability.

PARAMETERS:

- ***learning_rate = 0.001:*** This value was chosen based on the original PointNet network, it s small to create gradual and stable updates
- ***weight decay***=10^{-4}: L_2 regularization term added to the loss function, penalizing large parameter values, helps to prevent overfitting by encouraging simpler solutions.

CLASS BALANCE:

In the dataset, certain categories such as rails contain a very large number of points, whereas others, such as wires and poles, are far less represented, so to train the model it is necessary to balance this classes to increase the model's ability to learn the minority classes.

To create this balance, class-specific weights were added to the loss function based on the frequency of each class. For each class $c \in \{1, \ldots, m\}$, the weight was defined as:

$$w_c = \frac{\sum_{j=1}^{m} n_j}{m \cdot (n_c + \epsilon)},$$

where n_c is the number of points belonging to class c, m is the number of classes, and ϵ is a small constant introduced to avoid division by zero.

This assigns higher weights to underrepresented classes and smaller weights to majority classes, and introducing these weights into the cross-entropy loss, the contribution of each class is balanced, so that the model learns to recognize both frequent and rare railway components.

LOSS FUNCTION:

The loss function adopted for training is the **cross-entropy loss**. Cross-entropy measures the difference between the predicted probability distribution and the true labels. For a point i with true label y_i and predicted class probabilities $\hat{p}_{i,c}$, the loss is defined as:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N} \sum_{i=1}^{N} \log \hat{p}_{i,y_i},$$

where N is the number of points in the scene, y_i is the ground-truth class of point i, and $\hat{p}_{i,y_i}$ is the predicted probability assigned to that class, penalizing the model more heavily when it assigns low probability to the correct class.

In this model, cross-entropy loss is important because the it requires assigning exactly one semantic label to each point in the cloud, that combined with the class balancing weights w_c, the final loss used in training can be written as:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} w_{y_i} \log \hat{p}_{i,y_i}.$$

EARLY STOPPING:

The training process employed an early stopping strategy to prevent overfitting. After each epoch, the validation mean Intersection-over-Union (mIoU) was calculated, and the model parameters were saved if the validation mIoU surpassed the previous best value. If there is no improvement for 5 consecutive epochs and the validation mIoU is at least 0.85, the training is stopped, selecting as the final model the one with the highest validation miou instead of choosing the last epoch.

Once the model architecture was defined, the training was carried out using Google Colab, using an *NVIDIA A100 GPU*. The process was organized into training sessions of 100 epochs each, in order to monitor progress and evaluate improvements at regular intervals.

The first training, covering epochs 0 to 100, required approximately 15 minutes of computation, producing the following results:

Best epoch	**Train accuracy**	**Train mIoU**	**Validation mIoU**
90	0.8930	0.7754	0.7655

Table 4.10: Training and validation performance of the best model from epoch 0 to 100.

The following training covered from 100 to 200 epochs, also requiring 15 minutes and producing the following results for the best model:

Best epoch	Train accuracy	Train mIoU	Validation mIoU
146	0.8631	0.7239	0.7703

Table 4.11: Training and validation performance of the best model from epoch 100 to 200.

EVALUATION METRICS:

The criteria used to select the best model for the training is the **mean Intersection-over-Union (mIoU)** computed on the validation set. The IoU metric evaluates the common elements between the predicted segmentation and the true labels for a given class, defined as:

$$\text{IoU}_c = \frac{TP_c}{TP_c + FP_c + FN_c},$$

where TP_c (true positives) represents the number of points correctly predicted as class c, FP_c (false positives) the points incorrectly predicted as c, and FN_c (false negatives) the points of class c that were missed by the model. The **mean IoU (mIoU)** is obtained by averaging the IoU values across all classes:

$$\text{mIoU} = \frac{1}{m} \sum_{c=1}^{m} \text{IoU}_c,$$

with m being the number of classes. This metric is particularly suitable for segmentation tasks with imbalanced datasets, as it treats all classes equally regardless of their frequency.

In this project mIoU was chosen as the evaluation criteria because it provides a fair and comprehensive assessment of segmentation quality, so the best model of each training group of 100 epochs is the one that obtained the higher mIoU for the validation subset.

EVALUATION OF THE MODEL WITH THE TRAIN AND VALIDATION:

For each group of training epochs, the best final model had the following results:

For the first model (epoch 0 to 100), the validation subset achieved a **global accuracy** of ***0.8769*** and a **balanced accuracy** of ***0.8577***, and the **mean Intersection-over-Union (mIoU)** reached a value of ***0.7655***, which confirms that the model is capable of segmenting the infrastructure.

Class	Precision	Recall	F1-score	IoU
Class 3	0.9458	0.9992	0.9717	0.9450
Class 4	0.8455	0.6704	0.7478	0.5972
Class 5	0.8204	0.9034	0.8599	0.7543

Table 4.12: Class-specific evaluation metrics for the best model on the validation subset from epoch 0 to 100.

For **Class 3 (tracks)**, the model exhibits excellent results, with precision = 0.9458 and recall = 0.9992, obtaining an F1-score of 0.9717 and an IoU of 0.9450. This very high recall shows that almost all track points are correctly identified, while the slightly lower precision reflects a small number of false positives.

For **Class 4 (masts)**, the performance is more limited, with precision = 0.8455, recall = 0.6704, F1-score = 0.7478, and IoU = 0.5972, and gap between precision and recall reveals the presence of missed detections (false negatives).

For **Class 5 (cables)**, the model achieves a solid balance, with precision = 0.8204 and recall = 0.9034, leading to an F1-score of 0.8599 and an IoU of 0.7543. The high recall demonstrates the ability of the model to detect most cable points, although the lower precision indicates occasional confusion with other elements.

True \ Predicted	Class 3	Class 4	Class 5
Class 3	397,961	329	0
Class 4	22,818	188,574	69,881
Class 5	0	34,138	319,298

Table 4.13: Confusion matrix of the best model on the validation subset from epoch 0 to 100.

The confusion matrix shows that almost all Class 3 (tracks) points were correctly classified, with only 329 mislabeled as masts and none as cables. For Class

4 (masts), however, a considerable number of points were confused, with more than 22,000 predicted as tracks and nearly 70,000 predicted as cables, confirming this class as the most challenging. In contrast, Class 5 (cables) achieved strong performance, but over 34,000 points were misclassified as masts.

For the second model (epoch 100 to 200), the validation subset reached a **global accuracy** of ***0.8819*** and a **balanced accuracy** of ***0.8604***, with a **mean Intersection-over-Union (mIoU)** of ***0.7703***.

Class	**Precision**	**Recall**	**F1-score**	**IoU**
Class 3	0.9429	0.9980	0.9697	0.9412
Class 4	0.8988	0.6380	0.7463	0.5952
Class 5	0.8112	0.9451	0.8730	0.7746

Table 4.14: Class-specific evaluation metrics for the best model on the validation subset from epoch 100 to 200.

For **Class 3 (rails)**, the model delivers very strong results, achieving precision = 0.9429 and recall = 0.9980, which corresponds to an F1-score of 0.9697 and an IoU of 0.9412.

For **Class 4 (poles)**, the results are more modest: precision = 0.8988, recall = 0.6380, F1-score = 0.7463, and IoU = 0.5952. The low recall indicates that there are a lot of points from this class that are not correctly identified as poles.

For **Class 5 (wires)**, the model achieves a strong compromise, with precision = 0.8112 and recall = 0.9451, leading to an F1-score of 0.8730 and an IoU of 0.7746. The high value of recall shows that most of the points from this class are correctly identified as part of cables in the infrastructure, but the lower precision number shows there are points classified as part of class 5 when they do not belong in this category.

True \ Predicted	Class 3	Class 4	Class 5
Class 3	397,493	797	0
Class 4	24,058	179,460	77,755
Class 5	0	19,419	334,017

Table 4.15: Confusion matrix of the best model on the validation subset (epoch 100 to 200).

The confusion matrix shows that almost all Class 3 (rails) points are correctly classified, with only 797 mislabeled as poles and none confused with cables. For Class 4 (poles), misclassifications are more pronounced: over 24,000 points were predicted as rails and more than 77,000 as wires. In the case of Class 5 (wires), most points were correctly segmented, but around 19,000 were confused with masts.

4.3.3 Cable anomalies detection

The main goal of this project is to detect anomalies in the catenary systems of the railway infrastructure, making this part of the modelling stage (see Figure 4.13) the most important of the whole study.

The previous segmentation stage separated the cables (class 5) from the rest of the elements, making it possible to now detect anomalies in each of those cables, so the input of the anomaly detection stage corresponds to the point clouds previously segmented as cable elements by the PointNet-based model. The objective is to process these points and evaluate their geometric consistency, and create indicators to see if there are anomalies present in those cables.

It is important to emphasize that this analysis is exclusively conducted on the validation and test datasets. Training data are intentionally excluded from this process, since they have already been used to create and optimize the segmentation models in the previous stages. Using the validation and test data, the model is focusing on new, unseen data like it will happen with real cases.

CABLE SEPARATION:

In order to detect the anomalies in the cables, it is first necessary to separate

each cable from the complete set of points previously classified as *cable*. If all cable points were analyzed together, the detection process would be affected by the spatial distances that naturally exist between different cables. To avoid this, the cloud of points corresponding to the cable class is divided into different clusters that each represent a different cable, performing the detections on a cable-by-cable basis, rather than on the entire image simultaneously.

From a mathematical perspective, let the set of cable points obtained from segmentation be denoted as:

$$\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N\}, \quad \mathbf{p}_i \in \mathbb{R}^3 \tag{4.1}$$

where $\mathbf{p}_i = (x_i, y_i, z_i)$ represents the spatial coordinates of each point classified as cable. The goal is to partition $\mathcal{P}$ into K disjoint subsets, each corresponding to an individual cable:

$$\mathcal{P} = \bigcup_{k=1}^{K} \mathcal{C}_k, \quad \mathcal{C}_i \cap \mathcal{C}_j = \varnothing \;\; \forall i \neq j \tag{4.2}$$

To achieve this, a **density-based clustering algorithm (DBSCAN)** was applied to the 3D coordinates of the cable points. DBSCAN assigns a cluster label l_i to each point $\mathbf{p}_i$ based on two parameters:

- ε ***(eps)*:** The maximum distance between two points for considering them part of the same cluster
- $minPts$ ***(min_samples)*:** The minimum number of points required to define a new cluster.

Formally, DBSCAN groups points according to the following rule:

$$\text{Cluster}(\mathbf{p}_i) = \{\mathbf{p}_j \in \mathcal{P} \mid \|\mathbf{p}_i - \mathbf{p}_j\| \leq \varepsilon \text{ and density} \geq minPts\} \tag{4.3}$$

As a result, each cluster $\mathcal{C}_k$ corresponds to a different cable, while noise points (not belonging to any cluster) are labeled as -1. The points labeled as -1 could also be points misclassified from previous steps that are also considered noise if they do not belong to any of the created clusters.

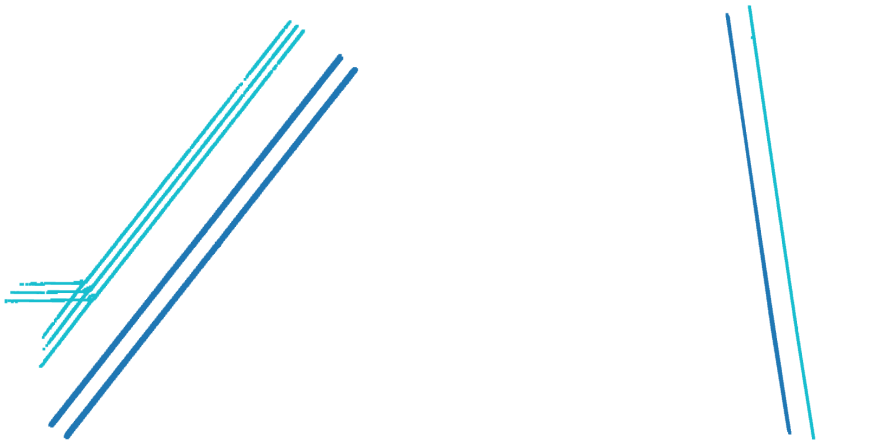
The selection of parameters ε and $minPts$ is the most important step, because a correct or incorrect separation of cables can have a very big impact in the final anomalies. These parameters were selected based on the separation results in the validation subset, and then they were applied to the test subset directly, without

any additional changes. This makes the model aplicable to other cases and to new unseen data without needing to redefine the parameters.

Parameter	Value
ε (eps)	0.045
Minimum samples (minPts)	10

Table 4.16: Selected parameters for cable separation using DBSCAN.

This preprocessing step provides the foundation for subsequent anomaly detection, where each cable is treated as an independent entity and its geometry can be evaluated without interference from neighboring cables.



(a) File *sncf_ 05* (b) File *hmls_ 01*

Figure 4.14: Cable separation

GLOBAL CABLE ANOMALIES

CABLE ALIGNMENT

The first type of global anomaly focuses on verifying whether the points of each cable are aligned. The motivation behind this analysis is to detect situations where the cable has suffered a rupture, a fracture, or a significant bending, which would deviate it from its expected straight geometry.

To evaluate the global alignment of the cable, the method is created based on the **Principal Component Analysis (PCA)** applied to the point cloud of each cable. Let the set of points for a given cable be denoted as:

$$\mathcal{C} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_M\}, \quad \mathbf{p}_i \in \mathbb{R}^3 \tag{4.4}$$

First, the point cloud is centered with respect to its mean:

$$\mathbf{p}'_i = \mathbf{p}_i - \bar{\mathbf{p}}, \quad \bar{\mathbf{p}} = \frac{1}{M}\sum_{i=1}^{M} \mathbf{p}_i \tag{4.5}$$

PCA is then applied to the centered points $\mathbf{p}'_i$, producing a set of orthogonal eigenvectors $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ associated with descending eigenvalues $\{\lambda_1, \lambda_2, \lambda_3\}$. The eigenvector corresponding to the largest eigenvalue, $\mathbf{v}_1$, is interpreted as the **main axis of the cable**:$mathbfe_{cable} = \mathbf{v}_1$ Each point is then projected onto this main axis: $pi_i = \mathbf{p}'_i \cdot \mathbf{e}_{cable}$

The variance of the projection explains how much the points are aligned with the main axis, returning a number from 0 to 1, 1 being completely explained by the main axis and 0 the opposite: $sigma^2_{axis} = \mathrm{Var}(\pi_i)$, while the overall variance of the cloud is obtained from the covariance matrix of the centered points:

$$\sigma^2_{total} = \mathrm{Tr}(\mathrm{Cov}(\mathbf{p}'_1, \mathbf{p}'_2, \ldots, \mathbf{p}'_M)) \tag{4.6}$$

The **explained variance ratio** is defined as:

$$R = \frac{\sigma^2_{axis}}{\sigma^2_{total}} \tag{4.7}$$

Finally, a cable is considered anomalous if the explained variance ratio R is lower than a predefined threshold τ, showing that they points do not have a clear alignment between them:

$$\text{Anomaly} = \begin{cases} 1, & \text{if } R < \tau \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

In this project, the threshold τ was empirically set to 0.90, meaning that if less than 90% of the variance of the cable points is explained by the principal axis, the cable is marked as anomalous.

Parameter	Value
Variance threshold (τ)	0.90

Table 4.17: Selected parameter for Cable Alignment anomaly detection.

CABLE INCLINATION

The second type of global anomaly focuses on the inclination of the cable with the reference system. This analysis wants to detect cases in which a cable has collapsed, is hanging abnormally, or does not follow the expected horizontal or vertical orientation.

From a geometric perspective, the alignment of a cable can be checked either with respect to a **dominant axis** or relative to a **reference plane**. The choice between these two approaches depends on the distribution of the principal axis of the cable in space. **Dominant axis condition:** Let $\mathbf{e}_{cable}$ be the main axis of the cable, obtained through PCA as in the previous anomaly:

$$\mathbf{e}_{cable} = (e_x, e_y, e_z), \quad \|\mathbf{e}_{cable}\| = 1 \tag{4.9}$$

A dominant axis exists if one component of $\mathbf{e}_{cable}$ is significantly larger (in absolute value) than the others. Formally, if

$$\frac{|e_{max}|}{|e_{second}|} \geq \rho \quad \text{and} \quad \frac{|e_{max}|}{|e_{third}|} \geq \rho \tag{4.10}$$

where $|e_{max}|$ is the largest component of $\mathbf{e}_{cable}$, $|e_{second}|$ and $|e_{third}|$ are the other two components, and ρ is a dominance ratio, then the cable is considered to follow primarily that axis. For this Project, the dominance ratio chosen was $\rho = 10$, so that the main axis is largely influenced by one only component.

The inclination angle of the cable with respect to this dominant axis is:

$$\theta = \arccos\left(|\mathbf{e}_{cable} \cdot \mathbf{e}_{ref}|\right) \tag{4.11}$$

where $\mathbf{e}_{ref}$ is the unit vector of the dominant axis. If θ exceeds a predefined threshold τ, the cable is marked as anomalously.

The dominance ratio chosen was $\rho = 10$, so that the main axis is largely influenced by one only component,and the threshold for anomaly in inclination was set to $\tau = 5°$

Planar condition: If no single component of $\mathbf{e}_{cable}$ is dominant, the cable can be better described as part of a plane. In this case, the two largest components of $\mathbf{e}_{cable}$ indicate the plane of orientation, while the smallest component is negligible. Let $\mathbf{n}_{plane}$ be the normal vector of the corresponding plane, the angle between $\mathbf{e}_{cable}$ and $\mathbf{n}_{plane}$ is given by:

$$\phi = \arccos\left(|\mathbf{e}_{cable} \cdot \mathbf{n}_{plane}|\right) \tag{4.12}$$

and the inclination with respect to the plane is then:

$$\theta_{plane} = 90° - \phi \tag{4.13}$$

If $\theta_{plane} > \tau$, the cable is again considered anomalous.

Every cable can be analyzed with respect to a plane, since its main axis always belongs to one of the coordinate planes, however, when one axis component of $\mathbf{e}_{cable}$ is much larger than the others, the cable geometry is better captured by comparing it to the corresponding dominant axis. Although this approach usually has slightly higher inclination angles than the planar approach, it is deliberately more conservative.

Parameter	Value
Dominant axis ratio (ρ)	10
Minimum plane ratio	0.5
Inclination threshold (τ)	5°

Table 4.18: Selected parameters for Cable Inclination anomaly detection.

POINT PER POINT ANOMALIES

POINT DEVIATION

The first point-level anomaly wants to identify local deviations of individual points from the expected cable trajectory. In practice, a cable should follow a continuous curve that can locally be approximated by a straight line. When certain points are located significantly away from this line, they may indicate irregular situations such as a cable being on top of tree branches, so the detection of this case is also very important to do a further inspection of the cable. Let $\mathcal{C} = \{\mathbf{p}_i\}_{i=1}^{M}$ be the set of points belonging to a cable, with $\mathbf{p}_i \in \mathbb{R}^3$. The global orientation of the cable is estimated through PCA, obtaining the unit direction vector of its principal axis $\mathbf{e}_{cable}$. Once the cable centroid is computed as

$$\bar{\mathbf{p}} = \frac{1}{M} \sum_{i=1}^{M} \mathbf{p}_i, \tag{4.14}$$

each point $\mathbf{p}_i$ can be orthogonally projected onto the axis defined by $(\bar{\mathbf{p}}, \mathbf{e}_{cable})$ according to

$$\widehat{\mathbf{p}}_i = \bar{\mathbf{p}} + \big((\mathbf{p}_i - \bar{\mathbf{p}}) \cdot \mathbf{e}_{cable}\big)\, \mathbf{e}_{cable}. \tag{4.15}$$

The distance between the original point and its projection is then defined as

$$d_i = \|\mathbf{p}_i - \widehat{\mathbf{p}}_i\|_2, \tag{4.16}$$

which measures the orthogonal deviation of each point with respect to the cable's main trajectory. A point is considered anomalous whenever its deviation exceeds a predefined threshold δ, that is,

$$\mathbf{p}_i \text{ is anomalous if } \; d_i > \delta. \tag{4.17}$$

For this project, δ was chosen equal to 0.05, allowing for small deviations but not big in reference to the main axis.

Parameter	Value
Deviation threshold (δ)	0.05

Table 4.19: Selected parameter for Point-Point Deviation anomaly detection.

SEGMENTS CHANGE IN DIRECTION

Another point-level anomaly is related to the detection of abrupt changes in the direction of the cable. Under normal conditions, the orientation of the cable should remain relatively stable when moving along consecutive portions of points, however, if two contiguous sections form a large angle between them, this may reveal structural problems such as a cable that has fallen, has been twisted, or is hanging irregularly. Detecting this is important to detect when a cable needs to be further inspectioned.

The method begins by dividing the sequence of points belonging to a cable into consecutive segments of fixed size S. Formally, given a set of ordered points

$$\mathcal{C} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N\}, \quad \mathbf{p}_i \in \mathbb{R}^3,$$

the function `split_segments` groups them into subsets

$$\mathcal{C} = \bigcup_{k=1}^{K} \mathcal{S}_k, \qquad \mathcal{S}_k = \{\mathbf{p}_{i_k}, \ldots, \mathbf{p}_{j_k}\}, \quad |\mathcal{S}_k| \approx S,$$

where K denotes the total number of segments.

For each segment $\mathcal{S}_k$, its dominant orientation vector is extracted using Principal Component Analysis (PCA). Let $\bar{\mathbf{p}}_k$ be the centroid of $\mathcal{S}_k$. The first principal component is computed by solving

$$\mathbf{e}_k = \arg\max_{\|\mathbf{u}\|=1} \mathbf{u}^\top \left(\frac{1}{|\mathcal{S}_k|} \sum_{\mathbf{p}_i \in \mathcal{S}_k} (\mathbf{p}_i - \bar{\mathbf{p}}_k)(\mathbf{p}_i - \bar{\mathbf{p}}_k)^\top \right) \mathbf{u},$$

yielding the unit vector $\mathbf{e}_k$ that best represents the direction of the segment.

To detect anomalies, the angle between consecutive segment directions is calculated. For two consecutive segments $\mathcal{S}_k$ and $\mathcal{S}_{k+1}$, the angle is given by

$$\theta_k = \arccos\left(|\mathbf{e}_k \cdot \mathbf{e}_{k+1}|\right).$$

This angle quantifies the directional change between contiguous segments. If the angle exceeds a predefined threshold τ, then the set of points belonging to both segments is flagged as anomalous:

$$\text{Anomaly at } k \quad \Longleftrightarrow \quad \theta_k > \tau.$$

For this model, the parameter S was empirically set to 1000 points, ensuring that each segment was large enough to provide a stable estimate of its orientation.The

other parameter threshold, τ was fixed at 30°, balancing the need to detect significant misalignments without over-penalizing minor natural fluctuations.

The algorithm returns the angular deviation between consecutive segments and the indices of the points corresponding to anomalous regions, which can later be visualized or quantified in the evaluation phase.

Parameter	Value
Segment size (S)	1000 points
Angle threshold (τ)	30°

Table 4.20: Selected parameters for Segment Change Direction anomaly detection.

SPACE BETWEEN POINTS

The final type of point-level anomaly is related to the detection of unusually large gaps between consecutive points along the cable. If the separation between consecutive points is larger than expected, this may indicate that the cable has been broken or displaced.

However, it must also be noted that due to the way in which elements are classified, cables are often supported by poles whose points do not belong to the cable class. As a result, when a pole intersects a cable, the gap caused by the missing points from the pole's region can lead to the false detection of an anomaly. This introduces an inherent limitation in the model, which must be considered when interpreting the results.

Formally, the detection process begins by estimating the main axis $\mathbf{e}_{cable}$ of the cable via PCA, as described in previous sections. Each point $\mathbf{p}_i$ is projected onto this axis:

$$\pi_i = (\mathbf{p}_i - \bar{\mathbf{p}}) \cdot \mathbf{e}_{cable}, \tag{4.18}$$

where $\bar{\mathbf{p}}$ is the centroid of the cable. The projections π_i are then sorted in ascending order, producing a one-dimensional sequence that preserves the longitudinal order of the cable.

The differences between consecutive projections quantify the spacing between

points along the axis:

$$\Delta_j = \pi_{j+1} - \pi_j, \quad j = 1, \ldots, M-1. \tag{4.19}$$

If any Δ_j exceeds a predefined threshold δ, the gap between the corresponding points is flagged as anomalous:

$$\text{Anomaly at gap } j \quad \Longleftrightarrow \quad \Delta_j > \delta. \tag{4.20}$$

The threshold was set to $\delta = 0.02$, as it represents a reasonable upper bound on the expected spacing between consecutive points. By fixing this parameter after validation, the same value was consistently applied to the test dataset, ensuring comparability and avoiding overfitting to specific cases. The output of the detector consists of the indices of the points that define each anomalous gap, together with the measured gap size.

Parameter	**Value**
Gap threshold (δ)	0.02

Table 4.21: Selected parameter for Space Between Points anomaly detection.

4.4 Testing and results

4.4.1 Railway presence classification

XGBoost

For the classification of railway presence, the first model created and evaluated is the XGBoost. The configuration and hyperparameters of the model are already selected during the modelling stage, some based on the training and others on the validation, like the F1 score, aiming to obtain the maximum F1 score possible.

Model	**depth**	**learning rate**	**estimators**	**Threshold**
MODEL 4	4	0.10	500	0.37

Table 4.22: Final model with all the parameters

The evaluation on the test dataset resulted in a **global accuracy** of ***0.7579*** and a **balanced accuracy** of ***0.7304***, while the **ROC AUC** and **PR AUC** achieved values of ***0.8399*** and ***0.7962***, respectively. These scores confirm a reasonably good discriminative capability of the model, although lower than the values obtained on the validation set (accuracy = 0.8270, balanced accuracy = 0.8183, ROC AUC = 0.8946, PR AUC = 0.8734).

Class	**Precision**	**Recall**	**F1-score**	**IoU**
Class 0	0.7618	0.8678	0.8114	0.6826
Class 1	0.7494	0.5931	0.6621	0.4949

Table 4.23: Class-specific evaluation metrics on the test set for XGBoost model

At the class level, the performance for Class 0 (non-railway presence) remains stable, with a precision of 0.7618, recall of 0.8678, F1-score of 0.8114, and IoU of 0.6826. However, there has been a great decrease in performance for Class 1 (railway presence), where the F1-score falls to 0.6621 compared to 0.7818 during the validation of the model. This is due to a decrease in the recall of the class (from 0.7748 to 0.5931), confirming that the model is worse at identifying the true positives in unseen data.

However, a worse result can be observed for Class 1 (railway presence), where the F1-score drops to 0.6621 compared to 0.7818 during validation. This reduction is mainly due to a recall decrease (from 0.7748 to 0.5931), which indicates that the model has greater difficulty in correctly identifying true positive railway samples when facing previously unseen data.

True \ Predicted	**Class 0**	**Class 1**
Class 0	14,980,039	2,209,291
Class 1	4,532,463	6,605,757

Table 4.24: Confusion matrix on the test set for XGBoost model

The confusion matrix provides additional information, out of the total Class 0 samples, 14,980,039 were correctly classified, while 2,209,291 were misclassified as Class 1. For Class 1, the model correctly identified 6,605,757 instances but incorrectly assigned 4,532,463 samples to Class 0. This imbalance in misclassification

explains the drop in recall for Class 1 and the corresponding decline in its F1-score. Even with this, the results of the test set are consistent and offer good results for both classes.

The difference between validation and test performance suggests a certain degree of overfitting to the validation distribution. This could happen when choosing the optimal threshold because the one that maximizes validation scores is selected and it is then fixed to that value for the test set. Consequently, this choice may not be optimal for the test distribution, which could explain the difference. It could also happen because of the variations in the distribution of test samples compared to training and validation.

LightGBM

The next model created for the railway presence classification is the LightGBM model, also using the same hyperparameters and threshold strategy defined in the modeling pase, which has the following parameters and threshold:

Model	**learning rate**	**depth**	**estimators**	**leaves**	**min child samples**	**Threshold**
Model 3	0.10	9	200	–	200	0.39

Table 4.25: Final LightGBM model and threshold applied

The evaluation on the independent test data had a **global accuracy** of ***0.7618*** and a **balanced accuracy** of ***0.7298*** and achieved a **ROC AUC** of ***0.8300*** and a **PR AUC** of ***0.7875***. Although these values remain consistent with a good level of performance, they are lower than the results obtained during validation, which indicates a worse performance when faced with unseen data.

Class	Precision	Recall	F1-score	IoU
Class 0	0.7562	0.8901	0.8177	0.6916
Class 1	0.7755	0.5694	0.6567	0.4889

Table 4.26: Class-specific evaluation metrics on the test set for the LightGBM model

At the class-specific level, it shows a consistent behavior for Class 0 (non-railway presence), reaching a precision of 0.7562, recall of 0.8901, F1-score of 0.8177, and IoU of 0.6916. In contrast, the performance on Class 1 (railway presence) decreases significantly, with its F1-score dropping to 0.6567 compared to 0.7818 in the validation phase, as in XGBoost caused by a reduction in recall (from 0.7748 to 0.5694).

True \ Predicted	Class 0	Class 1
Class 0	14,871,489	1,835,841
Class 1	4,795,617	6,342,603

Table 4.27: Confusion matrix on the test set for LightGBM model

The confusion matrix further illustrates these results. From the Class 0 samples, 14,871,489 were correctly classified, while 1,835,841 were wrongly predicted as Class 1. For Class 1, the model successfully identified 6,342,603 instances, but 4,795,617 samples were misclassified as Class 0. This large number of false negatives for Class 1 is directly reflected in its lower recall and explains the drop in the F1-score observed on the test set.

As in the XGBoost model, there is a big difference between the test and validation results, which could also be explained by the variations of the class distribution and the selection of the threshold based on the validation results, maximizing the validation F1 score, but without having the same impact on the test data.

4.4.2 Railway components segmentation: PointNet

For the railway component segmentation, as explained in the previous stage, a PointNet model was created. For the training and validation, groups of 100 epochs were used to obtain the best final model possible.

As part of the control of the creation and design of the model, each of these groups is also evaluated in the test results, in order to see the impact of increasing the number of epochs.

For epoch 0 to 100:

The best model obtained is

Best epoch	Train accuracy	Train mIoU	Validation mIoU
90	0.8930	0.7754	0.7655

Table 4.28: Training and validation performance of the best model from epoch 0 to 100.

The evaluation of the PointNet model over the first 100 epochs on the test dataset produced a **global accuracy** of ***0.8687***, a **balanced accuracy** of ***0.8347***, and a **mean IoU** of ***0.7344***. These results, although slightly below the validation performance (accuracy = 0.8769, balanced accuracy = 0.8577, mIoU = 0.7655), are great results for a segmentation model.

Class	Precision	Recall	F1-score	IoU
Class 3	0.9564	0.9918	0.9738	0.9489
Class 4	0.8119	0.6237	0.7055	0.5450
Class 5	0.7784	0.8887	0.8299	0.7093

Table 4.29: Class-specific evaluation metrics for the PointNet model on the test set from epoch 0 to 100.

At the class-specific level, the model shows very strong results for Class 3 (rails), with precision of 0.9564, recall of 0.9918, F1-score of 0.9738, and IoU of 0.9489, which remain close to the validation values, indicating that the network

identifies rail points with high reliability. Class 5 (cables) also has a very good performance, achieving F1 = 0.8299 and IoU = 0.7093, although both values are slightly lower compared to validation (F1 = 0.8599, IoU = 0.7543). The worst performing is Class 4 (posts), as occurred with the validation subset, where the F1-score drops to 0.7055 and the mIoU to 0.5450, compared to 0.7478 and 0.5972 during validation.

True \ Predicted	**Class 3**	**Class 4**	**Class 5**
Class 3	501,725	4,141	0
Class 4	22,866	175,091	82,767
Class 5	0	36,422	290,810

Table 4.30: Confusion matrix of the PointNet model on the test set from epoch 0 to 100.

The confusion matrix shows that, for Class 3, the majority of samples (501,725) were correctly classified, and in Class 5, the model correctly identified 290,810 instances, while 36,422 were misclassified as Class 4. The worst performing as expected is Class 4, where 82,767 samples were incorrectly labeled as Class 5, which explains the reduced recall and lower F1-score for this class.

For this model, different to the previous models, the results for the validation and test subsets are very similar, which means that the model performs very good with new, unseen data, and it is a model that can be used for segmenting effectively new images.

For epoch 100 to 200:

The best model created in this group is:

Best epoch	**Train accuracy**	**Train mIoU**	**Validation mIoU**
146	0.8631	0.7239	0.7703

Table 4.31: Training and validation performance of the best model from epoch 100 to 200.

The evaluation of the PointNet model in this of epochs (100–200) on the test dataset had a **global accuracy** of ***0.8242***, a **balanced accuracy** of ***0.8001***, and a **mean IoU** of ***0.6782***. These values, although they are good, show a much worse performance compared to the validation subset where the accuracy was 0.8819, balanced accuracy = 0.8604, and mIoU = 0.7703.

Class	Precision	Recall	F1-score	IoU
Class 3	0.9585	0.9070	0.9320	0.8727
Class 4	0.6570	0.6333	0.6449	0.4759
Class 5	0.7721	0.8600	0.8137	0.6859

Table 4.32: Class-specific evaluation metrics for the PointNet model on the test set from epoch 100 to 200.

At the class level, Class 3 remains the best performing, with an F1-score of 0.9320 and an IoU of 0.8727, but still dropped compared to validation (F1 = 0.9697, mIoU = 0.9412), mainly for the decrease in recall (0.9070 vs 0.9980). Class 5 maintains acceptable performance (F1 = 0.8137, mIoU = 0.6859), although these are again worse than validation (F1 = 0.8730, mIoU = 0.7746). The largest decline and worse performing class is Class 4, where the F1-score falls to 0.6449 and mIoU to 0.4759, compared to 0.7463 and 0.5952 during validation.

True \ Predicted	Class 3	Class 4	Class 5
Class 3	458,836	47,030	0
Class 4	19,884	177,787	83,053
Class 5	0	45,802	281,430

Table 4.33: Confusion matrix of the PointNet model on the test set from epoch 100 to 200.

As for the previous group of epochs, the confusion matrix shows that most of the Class 3 points were correctly classified, although more than 47,000 are misassigned as Class 4, explaining the reduction in recall, and for Class 5, most samples are predicted correctly (281,430), but 45,802 are confused with Class 4. Once again, the most worst case is Class 4, where a large number of samples (83,053) are wrongly labeled as Class 5, which significantly affects its recall and overall F1-score.

In contrast to the first 100 epochs, where test and validation results were almost identical, the interval between 100 and 200 epochs shows a more pronounced divergence. This suggests that the model may be starting to overfit the training distribution, losing some of its ability to generalize. While PointNet continues to perform robustly for Classes 3 and 5, the systematic weakness in segmenting posts (Class 4) becomes more evident in this phase, pointing to the need for further refinements or complementary strategies to improve its detection in unseen data.

COMPARISON BETWEEN GROUPS OF EPOCHS:

When comparing the two training intervals of PointNet, it shows that the model trained during the first 100 epochs provides a more reliable balance between validation and test performance. Although the second interval (100–200 epochs) slightly improves validation metrics, the test results are much lower than the validation scores and even worse than the test results for the previous training Interval.

The difference between validation and test results is bigger in the second interval, decreasing a lot in recall and segmentation quality, especially for posts, but by contrast, the model from epochs 0–100 is much more consistent between the two subsets, generalizing better for unseen data.

The reason behind this poorly performance for the test subset in the second interval may be caused by the overfitting of the train model, changing the weights to create a better model for the train set but having a much worse results with unseen data. On the contrary, the first interval of epochs, has a worse (but still good) results for the training model but can segment unseen data with higher accuracy. Therefore, the model trained within the first 100 epochs is the best choice, as it achieves high performance for unseen data.

4.4.3 Cable anomalies detection

For each of the anomaly detections described in the previous stage, the results are presented using two different perspectives that combine both quantitative and qualitative results. This approach makes the analysis not only reliable on numerical results, but is also supported by visual inspection of the cable geometries.

From a **quantitative perspective**, the outcomes of each detection model are summarized in a dataframe, stored for the combination of all the test and validation files. Each dataframe provides the numerical values associated with the detection, such as the alignment variance, inclination angle, deviation distances, or gap sizes, depending on the anomaly under study. In addition, a binary flag is included to explicitly indicate whether the corresponding cable (or set of points) has been classified as anomalous.

From a **qualitative perspective**, the point clouds of the original cables are visualized with a color code that reflects the detection result. This visual inspection allows us to directly validate the anomalies detected by the numerical analysis and to better understand their spatial context. The color code is the following:

- **Green**: The cable (or points) were analyzed and no anomaly of that type was detected.
- **Red**: An anomaly of the considered type was detected.

Depending on the nature of each anomaly, the visualization is applied either at the **global cable level** or at the **point level**.

GLOBAL CABLE ANOMALIES

CABLE ALIGNMENT

The output of this model is the structured dataframe, stored in both `pickle` and `csv` formats, where each row contains the information of an individual cable extracted from the dataset. The dataframe includes the file of origin, the cable identifier, the number of points composing the cable, the estimated principal axis, the explained variance associated with this axis, and a boolean flag indicating if it is anomalous or not.

File	Cable	Points	Invariance in Axis	Anomaly
sncf_05_norm_filtered.npz	0	88130	0.9990	False
sncf_05_norm_filtered.npz	1	12878	0.9975	False
sncf_05_norm_filtered.npz	2	80942	0.9991	False

Table 4.34: Example of the dataframe output for the file *sncf_05*.

File	Cable	Points	Invariance in Axis	Anomaly
hmls_01_norm_filtered.npz	0	11992	0.9991	False
hmls_01_norm_filtered.npz	1	4696	0.9996	False

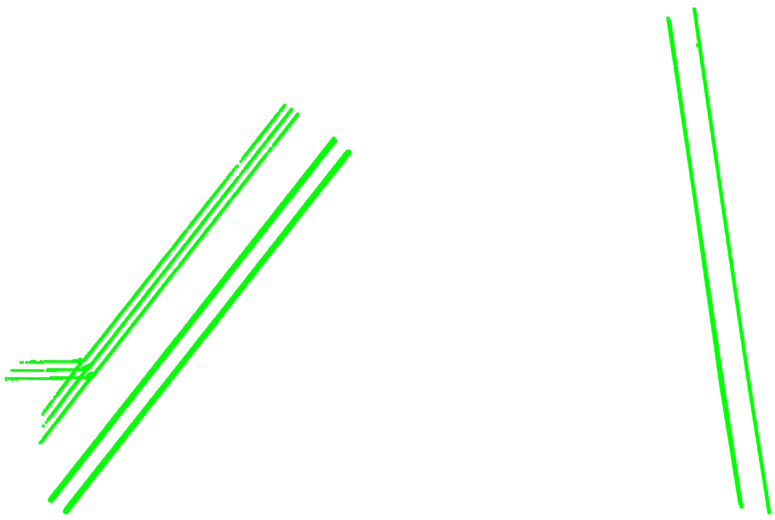
Table 4.35: Example of the dataframe output for the file *hmls_01*.

The numerical results provide a first **quantitative insight** into the anomalies in each cable. The column *Invariance in Axis* reflects the proportion of the total variance that is captured by the first principal component, high values (close to 1.0) indicate that the cable is well aligned with a dominant axis, while low values would suggest possible fractures, bends, or structural inconsistencies. The anomaly flag is directly derived from this measure, comparing it against the predefined threshold in the modelling stage.

Additionally, numerical evaluation is complemented with a **qualitative analysis** based on visualization. For each cable, the corresponding point cloud was displayed using a color code previously defined, allowing to visually assess whether the cable maintains a coherent alignment.

CABLE INCLINATION

The second global cable anomaly focuses on the inclination of the cables with respect to their expected orientation in space. The outputs of this detector are organized in a dataframe, where each row corresponds to a single cable. The recorded values include the file of origin, the cable identifier, the number of points, the estimated principal axis, the orientation criterion adopted (dominant axis or



(a) File *sncf_05* (b) File *hmls_01*

Figure 4.15: Cable alignment

plane), the measured angle of inclination, and a boolean flag indicating whether the anomaly threshold was exceeded.

File	Cable	Points	Angle of Inclination	Anomaly
sncf_05_norm_filtered.npz	0	122817	0.0261°	False
sncf_05_norm_filtered.npz	1	9455	1.1420°	False

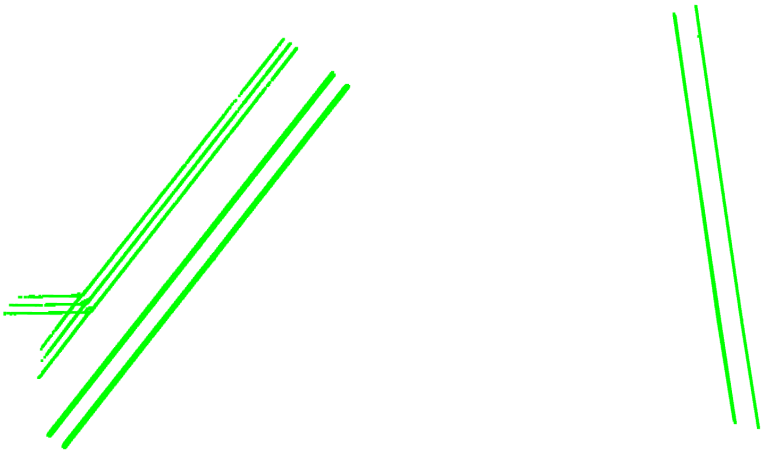
Table 4.36: Example of the dataframe output for `sncf_05`.

File	Cable	Points	Angle of Inclination	Anomaly
hmls_01_norm_filtered.npz	0	11992	0.2906°	False
hmls_01_norm_filtered.npz	1	4696	0.9172°	False

Table 4.37: Example of the dataframe output for `hmls_01`.

Numerical results show that the majority of cables are assigned to a planar orientation, with inclination angles consistently below the predefined threshold of 5°. The detection mechanism distinguishes between cables predominantly aligned

with one of the coordinate axes and those in a plane. In practice, all cables could be projected onto a plane; however, whenever a strong dominant axis is identified, the inclination is computed with respect to that axis to adopt a more conservative criterion. As in the previous anomaly detector, quantitative results are complemented by **visualization**. Each cable uses the predefined color code, with green indicating normal orientation and red marking cases that exceed the inclination threshold.



(a) File *sncf_ 05* (b) File *hmls_ 01*

Figure 4.16: Cable inclination

POINT PER POINT ANOMALIES

POINT PER POINT DEVIATION

The first point-level anomaly detector focuses on local deviations of individual points with respect to the main trajectory of the cable. The results of this analysis are stored in a dataframe, containing the mean orthogonal deviation of the points, the maximum deviation observed, the standard deviation of the deviations, the number of points exceeding the predefined threshold ($\delta = 0.05$), and the indices of such anomalous points.

File	Cable	Points	Mean Deviation	Max Deviation	Points over Threshold
sncf_05	0	122817	0.0121	0.0284	0
sncf_05	1	9455	0.0186	0.1489	238

Table 4.38: Example of the dataframe output for *sncf_05*.

File	Cable	Points	Mean Deviation	Max Deviation	Points over Threshold
hmls_01	0	11992	0.0122	0.0295	0
hmls_01	1	4696	0.0045	0.0192	0

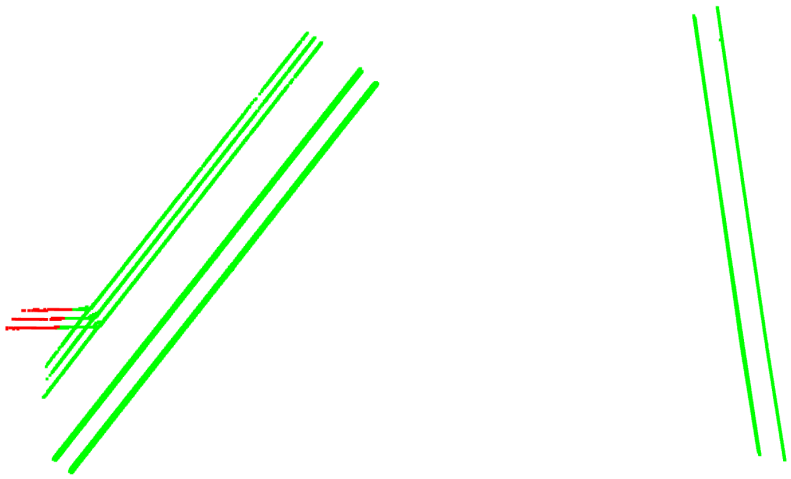
Table 4.39: Example of the dataframe output for *hmls_01*.

The **numerical results** indicate that the vast majority of points in the analyzed cables exhibit very small deviations from their projected trajectory, with mean deviations on the order of 10^{-2} and maximum deviations rarely exceeding 0.03.

As with previous detectors, these quantitative results were complemented by **visualizations** of the cables using the color code defined earlier. In this case, the absence of anomalies translated into cables being displayed in green, with no red-highlighted points. This dual representation reinforces the numerical evidence by visually confirming that no local irregularities such as detached points or spurious clusters are present in the tested samples.

SEGMENT CHANGE IN DIRECTION

The anomaly evaluates the consistency of the cable's orientation by analyzing the direction of consecutive segments. The results of this detector are stored in a dataframe, where each row corresponds to a pair of consecutive segments of a cable. The dataframe records the file of origin, the cable identifier, the number of points, the indices of the two segments under comparison, the computed angle between their dominant axes, and a flag indicating whether this angle exceeds the threshold of 30°.



(a) File *sncf_05* (b) File *hmls_01*

Figure 4.17: Point per point deviation

File	Cable	Points	First Segment	Second Segment	Angle (°)	Anomaly
sncf_05	0	122817	(0, 2000)	(2000, 4000)	37.74	True
sncf_05	0	122817	(2000, 4000)	(4000, 6000)	0.88	False
sncf_05	0	122817	(4000, 6000)	(6000, 8000)	2.47	False
sncf_05	0	122817	(6000, 8000)	(8000, 10000)	0.03	False
sncf_05	0	122817	(8000, 10000)	(10000, 12000)	0.95	False

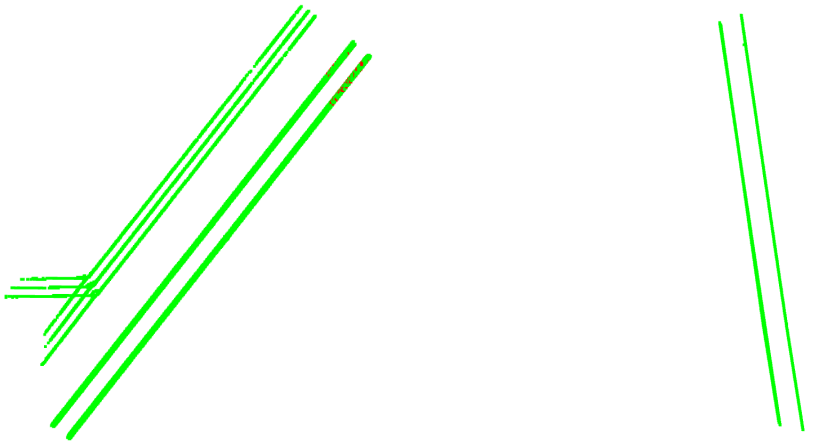
Table 4.40: Example of the dataframe output for *sncf_05*.

File	Cable	Points	First Segment	Second Segment	Angle (°)	Anomaly
hmls_01	0	11992	(0, 2000)	(2000, 4000)	1.74	False
hmls_01	0	11992	(2000, 4000)	(4000, 6000)	0.46	False
hmls_01	0	11992	(4000, 6000)	(6000, 8000)	0.11	False
hmls_01	0	11992	(6000, 8000)	(8000, 10000)	1.00	False
hmls_01	0	11992	(8000, 10000)	(10000, 11992)	0.12	False

Table 4.41: Example of the dataframe output for *hmls_01*.

The **numerical analysis** shows that most of the angles between consecutive segments remain below the threshold but there are some exceptions

As with the previous detectors, numerical analysis was complemented with **visualizations**. Cables were represented using the color code: green segments represent stable orientation, while red highlights abrupt directional changes.



(a) File *sncf_05* (b) File *hmls_01*

Figure 4.18: Point per point deviation

DISTANCE BETWEEN POINTS

The detection of anomalies based on the distance between consecutive points produced a dataframe containing, for each cable, the indices of the two points defining the gap, the computed distance, and if the gap exceeded the predefined threshold.

File	Cable	Points	Start Point	End Point	Gap Size
hmls_14_norm_filtered.npz	1	10462	4876	4877	0.0243

Table 4.42: Example of the dataframe output for the distance between points anomaly detection

This **quantitative analysis** shows a detailed desription of each anomaly for

each file, but only has information for those considered as anomalous, to reduce the size of the complete dataframe.

In addition to the numerical evidence provided in the dataframe, **qualitative validation** was performed through visual inspection. Cables without anomalous gaps are shown in green, while cables with one or more anomalous points, have those points in red.

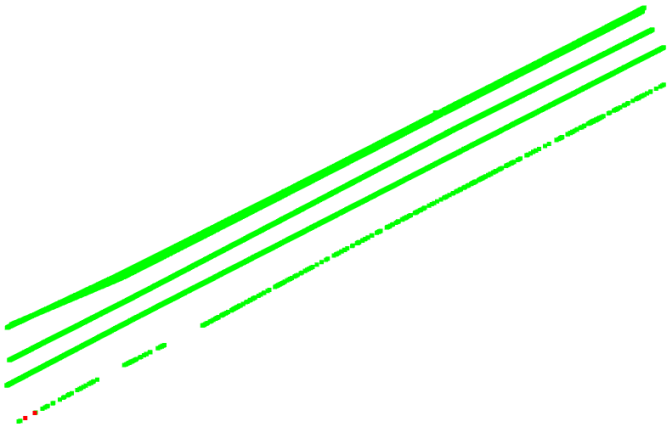


Figure 4.19: Space between points anomaly visualization

Chapter 5

Conclusions

This project has presented a comprehensive exploration of machine learning and deep learning models applied to the detection of anomalies in railway infrastructures. Through the combination of segmentation techniques based on PointNet, classification models with XGBoost, and a customed anomaly detection model, a complete end-to-end model has been developed to analyze LiDAR point cloud data. The experimentation across training, validation, and test datasets have shown the ability of these models to accurately segment structural elements, classify relevant infrastructure components, and identify global and local anomalies in overhead cables.

The potential application of these models is considerable. The results obtained how the model created manages complex data and detects problems in the structure analyzed. This project is mainly focused on the railway sector but the impact it has can be extended to other industries that also require structural monitoring and have large-scale data.

Potential of proposed model for anomaly detection

The proposed model, combining PointNet-based semantic segmentation, tree-boosting classification, and a custom anomaly detection model, demonstrates strong potential for reliable assessment of overhead cable integrity in railway environments. By design, the system jointly addresses **global** irregularities (alignment and inclination at the cable level) and **local** defects (point-wise deviations, directional changes between segments, and abnormal space between points), obtaining a

detailed analysis without sacrificing scene-level consistency. The evaluation of the model is also validated quantitatively and qualitative, using visualizations to improve the understanding of the whole scene and making it easier for non-specialized technicians to inspect each scene.

From an operational standpoint, the approach scales favorably to larger LiDAR datasets[35]. First, the anomaly detectors are based on linear-algebraic functions like centering, PCA, or projections. Second, the segmentation PointNet model is also designed to scale easily to larger databases and files with more points per scene.

For future investigation and further improvement of this model, two directions are particularly promising: · Enhanced point representations. Upgrading the segmentation from PointNet to PointNet++[57] would add hierarchical local geometry into feature learning and improve the segmentation of real-world LiDAR data. · Flexible classifiers. While gradient-boosted trees (XGBoost and LightGBM) provided an effective and interpretable classifier, modern GBDT frameworks such as CatBoost[54] could be added to see it the classifiers improve, offering efficiency gains and stronger handling of categorical/statistical drift in certain regimes.

Impact on the railway sector

Beyond the methodological contributions, the potential impact of this project on the railway sector is very big. By enabling automatic detection of anomalies in overhead cables from LiDAR data, the system directly addresses one of the key challenges of modern railway management: the continuous monitoring and maintenance of large-scale infrastructure networks.

First, the models support the transition from reactive to **predictive maintenance**[63, 9]. By identifying defects at the global and local scale, railway operators can plan interventions more efficiently, reducing unplanned downtime, preventing service disruptions and severe accidents.

Second, the approach contributes to a considerable **reduction of costs**, as it minimizes manual inspection campaigns, traditionally time-consuming and labor-intensive. Automatically identifying anomalies allows the team to use the resources available at the most critical areas, optimizing time and money.

Third, the adoption of this improves **safety**[1], not only for railway staff but also for passengers and cargo. The early detection of anomalies such as cable

fractures, excessive inclination, or abrupt directional changes prevents accidents.

Finally, the proposed method aligns with the broader agenda of **digitalization in railway infrastructure**[18]. The capacity to automatically process large LiDAR datasets and turn them into actionable insights places this work within the framework of "smart railways", where data-driven decision-making is important to improve operational efficiency, sustainability, and long-term duration of transport systems.

Applications of the project in other sectors

Although this project was designed for railway infrastructures, the proposed method can be also extended to other sectors where structural integrity and anomaly detection are critical. The combination of semantic segmentation, classification, and anomaly analysis from LiDAR or 3D point cloud data is not limited to cables and poles but can be adapted to other types of large-scale infrastructure.

A first area of application is the **energy sector**, where there are similar challenges in monitoring high-voltage power lines and transmission networks. Recent studies[32, 73] have shown the effectiveness of LiDAR-based analysis for detecting vegetation encroachment, cable sagging, or structural defects in power distribution systems.

Secondly, in **civil engineering**[79, 45], anomaly detection models can be applied to the structural health monitoring of bridges, tunnels, and buildings, where deviations or cracks in 3D geometry must be identified early to prevent catastrophic failures.

Another promising application is in the **telecommunications sector**[71], where overhead cables and antenna towers require continuous monitoring to ensure service continuity and resilience against environmental changes.

Finally, the methodology could be extended to the **aerospace and aeronautical industries**[47, 53], where LiDAR and 3D scanning technologies are increasingly used for quality control and damage detection on aircraft fuselages or spacecraft components.

Appendix A

Alignment of the project with the United Nations Sustainable Development Goals (SDGs)

In the current world, it has become more important for research and innovation to align with international objectives and sustainability. The United Nations Sustainable Development Goals (SDGs) provide a guide for governments, institutions and communities that want to create a fairer, environmentally responsible future, and is based on social, economical and environmental challenges all around the world.

The purpose of this section is to explain how this project aligns with the SDGs, showing the ways in which it makes it easier to commit to these goals. By thinking about the principles of the SDGs, applying them into its design and objectives, this model contributes to technological progress that improves the quality of life of everyone.

Specifically, this work aligns with at least five of the SDGs established by the United Nations in 2015 as objectives for the 2030 Agenda:

- **Industry, Innovation and Infrastructure (9).** Addressed in this project by promoting the use of artificial intelligence to optimize inspection, maintenance, and repairing tasks in critical infrastructure. The main goal is to reduce reliance on manual interventions while advancing the digitalization of this industry. This vision is in line with initiatives led by the European

Union Agency for Railways (ERA) and the Europe's Rail Joint Undertaking [17], which encourage the development of more resilient, intelligent, and sustainable infrastructures, as outlined in the EU's common transport policy.

- **Reduced inequalities (10).** This project develops tools that can be adopted by countries regardless of their level of technological resources. Because the methodology is based on reproducible models and open-access datasets, it can be adapted to different national contexts, from highly industrialized railway networks to less developed regional systems. This promotes equal access to advanced digital solutions, reducing dependence on costly technologies and enabling wider adoption, contributing to narrow the technological gap within and among countries, while fostering more inclusive and sustainable mobility on a global scale.

- **Sustainable cities and communities (11).** Promoting rail transport as an efficient, safe, and low-emission type of mobility directly supports urban sustainability. Strengthening the role of railways helps reduce dependence on private cars, alleviate congestion in cities, and lowers both noise and air pollution levels. Beyond these immediate benefits, it also contributes to encouraging more integrated and sustainable urban models.

- **Climate Action (13).** Sustainability is closely connected to the railway sector, as trains are one of the most sustainable modes of transport available today. Compared to road and air travel, the train has significantly lower greenhouse gas emissions, consumes less energy per passenger or ton of goods transported. This project reinforces the long-term sustainability of rail transport, helping it to remain a safe, efficient, and environmentally friendly alternative.

- **Partnerships for the goals (17).** This project is framed within the context of international cooperation to build a common, interoperable, and digital railway network. This goal is to create the Trans-European Transport Network (TEN-T)[21], which aims to connect rail systems across European Union's member countries and to ensure seamless cross-border mobility. European transport policy [17] emphasizes the importance of interoperability, standardization, and shared digital platforms so that national networks can operate as part of a single system. In doing so, it reinforces the idea that the modernization of the railway sector is a collective effort, where shared knowledge, common standards, and joint strategies are essential to achieving sustainable, efficient, and resilient transport across Europe.

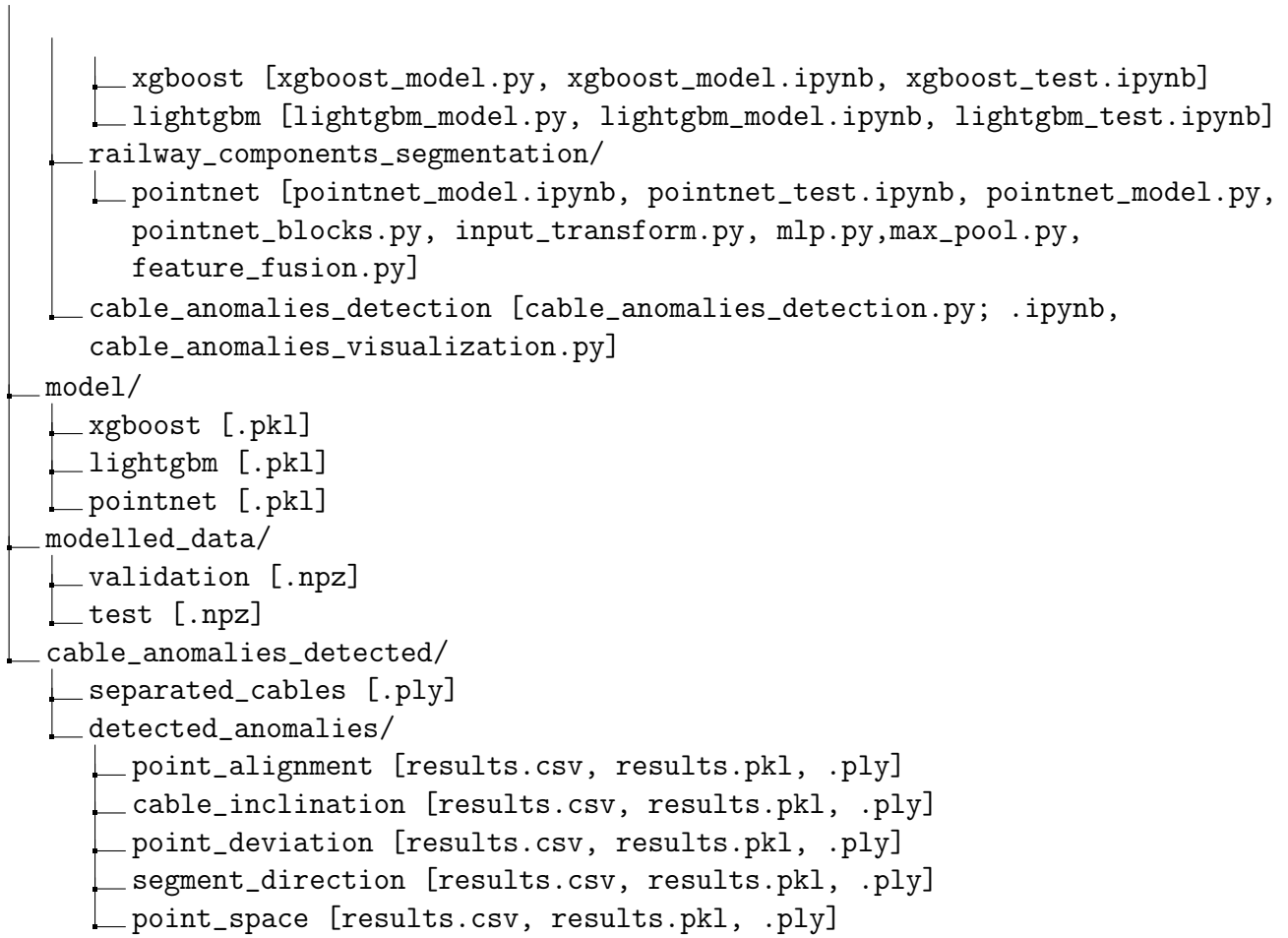
Appendix B

Project directory structure and source code repository

The complete source code of the project, together with all the processed data, experimental results, and supplementary documentation, is available at the following repository: ***Project Repository Link***

This repository provides full access to the implementation details and outcomes, ensuring transparency, reproducibility, and further analysis.

```
project/
├── raw_data [.ply]
├── processed_data/
│   ├── normalized_data [.npz]
│   ├── splitted_data [.npz]
│   ├── resampled_data [.npz]
│   └── filtered_data [.npz]
├── data_exploration [data_load.py, raw_data_analysis.ipynb]
├── data_processing/
│   ├── data_normalization [data_normalization.ipynb]
│   ├── data_split [data_split.py, data_split.ipynb]
│   ├── data_resample [data_resample.py, data_resample.ipynb]
│   └── data_filter [data_filter.py, data_filter.ipynb]
└── modelling/
    └── railway_presence_classification/
```


xgboost [xgboost_model.py, xgboost_model.ipynb, xgboost_test.ipynb]
lightgbm [lightgbm_model.py, lightgbm_model.ipynb, lightgbm_test.ipynb]
railway_components_segmentation/
pointnet [pointnet_model.ipynb, pointnet_test.ipynb, pointnet_model.py,
pointnet_blocks.py, input_transform.py, mlp.py,max_pool.py,
feature_fusion.py]
cable_anomalies_detection [cable_anomalies_detection.py; .ipynb,
cable_anomalies_visualization.py]
model/
xgboost [.pkl]
lightgbm [.pkl]
pointnet [.pkl]
modelled_data/
validation [.npz]
test [.npz]
cable_anomalies_detected/
separated_cables [.ply]
detected_anomalies/
point_alignment [results.csv, results.pkl, .ply]
cable_inclination [results.csv, results.pkl, .ply]
point_deviation [results.csv, results.pkl, .ply]
segment_direction [results.csv, results.pkl, .ply]
point_space [results.csv, results.pkl, .ply]

Appendix C

Detailed explanation of T-Net and MLP

T-NET

The T-Net is a component of the PointNet architecture, introduced to create invariance of the model to geometric transformations of the input point cloud. Its main goal is to learn an affine transformation matrix that aligns the input data into a canonical space before feature extraction, reducing the sensitivity of the network to rotations and translations.

Formally, given an input point cloud

$$X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}, \quad \mathbf{x}_i \in \mathbb{R}^d,$$

where n is the number of points and d the dimensionality of each point, the T-Net learns a transformation matrix

$$\mathbf{T} \in \mathbb{R}^{d \times d}.$$

The transformed points are then obtained as

$$\mathbf{x}'_i = \mathbf{T}\mathbf{x}_i, \quad \forall i \in \{1, \ldots, n\}.$$

The transformation matrix $\mathbf{T}$ is estimated by a small network with a similar structure to PointNet itself (see Figure C.1). This network applies a series of shared multilayer perceptron (MLP) layers to each point, followed by a max-pooling operation to aggregate a global feature. Finally, a set of fully connected layers regresses the entries of $\mathbf{T}$.

Since $\mathbf{T}$ is required to be close to an orthogonal matrix to avoid degenerate transformations, a regularization term is added to the loss function:

$$\mathcal{L}_{\text{reg}} = \left\| \mathbf{I} - \mathbf{T}\mathbf{T}^{\top} \right\|_F^2 ,$$

where $\| \cdot \|_F$ denotes the Frobenius norm and $\mathbf{I}$ is the identity matrix.

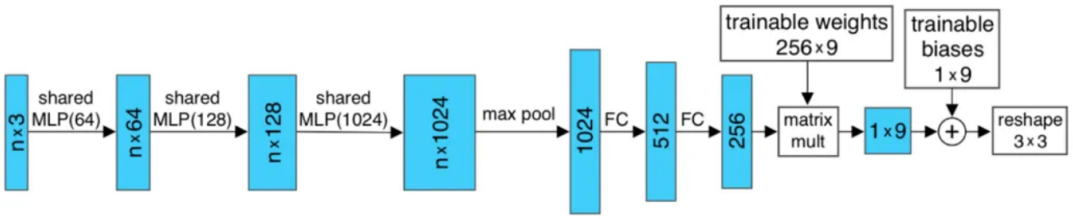


Figure C.1: T-Net network arquitecture

Multilayer Perceptron (MLP)

Another fundamental component of PointNet is the Multilayer Perceptron (MLP), which is used extensively throughout the architecture for feature extraction. The MLP operates on each point individually, applying a sequence of affine transformations (see Figure C.2) followed by nonlinear activations in order to map the input space into progressively higher-dimensional feature spaces.

Formally, given an input point

$$\mathbf{x} \in \mathbb{R}^d,$$

an MLP with L layers computes a transformation

$$f(\mathbf{x}) = f^{(L)}\big(f^{(L-1)}(\dots f^{(1)}(\mathbf{x}))\big),$$

where each layer is defined as

$$f^{(l)}(\mathbf{h}) = \sigma\left(\mathbf{W}^{(l)}\mathbf{h} + \mathbf{b}^{(l)}\right),$$

with $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ denoting the weight matrix and bias vector of layer l, and $\sigma(\cdot)$ representing a nonlinear activation function such as ReLU.

In PointNet, the MLP is shared across all input points. That is, the same weights are applied independently to each point $\mathbf{x}_i$, ensuring permutation invariance of the point set:

$$\mathbf{h}_i = f(\mathbf{x}_i), \quad \forall i \in \{1, \ldots, n\}.$$

By stacking multiple MLP layers, PointNet is able to progressively capture increasingly complex geometric features of individual points. When combined with a symmetric aggregation function (such as max pooling), these per-point features are integrated into a global feature vector that encodes the overall structure of the point cloud.

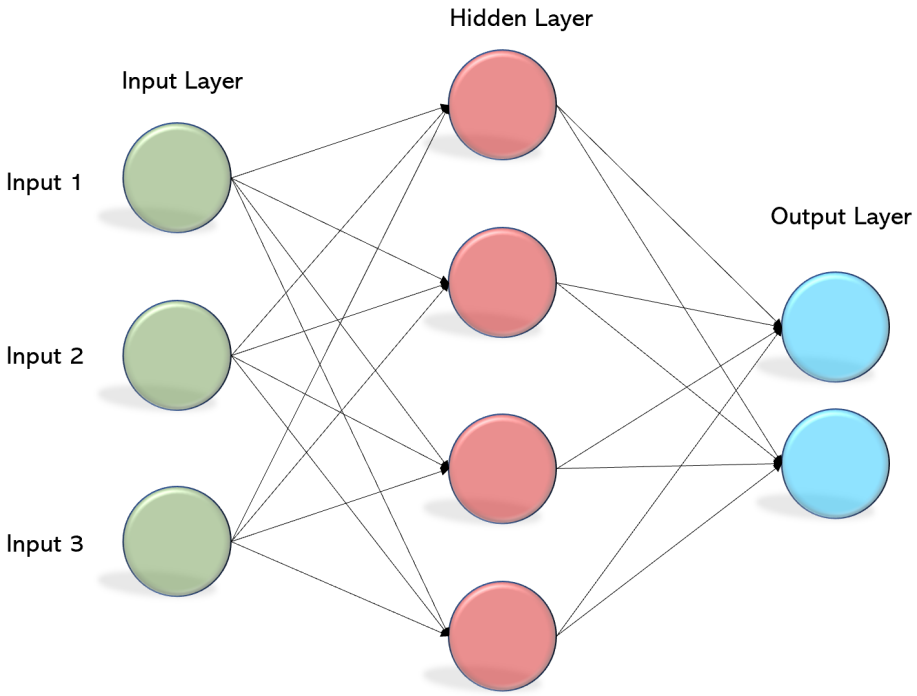


Figure C.2: MLP arquitecture

Bibliography

[1] M. O. Adeagbo. Revamping structural health monitoring of rail infrastructure using digital twins and digital shadows. *Advanced Engineering Informatics*, 2024. Explores DT applications for structural health monitoring in rail systems.

[2] AdorTech. *Non-destructive testing*, 2023.

[3] Anonymous. *Active learning for railway infrastructure point cloud semantic segmentation. arXiv preprint arXiv:2410.13383*, 2024.

[4] Anonymous. *Railway Infrastructure Point Cloud Semantic Segmentation Using Active Learning. arXiv preprint arXiv:2410.10832*, 2024.

[5] Mostafa Arastounia. *Automated recognition of railroad infrastructure in rural areas from LiDAR data. Remote Sensing*, 7(11):14916–14938, 2015.

[6] Mostafa Arastounia. *Automated recognition of railroad infrastructure from LiDAR data based on image processing techniques. Sensors*, 17(9):2054, 2017.

[7] Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, 6(1):20–29, 2004.

[8] G. Bianchi, C. Fanelli, F. Freddi, F. Giuliani, and A. La Placa. *Systematic review railway infrastructure monitoring: From classic techniques to predictive maintenance. Advances in Mechanical Engineering*, 17(1), 2025.

[9] Giovanni Bianchi, Chiara Fanelli, Francesco Freddi, Felice Giuliani, and Aldo La Placa. Systematic review of railway infrastructure monitoring: From classical techniques to predictive maintenance. *Advances in Mechanical Engineering*, 17(1):1–26, 2025.

[10] Y. Cao et al. *Railway catenary condition monitoring: A systematic mapping of recent research. Sensors*, 24(2):1034, 2024.

[11] Y. Cao, T. Zhang, X. Li, et al. *Meta-learning with GANs for anomaly detection in high-speed railway inspections. Expert Systems with Applications*, 200:116921, 2022.

[12] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[13] J. Chen, Z. Liu, H. Wang, A. Nunez, and Z. Han. *Automatic defect detection of fasteners on the catenary support device using deep convolutional neural network. IEEE Transactions on Instrumentation and Measurement*, 67(2):257–269, 2018.

[14] L. Chen, J. Jung, and G. Sohn. *Multi-scale hierarchical CRF for railway electrification asset classification from mobile laser scanning data. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(8):3131–3148, 2019.

[15] Tianqi Chen and Carlos Guestrin. *XGBoost: A Scalable Tree Boosting System.* In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, pages 785–794. ACM, 2016.

[16] M. Corongiu, A. Masiero, and G. Tucci. *Classification of railway assets in mobile mapping point clouds.* In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume XLIII-B1-2020, pages 219–225, 2020.

[17] Council of the European Union. *Rail transport policy*. `https://www.consilium.europa.eu/es/policies/rail-transport-policy/`, 2024.

[18] Lorenzo De Donato, Ruth Dirnfeld, Alessandra Somma, Alessandra De Benedictis, Francesco Flammini, Stefano Marrone, Mehdi Saman Azari, and Valeria Vittorini. Towards ai-assisted digital twins for smart railways: Preliminary guideline and reference architecture. *Springer Journal of Computing*, 2023. In: Lecture Notes in Networks and Systems.

[19] L. Dekker, R. van der Meij, D. Louter, et al. *Digital twins in railway infrastructure through LiDAR and MLS technologies. Automation in Construction*, 147:104695, 2023.

[20] European Environment Agency (EEA). *Rail and waterborne transport*. https://www.eea.europa.eu/publications/rail-and-waterborne-transport, 2021.

[21] European Parliament. *El transporte ferroviario*. https://www.europarl.europa.eu/factsheets/es/sheet/130/el-transporte-ferroviario, 2024.

[22] Federal Railroad Administration. *Track Inspection Time Study*. Technical report, U.S. Department of Transportation, Federal Railroad Administration, 2009.

[23] Yoav Freund and Robert E. Schapire. *A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[24] Jerome H. Friedman. *Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics*, 29(5):1189–1232, 2001.

[25] Vito Gagliardi, Giuseppe Galasso, and Antonio Benedetto. Railway infrastructure monitoring using mobile mapping point clouds. *Infrastructures*, 9(4):71, 2024.

[26] M. García and J. López. *Integration of LiDAR and computer vision for railway asset management. Journal of Rail Transport Planning & Management*, 11:100–112, 2019.

[27] I. Ghazouani et al. *A systematic literature review of defect detection in railways using machine vision-based inspection methods. Case Studies in Nondestructive Testing and Evaluation*, 15:100204, 2024.

[28] M. Ghiasi et al. *Unsupervised anomaly detection in railway track geometry using One-Class SVMs. Transportation Engineering*, 2024.

[29] L. González-deSantos, C. Ordóñez, J. Balado, et al. *Digital Twinning of Railway Overhead Line Equipment from Airborne LiDAR Data. Automation in Construction*, 118:103281, 2020.

[30] J. Grandío, B. Riveiro, M. Soilán, and P. Arias. *Point cloud semantic segmentation of complex railway environments using deep learning. Automation in Construction*, 141:104425, 2022.

[31] Groupe SNCF. *Catenary incident*. Technical report, SNCF, 2023.

[32] Yulan Guo et al. Lidar-based inspection of power transmission lines: A review of current status and future trends. *IEEE Transactions on Power Delivery*, 2022.

[33] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. *Deep Learning for 3D Point Clouds: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(12):4338–4364, 2020.

[34] J. Harb, N. Rébéna, R. Chosidow, G. Roblin, R. Potarusov, and H. Hajri. *FRSign: A large-scale traffic light dataset for autonomous trains. arXiv preprint arXiv:2002.05665*, 2020.

[35] Qiangui Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11108–11117, 2020.

[36] International Railway Journal. *Dutch freight train fire caused by temporary catenary fix*, 2023.

[37] International Union of Railways (UIC). *The modal share of rail in inland transport and infrastructure investment*. `https://uic.org/com/IMG/pdf/the_modal_share_of_rail_in_inland_transport_and_infrastructure_investment.pdf`, 2023.

[38] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. *LightGBM: A Highly Efficient Gradient Boosting Decision Tree. arXiv preprint arXiv:1712.01042*, 2017.

[39] A. Kharroubi, Z. Ballouch, R. Hajji, A. Yarroudh, and R. Billen. *Multi-context point cloud dataset and machine learning for railway semantic segmentation (Rail3D). Infrastructures*, 9(4):71, 2024.

[40] A. Kharroubi, Z. Ballouch, R. Hajji, A. Yarroudh, and R. Billen. *Multi-context point cloud dataset and machine learning for railway semantic segmentation (Rail3D). Infrastructures*, 9(4):71, 2024.

[41] D. Lamas, B. Riveiro, P. Arias, et al. *Railway infrastructure 3D point cloud classification using heuristic workflows. Remote Sensing*, 13(12):2332, 2021.

[42] P. Leibner, F. Hampel, and C. Schindler. *GERALD: A novel dataset for the detection of German mainline railway signals. Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, 237(10):1332–1342, 2023.

[43] H. Li et al. *Anomaly-ShapeNet: A synthetic benchmark and IMRNet for 3D anomaly detection. arXiv preprint arXiv:2403.12345*, 2024.

[44] H. Li et al. *MAESTRO: 3D anomaly detection via reconstruction-only.* In *Proceedings of VISAPP 2025*, 2025.

[45] H. Li and J. Zhou. Applications of 3d point cloud analysis in structural health monitoring: A review. *Engineering Structures*, 212:110522, 2020.

[46] Y. Liu, H. Chen, X. Li, et al. *A Railway LiDAR Point Cloud Reconstruction Based on Target Detection and Trajectory Filtering. ISPRS Journal of Photogrammetry and Remote Sensing*, 2022.

[47] Y. Liu et al. 3d point cloud processing for aerospace structural integrity: Challenges and opportunities. *Aerospace Science and Technology*, 138:107430, 2023.

[48] Z. Liu, L. Chen, H. Wang, et al. *Accurate track geometry inspection using MLS-based LiDAR point clouds. IEEE Transactions on Intelligent Transportation Systems*, 22(3):1231–1242, 2020.

[49] Llew Mason, Jonathan Baxter, Peter Bartlett, and Marcus Frean. *Functional Gradient Techniques for Combining Hypotheses. Journal of Artificial Intelligence Research*, 10:163–190, 1999.

[50] McKinsey & Company. *Safe, smart, and green: Boosting European passenger rail's modal share*. `https://www.mckinsey.com/industries/infrastructure/our-insights/safe-smart-and-green-boosting-european-passenger-rails-modal-share`, 2022.

[51] Muchahistoria. *Ferrocarril*, 2023.

[52] Office of Rail Regulation. *Periodic Review 2008: Consultation on Network Rail's Strategic Business Plan.* Technical report, UK Office of Rail Regulation, 2008.

[53] S. Park and J. Kim. Damage detection in aircraft structures using lidar and deep learning. *Journal of Aerospace Engineering*, 34(5):04021055, 2021.

[54] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pages 6639–6649, 2018.

[55] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017.

[56] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation.* In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017.

[57] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[58] B. Qiu, Y. Zhou, L. Dai, B. Wang, J. Li, Z. Dong, C. Wen, Z. Ma, and B. Yang. *WHU-Railway3D: A diverse dataset and benchmark for railway point cloud semantic segmentation. IEEE Transactions on Intelligent Transportation Systems*, 2024.

[59] R. Qiu et al. *Multi-rotor UAV-based track geometry measurement system for railway inspection.* In *University of Nevada, Las Vegas Technical Reports*, 2022.

[60] L. Ramos, A. de la Fuente, and P. Gómez. *Predictive maintenance of railway infrastructure using IoT and machine learning: A case study*. *Transportation Research Part C: Emerging Technologies*, 120:102777, 2020.

[61] ResearchGate. *High-speed train and its running environment.* Image, 2018.

[62] Soloagentes. *Historia del ferrocarril: Un viaje por el pasado ferroviario*, 2023.

[63] R. Tang et al. A survey on artificial intelligence applications in railway transport. *Transportation Research Part C: Emerging Technologies*, 2022. Systematic literature review of AI in railway transport, highlighting predictive maintenance.

[64] The Local France. *Eurostar severely disrupted after fatal accidents*, 2025.

[65] R. Tilly, P. Neumaier, K. Schwalbe, P. Klasek, R. Tagiew, P. Denzler, T. Klockau, M. Boekhoff, and M. Köppel. *Open sensor data for rail 2023 (OSDaR23)*, 2023.

[66] B. Ton. *Labelled high resolution point cloud dataset of 15 catenary arches in the Netherlands*, 2022.

[67] B. Ton, F. Ahmed, and J. Linssen. *Semantic segmentation of terrestrial laser scans of railway catenary arches: A use case perspective. Sensors*, 23(1):222, 2023.

[68] Trenvista. *Fallo en la catenaria interrumpe la LAV Madrid–Andalucía durante 15 horas*, 2023.

[69] Mikaela A. Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1588–1597, 2019.

[70] B. van der Werf. *Learning behaviour of SPVConv-based models for railway infrastructure point cloud classification.* PhD thesis, University of Twente, 2023.

[71] K. Vasdev. Drones technology in gis for telecom cell tower inspection. *International Journal on Science and Technology (IJSAT)*, 12(3):1–5, 2021. Demonstrates the use of LiDAR-equipped drones and GIS for structural monitoring of telecom towers.

[72] L. Wang, M. Zhao, et al. *Measurement-based condition monitoring of railway signaling cables. arXiv preprint arXiv:2109.01781*, 2021.

[73] X. Wang et al. Automatic detection of power line defects from large-scale lidar data using deep learning. *Remote Sensing*, 15(3):543–558, 2023.

[74] Y. Wang. *Railway SLAM Dataset*, 2022.

[75] J. Wen, X. Liu, et al. *DHT-CL: Multi-modal contrastive learning for obstacle detection in railways under adverse weather. Electronics*, 13(1):220, 2024.

[76] M. Werder, F. van der Meer, and G. Vosselman. *Point cloud analysis of railway infrastructure: a systematic literature review. Journal of Rail Transport Planning & Management*, 22:100309, 2022.

[77] Q. Wu, T. Zhang, et al. *Unsupervised anomaly detection for split pin defects in railway catenary systems. Automation in Construction*, 160:105325, 2024.

[78] Q. Wu, T. Zhang, M. Li, et al. *Anomaly Detection of Railway Infrastructure Using Deep Learning and Point Cloud Data. Automation in Construction*, 158:105081, 2024.

[79] L. Xu and J. Zhang. Bridge structural health monitoring with lidar and ai-based anomaly detection. *Structural Control and Health Monitoring*, 2021.

[80] A. Yarroudh, R. Hajji, A. Kharroubi, et al. *UAV-based LiDAR for monitoring catenary systems in railway networks. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-4/W2-2024:233–240, 2024.

[81] A. Yarroudh, A. Kharroubi, R. Hajji, et al. *Railway point cloud semantic segmentation benchmark: new ISPRS archives contribution.* In *ISPRS Archives*, volume XLVIII-2-W8, pages 477–484, 2024.

[82] Y. Yu, J. Li, W. Zhao, et al. *Voxel-based multi-scale neural network for real-time rail recognition. arXiv preprint arXiv:2201.02726*, 2022.

[83] O. Zendel, M. Murschitz, M. Zeilinger, D. Steininger, S. Abbasi, and C. Beleznai. *RailSem19: A dataset for semantic rail scene understanding*, 2019. Dataset.

[84] T. Zhang, Y. Li, et al. *Inspection of railway catenary systems using machine learning with domain knowledge integration. Scientific Reports*, 15:1289, 2025.

[85] Yiming Zhang and Michael Rabbat. A deep learning framework for point clouds: Pointnet and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[86] Y. Zheng, Y. Wang, and Z. He. *Railway track condition monitoring using 3D point cloud data and deep learning. Automation in Construction*, 124:103579, 2021.

[87] X. Zhou, H. Liu, et al. *Defect diagnosis of rigid catenary system based on pantograph vibration performance. Actuators*, 13(5):162, 2023.