Interprétation automatique de données géophysiques par techniques d'apprentissage

Douba Jafuno

UNIVERSITÉ SAVOIE MONT BLANC

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ SAVOIE MONT BLANC

Spécialité : **Sciences et techniques de l'information et de la communication, Traitement de l'information**

Arrêté ministériel : 25 mai 2016

Présentée par

Douba JAFUNO

Thèse dirigée par **Guillaume GINOLHAC** et
codirigée par **Ammar MIAN** et **Nickolas STELZENMULLER**

préparée au sein du **Laboratoire LISTIC**
et de **l'École Doctorale SIE**

Interprétation automatique de données géophysiques par techniques d'apprentissage

Thèse soutenue publiquement le **4 Novembre 2025**
devant le jury composé de :

Mme Audrey GIREMUS
Professeure, Université de Bordeaux, (Rapportrice)
M. Sylvain CHEVALLIER
Professeur, Université de Paris-Saclay, (Rapporteur)
M. Thomas OBERLIN
Professeur, ISAE-SupAero Toulouse, (Examinateur)
Mme Sophie GIFFARD-ROISIN
Chargé de Recherche, IRD, Université Grenoble Alpes, (Examinatrice)
M. Guillaume GINOLHAC
Professeur, Université Savoie Mont Blanc, (Directeur de thèse)
M. Ammar MIAN
Maitre de conférence, Université Savoie Mont Blanc, (Co-directeur de thèse)
M. Nickolas STELZENMULLER
Ingénieur de recherche RDI, Géolithe, Crolles, (Co-directeur de thèse)

Remerciements

Cette thèse a été réalisée au sein du Laboratoire d'Informatique, Systèmes, Traitement de l'Information et de la Connaissance (LISTIC), à Annecy.

Je tiens tout d'abord à exprimer ma profonde reconnaissance à Guillaume Ginolhac, directeur de cette thèse. Merci pour ta confiance, ton écoute, ta rigueur scientifique et pour ton accompagnement constant tout au long de ces années. Je remercie également Ammar pour son encadrement de proximité, la qualité de nos échanges, sa bienveillance et ses conseils, toujours justes et constructifs. Votre complémentarité, votre implication et votre disponibilité ont été déterminantes dans l'avancée de ce travail, et je vous en suis profondément reconnaissant.

Ce travail s'inscrit dans le cadre d'une collaboration avec l'entreprise Géolithe. Je remercie sincèrement Nickolas Stelzenmuller pour son implication, ses retours pertinents et sa disponibilité à chaque étape du projet. J'adresse également mes remerciements à Alexandra Royer, Aline Esparel, Fabrice Guyoton et Jean-Marc Verdier pour leur soutien, leur professionnalisme et leur participation active à cette recherche.

Je remercie chaleureusement Florent Bouchard et Matthieu Gallet pour leur collaboration enthousiaste et pour les échanges scientifiques que nous avons pu avoir. Je n'oublie pas non plus Arnaud Breloy, à qui je dois beaucoup, puisqu'il m'a parlé de cette opportunité de thèse. Merci pour ta confiance et ton soutien dès le départ.

Je remercie tous les doctorants du LISTIC pour leur convivialité, leur entraide et leur bonne humeur, qui ont largement contribué à rendre ce parcours agréable. Merci également à Emmanuel, directeur du laboratoire, pour son accueil et pour la qualité du cadre de travail mis en place au LISTIC.

Madame Audrey Giremus, Monsieur Sylvain Chevallier, Monsieur Thomas Oberlin et Madame Sophie Giffard-Roisin m'ont fait l'honneur de juger cette thèse. Je leur adresse mes sincères remerciements pour le temps qu'ils ont consacré à l'évaluation de ce travail et pour la richesse de leurs remarques.

Enfin, je souhaite exprimer toute ma gratitude à ma famille. À ma mère, pour son amour, son courage et son soutien indéfectible. À mon père, aujourd'hui disparu, qui m'a toujours encouragé dans mes études ; cette thèse lui est dédiée. À mes frères et à mes sœurs, pour leur affection, leur présence et leurs encouragements constants. Merci du fond du cœur.

Table des matières

Table des figures

Introduction

1) Le radar à pénétration de sol

Le radar à pénétration de sol (GPR, pour *Ground Penetrating Radar*) est une technologie d'imagerie non destructive qui permet d'explorer les structures du sous-sol en analysant les échos réfléchis des ondes électromagnétiques émises dans le sol. À l'aide d'une émission d'une onde appelée Ricker et du déplacement du système, le GPR fournit une imagerie appelée B-scan, représentant une coupe longitudinale de la surface sondée [2]. Les objets enfouis, tels que des câbles ou des cavités, apparaissent typiquement sous la forme d'hyperboles.

Ces images sont cependant fortement bruitées : le rapport signal sur bruit (RSB) est généralement faible en raison des hétérogénéités fines du sol. Comme la vitesse de l'onde est inconnue et varie selon le milieu observé (à travers la permittivité diélectrique), les méthodes classiques d'imagerie radar, telles que la synthèse d'ouverture (Synthetic Aperture Radar, SAR), ne sont pas applicables.

Un nombre important de travaux consiste à améliorer l'image, en particulier le RSB. La plupart sont basés sur des méthodes d'inversion [3–5].

Dans cette thèse, nous souhaitons aller plus loin et proposons de construire des modèles permettant de détecter, de localiser et de classer les objets enfouis. On se concentrera au début sur l'étape de classification, en utilisant l'imagette de l'hyperbole après un processus de pré-traitement, pour évaluer si ces données sont suffisantes pour atteindre de bonnes performances. Dans ce travail, nous nous plaçons dans un cadre supervisé, car nous disposons d'un jeu de données annoté fourni par la société **Geolithe**.

Néanmoins, ce jeu de données contient un nombre limité d'exemples et présente une grande variabilité en termes de conditions expérimentales (altitude, humidité, nature du sol, etc.). Le but de la thèse est donc de construire des modèles robustes face à ces conditions.

2) Une approche hybride entre apprentissage profond et covariance

Plusieurs approches ont été proposées pour classer les objets dans les images GPR. Certaines s'appuient uniquement sur la forme géométrique des hyperboles et utilisent des classifieurs simples comme les SVM (*Support Vector Machine*) ou les forêts aléatoires (*Random Forest*) [6, 7]. Ces méthodes restent limitées dans des contextes bruités ou lorsque les objets présentent une diversité de signatures.

D'autres travaux ont utilisé des réseaux de neurones convolutionnels (Convolutional Neural Network CNN) , mais avec des architectures peu profondes [8–10], limitant ainsi leur capacité à généraliser. À l'inverse, les modèles profonds comme ResNet, AlexNet ou MobileNetV2 sont puissants, mais nécessitent de grands volumes de données annotées, difficilement accessibles dans le contexte du GPR, en particulier dans cette thèse.

Une alternative consiste à combiner les CNN avec des représentations statistiques plus robustes. Dans [11], il a été montré qu'il est possible d'extraire une matrice de covariance à partir des activations intermédiaires d'un CNN. Cette matrice encode les corrélations entre filtres, offrant une représentation

plus stable que l'image brute. Les expériences menées utilisaient alors uniquement des classifieurs comme SVM ou Random Forest en aval, sans entraînement de bout en bout.

Dans cette thèse, nous proposons d'aller plus loin en intégrant ces descripteurs de covariance dans une architecture entraînable de bout en bout, en utilisant des couches différentiables adaptées aux matrices SPD (*Symmetric Positive Definite*) [12–15] . Nous appliquerons cette démarche à la fois à la classification, à la détection et à la localisation.

3) Problématique scientifique et objectifs

Cette thèse s'inscrit dans le cadre de la **détection** et de la **classification d'objets enfouis** à partir d'images de radar à pénétration de sol (GPR) acquises dans le cadre du projet industriel Géolithe ; l'enjeu principal a donc été de concevoir des modèles d'apprentissage capables d'exploiter efficacement la structure spatiale et statistique des radargrammes, tout en demeurant **robustes aux variations expérimentales**.

Les **principaux défis scientifiques** identifiés sont les suivants :

- **Faible volume de données** : le nombre d'échantillons annotés est limité, et la labellisation manuelle des hyperboles reste une tâche longue et subjective ;
- **Faible rapport signal/bruit** : les données GPR sont souvent fortement bruitées en raison des réflexions parasites, du couplage antenne-sol ou des hétérogénéités du milieu, ce qui complique l'extraction de caractéristiques discriminantes et la détection fiable des objets enfouis ;
- **Variabilité expérimentale et complexité d'interprétation** : les signaux GPR dépendent fortement des paramètres physiques du système (type de sol, fréquence, élévation, profondeur des objets) et la forme des hyperboles observées résulte à la fois des propriétés du milieu et de celles de l'objet, rendant difficile la distinction entre **causes physiques** et **effets géométriques**.

Ces difficultés rapprochent le GPR d'autres modalités, comme l'échographie médicale ou l'imagerie EEG, où les signaux sont également bruités et dépendent de conditions expérimentales variables. Ainsi, les défis rencontrés avec le GPR sont représentatifs de problématiques plus générales en apprentissage automatique : robustesse, invariance et généralisation à partir de jeux de données limités.
L'objectif général de cette thèse est donc de **développer un modèle robuste et invariant**, capable de détecter et de classer automatiquement les objets enfouis dans les radargrammes GPR, malgré un faible volume de données annotées et de **fortes variations expérimentales**.

Ce travail s'articule autour de **quatre problématiques principales**, qui structurent l'ensemble du manuscrit :

- **P1 — Structuration et préparation des données GPR :** Comment structurer et prétraiter les données GPR aéroportées issues du projet Géolithe afin de constituer un ensemble exploitable d'imagettes d'hyperboles pour l'apprentissage automatique ?
- **P2 — Introduction des méthodes d'apprentissage supervisé :** Comment introduire et adapter des méthodes d'apprentissage automatique classiques (ACP, SVM et RF) à la classification des données GPR ?
- **P3 — Robustesse des modèles face aux variations expérimentales :** De quelle façon améliorer la robustesse des modèles de classification des hyperboles radar, notamment face aux décalages entre les données d'entraînement et de test ?
- **P4 — Extension à la détection d'objets enfouis :** Comment étendre cette approche à la détection d'objets en intégrant SPDNet dans une architecture de type Faster R-CNN ?

Enfin, **chaque chapitre** du manuscrit apportera une réponse progressive à ces problématiques : le **chapitre 1** répondra à **P1**, le **chapitre 2** à **P2**, le **chapitre 3** à **P3**, et le **chapitre 4** à **P4**.

4) Structure de la thèse et contributions

Pour répondre à ces problématiques, ce manuscrit de thèse est structuré en quatre chapitres principaux :

- Le **chapitre 1** présente les spécificités du GPR et les principes physiques liés à la propagation électromagnétique. Nous explorons ensuite des éléments de la campagne expérimentale de Géolithe ainsi que les étapes de construction du dataset.
- Le **chapitre 2** compare plusieurs stratégies classiques de classification supervisée sur des données GPR fournies par la société Géolithe. Il débute par l'étude de méthodes classiques combinant une réduction de dimension par Analyse en Composantes Principales (ACP) avec une classification par machines à vecteurs de support (SVM) ou forêts aléatoires (Random Forest). Le chapitre explore ensuite une approche reposant sur l'extraction de caractéristiques par réseaux de neurones convolutionnels, suivie d'un covariance pooling et d'une classification via SVM ou Random Forest. Une architecture convolutive peu profonde (Shallow Convolutional Neural Network, S-CNN) est également évaluée avant d'envisager des modèles convolutifs profonds (AlexNet, ResNet34, MobileNetV2) entraînés soit à partir de zéro, soit par transfert d'apprentissage. Enfin, le chapitre s'intéresse aux enjeux de robustesse des modèles face au faible volume de données, à la présence de labels bruités et aux variations expérimentales.
- Le **chapitre 3** introduit donc des modèles conçus pour améliorer la robustesse de la classification d'images GPR face à un faible volume de données et à de fortes variations expérimentales. L'approche repose sur l'extraction de caractéristiques par un réseau ResNet-34, à partir duquel deux variantes sont explorées : la première, notée SRCNet, consiste à concaténer les cartes de caractéristiques issues de plusieurs couches intermédiaires du réseau ; la seconde, RCNet, se limite à la dernière couche. Ces sorties sont ensuite utilisées pour construire un tenseur de caractéristiques à trois dimensions, à partir duquel une matrice de covariance normalisée (basée sur la Sample Covariance Matrix, SCM) est calculée, comme cela avait été introduit au chapitre précédent.

  La nouveauté majeure de ce chapitre réside dans le fait que, contrairement aux méthodes précédentes où la matrice de covariance était classifiée via un SVM ou une forêt aléatoire, on exploite ici SPDNet [12], un réseau de neurones conçu pour traiter directement les matrices définies positives. Ce réseau opère dans l'espace riemannien des matrices SPD, et nécessite donc une formulation spécifique des opérations différentiables ainsi que des règles de rétropropagation adaptées à cette géométrie. Le chapitre détaille ces étapes, ainsi que la fonction de perte utilisée.

  Enfin, les performances sont ensuite évaluées au travers de simulations comparatives avec les meilleurs modèles du chapitre 2, selon différents scénarios : faible nombre d'exemples d'entraînement, données mal étiquetées ou changement de distribution.
- Le **chapitre 4** constitue une extension naturelle du chapitre précédent, dans lequel nous avons proposé un modèle de classification robuste combinant un extracteur convolutif profond (ResNet-34), un pooling de covariance et une architecture SPDNet adaptée à la classification de matrices de covariance. Dans ce nouveau chapitre, l'objectif est d'aller au-delà de la simple classification : nous visons désormais la détection, la localisation et la classification conjointe des cibles dans les images GPR.

  Pour cela, nous nous appuyons sur l'architecture Faster R-CNN, un cadre de détection d'objets parmi les plus performants et les plus utilisés. Nous débutons par un rappel des architectures précédentes de type R-CNN, afin de mieux situer les apports de Faster R-CNN, notamment en termes d'intégration bout-à-bout (*end-to-end*) de la détection et de la classification. Un exemple détaillé de sa mise en œuvre avec ResNet-50 et FPN (Feature Pyramid Network) est ensuite présenté, suivi d'une explication des fonctions de perte spécifiques à cette architecture.

  La seconde partie du chapitre est consacrée à notre adaptation de ce cadre aux données GPR. Nous introduisons notre modèle RC R-CNN, qui associe Faster R-CNN à une étape de covariance pooling appliquée après l'empilement des régions d'intérêt (RoIs Align). Cette opération permet de capturer

des relations de second ordre entre les activations régionales. Les matrices de covariance ainsi obtenues sont ensuite normalisées à l'aide de l'opérateur MPNCov (Matrix Power Normalization Covariance), avant d'être classées par un bloc SPDNet, comme dans le chapitre précédent. Cette chaîne permet d'exploiter les propriétés géométriques des matrices SPD tout en restant intégrée dans un pipeline de détection.

Enfin, la dernière partie du chapitre présente les résultats obtenus avec ce modèle sur le jeu de données fourni par Géolithe. Plusieurs configurations expérimentales sont explorées : étude du prétraitement, choix des hyperparamètres, critères d'évaluation fondés sur les métriques COCO (Common Objects in Context), mais aussi robustesse face aux changements de distribution des données. Ces expérimentations permettent d'évaluer les apports de notre méthode dans un cadre plus réaliste de détection automatique d'objets enfouis dans des images radar.

- Enfin, le **chapitre 5** présente les conclusions de cette thèse et les perspectives pour la suite de ces travaux.

Les principales contributions sont les suivantes :

- Une méthode de classification robuste pour des images GPR bruitées, basée sur des descripteurs de covariance du second ordre ;
- L'intégration de ces descripteurs dans une architecture convolutive entraînée de bout en bout, via des couches SPD différentiables ;
- L'adaptation de ce principe à un détecteur d'objets de type Faster R-CNN, par opération de covariance ;
- **Trois publications scientifiques issues de ces travaux** :
  - Une communication au congrès GRETSI 2023 [16], portant sur une méthode de classification exploitant l'extraction de caractéristiques par MobileNetV2, suivie d'un pooling MPN-COV et d'un classifieur SPDNet ;
  - Une publication à la conférence IGARSS 2024 [17], décrivant l'architecture développée au chapitre 3, combinant ResNet-34 (RCNet ou SRCNet), le pooling de covariance et SPDNet, pour la classification d'objets enfouis ;
  - Un article de journal publié dans *IEEE J-STARS* [18] présente une version étendue du travail IGARSS, avec des résultats complets sur l'évaluation du modèle dans différentes configurations expérimentales.

Chapitre 1

Principe et nature des données en GPR

Le système de géoradar (GPR) repose sur l'émission et la réception d'ondes électromagnétiques pour sonder les sous-sols, mais les signaux recueillis sont souvent bruités, rendant l'interprétation des images complexe. Ce chapitre explore d'abord le fonctionnement de base du GPR, en détaillant les caractéristiques du signal émis et reçu, ainsi que les défis liés à l'analyse d'images GPR bruitées. Ensuite, il aborde les traitements classiques pour améliorer la qualité des images, que ce soit par des outils intégrés aux appareils ou développés par la communauté, avant de s'intéresser au débruitage, aux problèmes inverses en imagerie et aux méthodes statistiques pour détecter des objets. Enfin, il présente la problématique de cette thèse : la classification automatique des objets détectés dans les images GPR par apprentissage automatique. Pour répondre à cet enjeu, un jeu de données (dataset) spécifique a été créé à partir des données fournies par la société Géolithe. Afin d'améliorer la robustesse des méthodes face aux variations expérimentales (conditions d'acquisition, nature des sols, profondeur), une approche hybride combinant descripteurs de covariance et réseaux profonds a été proposée, conciliant la compacité des représentations et la performance.

Sommaire

1.1 Panorama général de l'imagerie radar

L'imagerie radar constitue un domaine clé de l'observation active des milieux, permettant la caractérisation des structures par l'analyse du signal électromagnétique réfléchi. Contrairement à l'imagerie passive (comme l'optique ou l'infrarouge), les systèmes radar émettent leurs propres ondes, ce qui les rend opérationnels, quelles que soient les conditions d'éclairage ou météorologiques.

1.1.1 Types de systèmes radar

Les radars se déclinent selon plusieurs configurations instrumentales, adaptées à des applications distinctes :

- **SAR (Synthetic Aperture Radar)** : utilisé en observation spatiale ou aérienne, il permet de reconstruire des images 2D haute résolution de la surface terrestre grâce au déplacement de l'antenne.
- **Radar MIMO (Multiple-Input Multiple-Output) et polarimétrique** : ces systèmes exploitent plusieurs antennes en émission et en réception (MIMO), ainsi que différentes polarisations, afin d'augmenter l'information spatiale et structurelle disponible pour l'analyse de la scène observée.
- **GPR (Ground Penetrating Radar)** : destiné à sonder les couches superficielles du sol, il permet la détection et la caractérisation d'objets ou de structures enfouies.

1.1.2 Principes communs et différences

Ces systèmes reposent sur des principes communs : l'émission d'une onde électromagnétique, son interaction avec le milieu, puis la réception et le traitement du signal. Ils se distinguent cependant par leurs fréquences d'émission, leur résolution spatiale et la nature du milieu observé.

1.1.3 Caractéristiques générales des images radar

De manière générale, les images radar présentent des caractéristiques communes :

- présence de bruit de speckle et d'interférences multiples,
- dépendance aux paramètres d'acquisition (fréquence, polarisation, géométrie),
- complexité de l'interprétation des signaux réfléchis.

L'imagerie radar repose sur des principes physiques communs, mais soulève des défis d'interprétation spécifiques selon la nature du milieu sondé. Le GPR se distingue par sa sensibilité aux conditions expérimentales et par la complexité des signaux produits.

Les parties suivantes détailleront ces aspects en présentant le **fonctionnement du GPR**, la **modélisation du signal** et les **traitements classiques appliqués aux radargrammes**. Cette base constituera le **fondement des méthodes d'apprentissage** développées par la suite pour la classification et la détection d'objets enfouis.

1.2 Principe du GPR

Dans cette section, nous présentons le principe d'acquisition des données GPR, en détaillant la forme du signal émis par le système ainsi que les paramètres physiques essentiels qui influencent la propagation des ondes dans le milieu sondé.

1.2.1 Principe d'acquisition

Le système d'acquisition utilise une antenne radar montée sur un chariot mobile, accompagnée d'une console d'enregistrement (ordinateur ou tablette). Ce dispositif, fonctionnant en mode monostatique lorsque l'émetteur et le récepteur partagent la même antenne, émet de brèves impulsions électromagnétiques à haute fréquence (10 MHz à 2 GHz). Le chariot assure un déplacement constant de l'antenne le long d'un profil d'acquisition, comme illustré sur la Figure 1.1.

FIGURE 1.1 – Exemple illustratif d'un système GPR déployé sur le terrain il s'agit du dispositif d'acquisition avec le 3D-radar de Géolithe. Cette image ne correspond pas au dispositif utilisé pour les campagnes décrites dans cette thèse, mais sert uniquement à visualiser un exemple de configuration typique.

Les ondes émises par l'antenne pénètrent dans le sol, se réfléchissent, se transmettent et/ou se réfractent aux interfaces séparant deux milieux aux propriétés électromagnétiques différentes. Ces signaux sont ensuite enregistrés à intervalles réguliers pour analyse, produisant un radargramme brut. Le trajet des ondes radar dans le sous-sol et les traces obtenues en déplaçant l'antenne le long du profil sont schématisés sur la Figure 1.2.

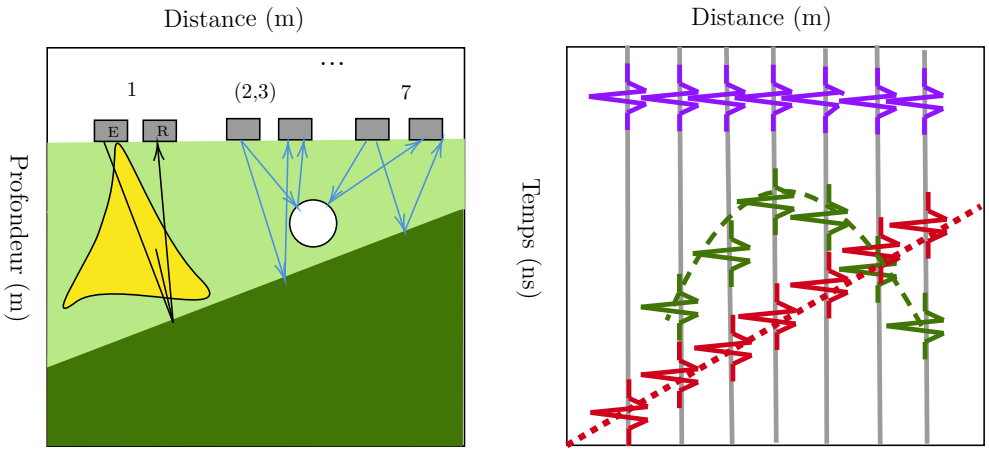


FIGURE 1.2 – Sur la Figure de gauche, le schéma illustre les trajets des ondes radar dans un sous-sol contenant deux objets de type abri en bois (en jaune) ainsi qu'un objet circulaire (en blanc). Sur la Figure de droite, les signaux mauve correspondent aux ondes directes, les verts à la réflexion sur l'objet circulaire et les rouges à la réflexion sur l'interface pentée.

Comme illustré dans la Figure 1.3, un **A-scan** est la représentation d'une trace ou forme d'onde géoradar en une dimension, c'est-à-dire l'amplitude du signal en fonction du temps à un endroit donné.

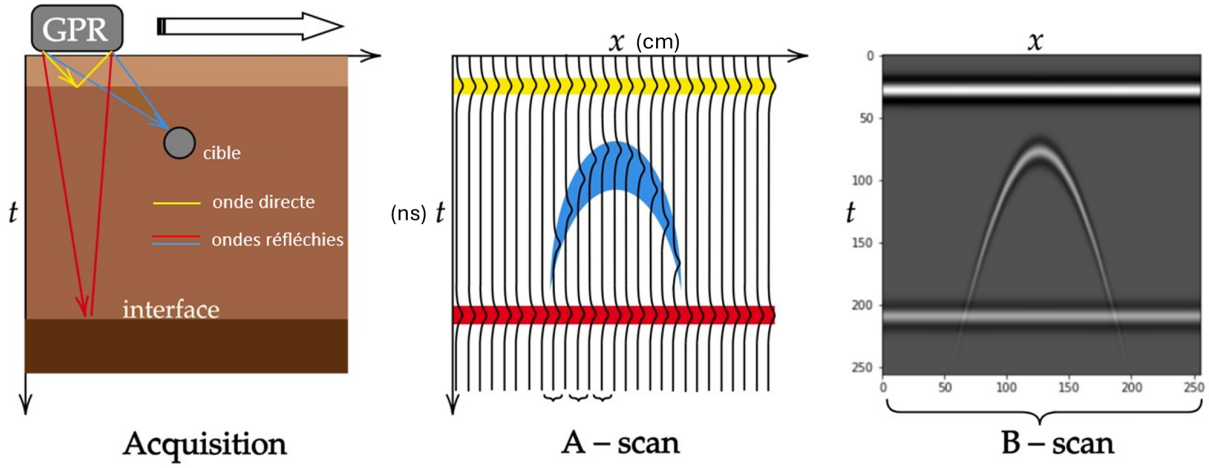


FIGURE 1.3 – Principe de fonctionnement du GPR. **(Gauche)** Schéma de l'acquisition : une antenne émet une onde électromagnétique dont une partie est réfléchie par des interfaces (couche géologique ou cible enfouie), tandis qu'une autre continue sa propagation. **(Centre)** Représentation des A-scans (signaux en fonction du temps t, pour chaque position x) montrant les échos reçus. **(Droite)** Le B-scan est l'image formée par l'empilement des A-scans le long de la trajectoire de l'antenne. L'axe vertical t correspond au temps aller-retour des ondes, exprimé en nanosecondes (ns), et l'axe horizontal x à la position de l'antenne lors du déplacement, exprimée en centimètres (cm). Une cible ponctuelle génère typiquement une hyperbole dans le B-scan.

Un ensemble de traces radar consécutives suivant une direction particulière représente un **B-scan (ou radargramme)**. Un B-scan permet donc d'obtenir une vue en deux dimensions dans la direction choisie. Si une valeur selon l'axe z est fixée comme constante.

Dans cette représentation bidimensionnelle, un objet se manifeste souvent sous la forme d'une trace

hyperbolique, et la forme du signal réfléchi varie selon les propriétés électromagnétiques et géométriques des couches de sol traversées.

Traditionnellement, le système est placé sur le sol. De plus en plus de systèmes sont placés sur un drone, et dans cette thèse, le géoradar (GPR) sera déployé en hauteur. Pour simuler le vol d'un drone, le système GPR sera surélevé à l'aide d'une plateforme.

1.2.2 Modélisation signal émis

Pour modéliser le signal émis par le radar à pénétration de sol (GPR), nous commençons par rappeler l'expression de l'ondelette de Ricker, illustrée sur la Figure 1.4. Cette modélisation est essentielle pour comprendre comment les signaux se propagent et interagissent avec les objets enfouis. Elle permet ainsi d'identifier, dans les signaux réfléchis enregistrés, ce qui relève du milieu traversé et ce qui est caractéristique d'une cible. Cela constitue une étape clé pour la suite de cette thèse, notamment pour la classification automatique basée sur la forme des échos.

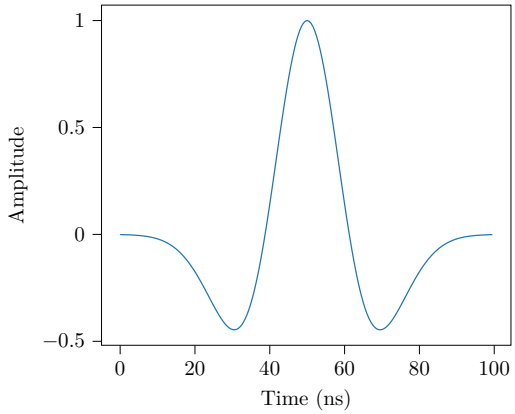


FIGURE 1.4 – Signal typique émis par un radar GPR impulsionnel : ondelette de Ricker. L'axe des abscisses représente le temps en nanosecondes (ns) et l'axe des ordonnées l'amplitude (unité arbitraire).

La Figure 1.4 représente le signal transmis par le GPR [19, 20], dont l'expression est la suivante :

$$e(t \mid \omega_0) = \left(1 - \frac{\omega_0^2 t^2}{2}\right) \exp\left(-\frac{\omega_0^2 t^2}{4}\right), \tag{1.1}$$

où :

- t est le temps (en secondes), représentant la variation du signal au cours du temps ;
- ω_0 est la fréquence angulaire (ou pulsation centrale, en rad/s) du radar.

Dans la Figure 1.4, on observe une "polarisation" typique de type $-+-$, avec un pic négatif suivi d'un pic positif, puis d'un autre pic négatif. Cette forme caractéristique peut être altérée ou inversée dans les signaux réfléchis, selon les propriétés électromagnétiques du sol ou des objets traversés. Cette variabilité dans les échos constitue une information précieuse pour la détection et la classification des cibles.

1.2.3 Paramètres physiques qui influencent la propagation de l'onde

Le sol est modélisé comme étant constitué de L couches successives de matériaux différents, chacune caractérisée par ses propres propriétés électromagnétiques (permittivité, conductivité, perméabilité). Ces propriétés influencent la propagation de l'onde électromagnétique en modifiant sa vitesse, son atténuation et sa direction selon les interfaces rencontrées.

Permittivité, conductivité et perméabilité

Les propriétés électromagnétiques d'un matériau sont définies par trois paramètres fondamentaux : la permittivité (ε), la conductivité (σ) et la perméabilité (μ). Ces paramètres influencent la manière dont les ondes électromagnétiques interagissent avec le matériau.

La **permittivité** (ε), exprimée en farads par mètre (F/m), mesure la capacité d'un matériau à se polariser sous l'influence d'un champ électrique. Elle est reliée à la permittivité du vide $\varepsilon_0 = 8.854 \times 10^{-12}\,\mathrm{F/m}$ et à la permittivité relative ε_r par la relation suivante :

$$\varepsilon = \varepsilon_r \varepsilon_0.$$

La permittivité relative (ε_r) est sans unité et caractérise la réponse électrique du matériau par rapport à celle du vide. Les matériaux présentant une permittivité élevée ralentissent davantage les ondes électromagnétiques et augmentent leur capacité de stockage d'énergie, comme nous le verrons.

La **conductivité** (σ), exprimée en siemens par mètre (S/m), caractérise la facilité avec laquelle les charges électriques se déplacent dans un matériau sous l'effet d'un champ électrique. La conductivité joue un rôle crucial dans l'atténuation des ondes électromagnétiques. Une conductivité élevée entraîne une forte dissipation d'énergie, limitant la pénétration des ondes dans le matériau. Cette propriété est particulièrement importante dans les sols saturés ou les milieux conducteurs.

Enfin, la **perméabilité** (μ), exprimée en henrys par mètre (H/m), mesure la capacité d'un matériau à soutenir un champ magnétique induit. La perméabilité absolue est liée à la perméabilité relative (μ_r) et à la perméabilité du vide ($\mu_0 = 4\pi \times 10^{-7}\,\mathrm{H/m}$) par la relation :

$$\mu = \mu_r \mu_0.$$

La perméabilité relative (μ_r) est une grandeur sans unité qui décrit les propriétés magnétiques d'un matériau par rapport à celles du vide. Bien que la perméabilité soit importante dans les matériaux ferromagnétiques, elle a peu d'effet dans les milieux non magnétiques, comme la plupart des sols naturels. Dans ce cas, on l'approche généralement par $\mu_r \approx 1$, ce qui revient à considérer la perméabilité du vide μ_0. Cette approximation est couramment admise dans le domaine du radar GPR [2, 21].

En résumé, les paramètres ε, σ et μ définissent la réponse électrique et magnétique d'un matériau, influençant la vitesse, l'atténuation et la réflexion des ondes électromagnétiques qui s'y propagent.

Le tableau 1.1 présente les caractéristiques électromagnétiques (permittivité diélectrique relative et conductivité électrique) de plusieurs matériaux utilisés pour la modélisation de la tranchée de sable humide (Voir 1.6.1 plus loin) . Les valeurs ont été fournies par Géolithe sur la base de mesures ou d'estimations internes.

Matériau	**Permittivité relative ε_r**	**Conductivité σ [S/m]**
Sable humide	12	1×10^{-7}
Air	1	1×10^{-12}
Bois	3	1×10^{-3}
Métal	∞	1×10^{-12}
PVC	4	5×10^{-3}
Résine	5	5×10^{-18}
Béton	6	5×10^{-3}
Cire	2.5	1×10^{-3}
Eau	81	0.02

TABLE 1.1 – Caractéristiques électromagnétiques (permittivité relative ε_r et conductivité σ) des matériaux utilisés par Géolithe pour la modélisation de la tranchée de sable humide.

Vitesse de propagation dans un milieu quelconque La vitesse de propagation v d'une onde électromagnétique dans un matériau linéaire, isotrope et homogène est donnée par l'expression générale suivante :

$$v = \sqrt{\frac{2}{\mu\varepsilon}} \left[\left(1 + \left(\frac{\sigma}{\omega\varepsilon} \right)^2 \right)^{1/2} + 1 \right]^{-1/2} \tag{1.2}$$

où $\omega = 2\pi f$ est la pulsation angulaire de l'onde électromagnétique avec f la fréquence de l'onde.

Approximation pour les conditions du GPR Dans le contexte du GPR, les milieux sont généralement faiblement conducteurs, les signaux GPR sont caractérisés comme étant à haute fréquence. On se place donc dans le régime :

$$\sigma \ll \omega\varepsilon$$

En posant $\delta = \dfrac{\sigma}{\omega\varepsilon}$, avec $\delta \ll 1$, on peut développer la racine :

$$\sqrt{1+\delta^2} = 1 + \frac{1}{2}\delta^2 + \mathcal{O}(\delta^4)$$

D'où :

$$v \approx \sqrt{\frac{2}{\mu\varepsilon}} \left[1 + \frac{1}{2}\delta^2 + 1 \right]^{-1/2} \tag{1.3}$$

$$= \sqrt{\frac{2}{\mu\varepsilon}} \left[2 + \frac{1}{2}\delta^2 \right]^{-1/2} \tag{1.4}$$

$$\approx \sqrt{\frac{1}{\mu\varepsilon}} \left(1 - \frac{1}{8}\delta^2 \right) \tag{1.5}$$

En négligeant le terme correctif, on retrouve l'approximation usuelle :

$$v \approx \frac{1}{\sqrt{\mu\varepsilon}} = \frac{c}{\sqrt{\mu_r \varepsilon_r}} \tag{1.6}$$

Cas d'un milieu non magnétique Dans les matériaux non magnétiques usuels, $\mu_r = 1$, ce qui donne l'approximation :

$$v \approx \frac{c}{\sqrt{\varepsilon_r}} \tag{1.7}$$

Exemples numériques - Dans l'air ($\varepsilon_r \approx 1$) : $v \approx 0{,}3\,\text{m/ns}$ - Dans l'eau ($\varepsilon_r \approx 81$) : $v \approx 0{,}03\,\text{m/ns}$

Cas limite d'un conducteur parfait Lorsque $\sigma \gg \omega\varepsilon$, comme dans les métaux, la vitesse tend vers zéro. L'onde est alors fortement réfléchie et ne pénètre pratiquement pas dans le matériau.

Atténuation

La constante d'atténuation α intervient dans la relation suivante, qui décrit la diminution de l'amplitude d'une onde électromagnétique après avoir parcouru une distance z :

$$\frac{|\mathbf{A}(z)|}{|\mathbf{A}_0|} = e^{-\alpha z} \tag{1.8}$$

où $\mathbf{A}_0$ est l'amplitude initiale de l'onde, $\mathbf{A}(z)$ son amplitude après propagation, et α le coefficient d'atténuation caractéristique du milieu traversé.

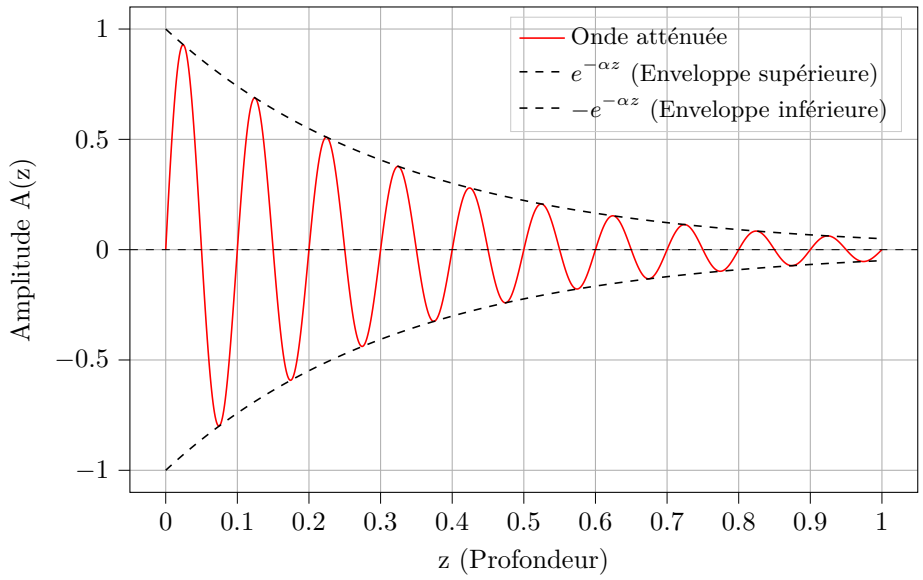


FIGURE 1.5 – Atténuation exponentielle d'une onde avec une amplitude initiale $A_0 = 1$ et un coefficient d'atténuation $\alpha = 3$.

Comme illustré dans la Figure 1.5, plus α est élevé, plus l'amplitude décroît rapidement. Lorsque $z \to \infty$, l'amplitude tend vers zéro. Toujours dans les conditions ($\sigma \ll \varepsilon\omega$) du GPR, pour une onde électromagnétique ayant parcouru une distance z, la constante d'atténuation α est approximée par :

$$\alpha = \omega\sqrt{\frac{\mu\varepsilon}{2}}\left[\sqrt{1+\left(\frac{\sigma}{\omega\varepsilon}\right)^2}-1\right]^{1/2} \approx \frac{\sigma}{2}\sqrt{\frac{\mu}{\varepsilon}} \tag{1.9}$$

Justifions cette approximation en supposant que le rapport $\delta = \dfrac{\sigma}{\omega\varepsilon}$ est très petit ($\delta \ll 1$), ce qui est typiquement le cas dans les applications GPR.

On commence par poser :

$$\alpha = \omega\sqrt{\frac{\mu\varepsilon}{2}}\left[\sqrt{1+\delta^2}-1\right]^{1/2}$$

Développons $\sqrt{1+\delta^2}$ en série de Taylor autour de $\delta = 0$:

$$\sqrt{1+\delta^2} = 1 + \frac{1}{2}\delta^2 - \frac{1}{8}\delta^4 + \cdots$$

En négligeant les termes d'ordre supérieur à δ^2, on obtient :

$$\sqrt{1+\delta^2} - 1 \approx \frac{1}{2}\delta^2$$

On remplace cette approximation dans l'expression de α :

$$\alpha \approx \omega\sqrt{\frac{\mu\varepsilon}{2}}\left(\frac{1}{2}\delta^2\right)^{1/2} = \omega\sqrt{\frac{\mu\varepsilon}{2}} \cdot \frac{\delta}{\sqrt{2}}$$

En remplaçant $\delta = \dfrac{\sigma}{\omega\varepsilon}$, il vient :

$$\alpha \approx \omega\sqrt{\frac{\mu\varepsilon}{2}} \cdot \frac{1}{\sqrt{2}} \cdot \frac{\sigma}{\omega\varepsilon} = \frac{\sigma}{2}\sqrt{\frac{\mu}{\varepsilon}}$$

Ainsi, dans le régime faiblement conducteur, on obtient bien l'approximation :

$$\boxed{\alpha \approx \frac{\sigma}{2}\sqrt{\frac{\mu}{\varepsilon}}}$$

Ce résultat montre que, dans ce cas, l'atténuation est indépendante de la fréquence ω et proportionnelle à la conductivité électrique σ du milieu.

Les matériaux à forte permittivité et perméabilité, comme l'eau, absorbent davantage d'énergie, ce qui augmente l'atténuation. Dans les matériaux hautement conducteurs (ex. métaux), la conductivité entraîne une forte atténuation et limite la pénétration des ondes, qui sont principalement réfléchies.

En particulier, les matériaux à forte conductivité absorbent davantage l'énergie des ondes en raison des pertes par effet Joule. De même, les matériaux à forte permittivité peuvent ralentir et atténuer les ondes électromagnétiques, bien que l'absorption réelle dépende aussi de la conductivité. Enfin, dans certains cas spécifiques, des matériaux à forte perméabilité peuvent également contribuer à des pertes d'énergie, notamment dans des champs magnétiques alternatifs.

Dans un milieu avec des sols conducteurs dans ces même conditions ($\sigma \ll \varepsilon\omega$), l'atténuation en dB/m peut être approximée par [22] :

$$\alpha \approx 1690\frac{\sigma}{\sqrt{\varepsilon_r}} \quad \text{(en dB/m)} \tag{1.10}$$

L'atténuation influe directement sur la profondeur jusqu'à laquelle les ondes radar peuvent être efficacement transmises et reçues. Une conductivité élevée réduit la profondeur d'investigation, rendant le matériau presque totalement réfléchissant.

Profondeur d'investigation

La profondeur d'investigation est liée à l'atténuation et dépend de la fréquence de l'antenne et des propriétés du sol. L'épaisseur de peau δ, qui représente la distance à laquelle l'onde est atténuée d'un facteur e^{-1}, c'est-à-dire de 37% par rapport à son amplitude initiale , est définie par :

$$\delta = \frac{1}{\alpha} \tag{1.11}$$

La profondeur d'investigation peut être approximée [23] par :

$$D = 3\delta \approx 0.0159\frac{\varepsilon_r}{\sigma} \quad \text{si } \sigma \ll \varepsilon\omega \tag{1.12}$$

Cette relation montre que la profondeur de peau atteint une limite qui ne dépend pas de la fréquence dans les conditions $\sigma \ll \varepsilon\omega$ et que la conductivité élevée réduit la pénétration des ondes dans le sol. Des sols conducteurs seront très absorbants et la profondeur d'investigation sera faible, elle sera plus grande pour les matériaux ayant des permittivités diélectriques plus élevées.

Dans la Figure 1.6 on montre un exemple comparant l'atténuation des ondes radioélectriques dans l'air et dans un milieu conducteur.

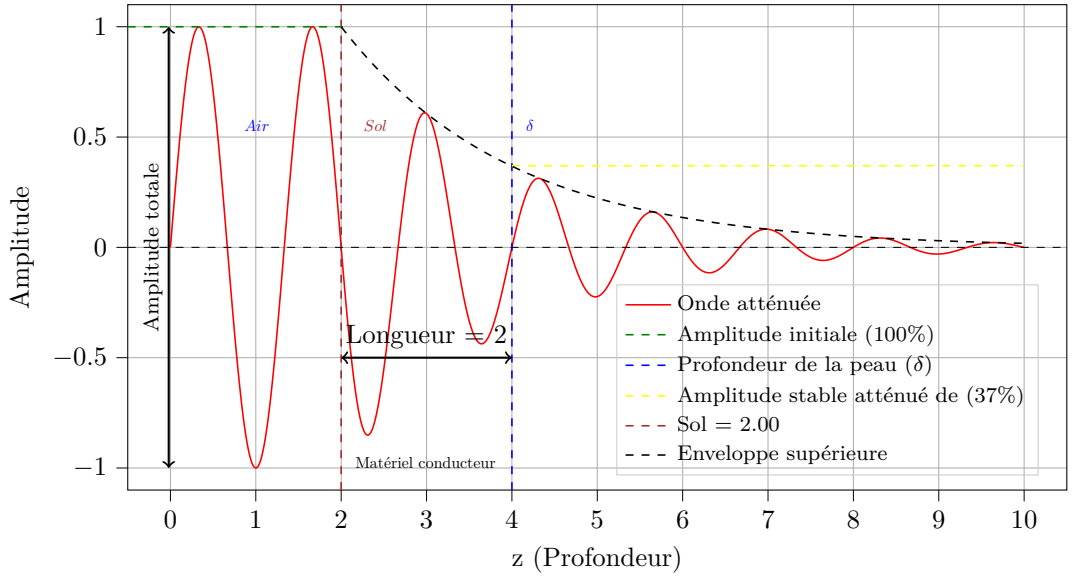


FIGURE 1.6 – Atténuation d'ondes électromagnétiques dans l'air (ligne verte pointillée, amplitude constante) et dans un matériau conducteur (ligne en pointillé du sol à la ligne bleue de la profondeur de peau, atténuation exponentielle). La profondeur de peau δ (trait vertical) indique la distance à laquelle l'amplitude dans le conducteur chute à 37% $(1/e)$ de sa valeur initiale. L'enveloppe en trait discontinu illustre cette décroissance caractéristique pour $\alpha = \frac{1}{2}$. Contrairement à l'air où l'onde se propage sans perte, le milieu conducteur dissipe l'énergie par courants induits.

Toutefois, atteindre de grandes profondeurs ne garantit pas nécessairement une bonne qualité d'image : la résolution devient alors un facteur essentiel.

Résolution

La résolution détermine la plus petite distance entre deux objets distinguables sur un radargramme. Elle dépend de la fréquence de l'antenne et des propriétés physiques du sol.

Résolution verticale La résolution verticale est la capacité à distinguer deux objets ou interfaces adjacents en profondeur. Elle est inversement proportionnelle à la fréquence centrale de l'antenne. On considère que la résolution verticale r_{vert} est comprise entre un quart et une moitié de la longueur d'onde λ.

$$r_{\text{vert}} \in \left[\frac{\lambda}{4}, \frac{\lambda}{2}\right] \quad \text{où} \quad \lambda = \frac{v}{f_c} \tag{1.13}$$

v est la vitesse des ondes radar dans le milieu et f_c la fréquence centrale de l'antenne.

Le tableau 1.2 présente la période du pulse (inverse de la fréquence centrale), la longueur d'onde et les résolutions attendues pour trois fréquences centrales, dans du sable (permittivité diélectrique relative de 7).

	200 MHz	**350 MHz**
Période du pulse (ns)	5	3
Longueur d'onde (m)	0,6	0,3
Résolution minimale (cm)	14	8
Résolution maximale (cm)	28	16

TABLE 1.2 – Exemples de résolution verticale théorique pour des antennes radar de 200 MHz et 350 MHz de Géolithe dans du sable sec ($\varepsilon_r = 7$). On y indique la période du pulse, la longueur d'onde associée, ainsi que les bornes minimale et maximale de la résolution verticale estimée par $\lambda/4$ et $\lambda/2$.

Ce tableau permet d'illustrer le lien entre la fréquence centrale de l'antenne et la finesse de détection verticale. Une fréquence plus élevée entraîne une longueur d'onde plus courte, et donc une résolution verticale plus fine. Cependant, ce gain en précision se fait généralement au détriment de la profondeur d'investigation, en raison d'une atténuation plus importante des ondes hautes fréquences. Il s'agit donc d'un compromis classique dans les études GPR : choisir une fréquence adaptée à l'objectif d'imagerie, qu'il s'agisse de détecter des structures profondes ou de discriminer des objets rapprochés.

Dans la pratique, les résolutions verticales seront moins bonnes, en particulier à cause du couplage entre le sol et l'antenne qui altère la forme et l'amplitude du train d'onde [22].

Résolution horizontale La résolution horizontale d'un radar dépend de la **zone de Fresnel**, qui représente la région autour de la trajectoire de propagation de l'onde où l'interférence constructive peut se produire, permettant ainsi de détecter des objets qui se trouvent côte à côte dans le plan horizontal (c'est-à-dire sur le sol). Une résolution plus élevée permet de distinguer des détails plus fins et d'identifier plus précisément des éléments proches les uns des autres. Cette résolution est approximée par l'équation suivante, qui calcule la largeur de la zone de Fresnel avec des objets à une profondeur z par [24] :

$$r_{\text{hor}} \approx \sqrt{\frac{\lambda z}{2} + \frac{\lambda^2}{16}} \tag{1.14}$$

où λ est la longueur d'onde associée à la fréquence centrale de l'onde électromagnétique. Cette formule montre que la largeur de la zone de Fresnel dépend de la distance z et de la fréquence utilisée : une fréquence plus élevée entraîne une longueur d'onde plus courte, améliorant ainsi la précision de la détection.
Outre la résolution spatiale, la manière dont les ondes interagissent avec les interfaces dépend également de leur polarisation, ce qui influence fortement la nature des réflexions observées dans les données radar.

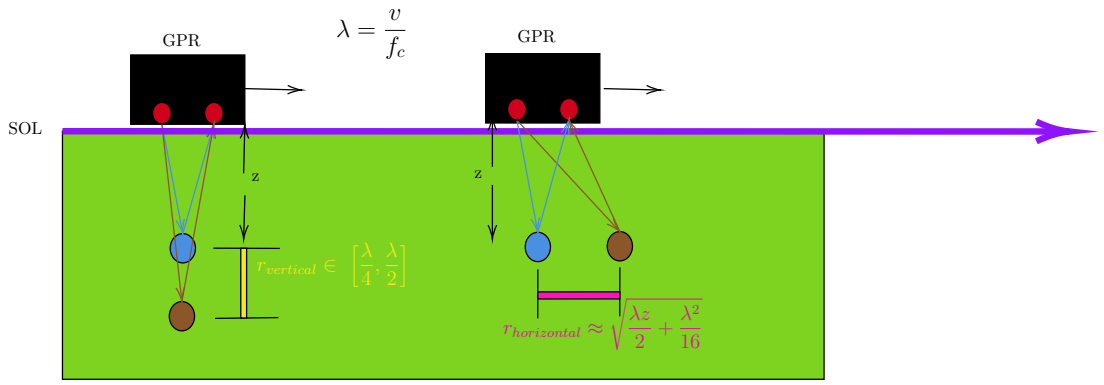


FIGURE 1.7 – Schéma des résolutions GPR verticale (gauche) et horizontale (droite).

Dans cette section, nous avons rappelé les principaux paramètres physiques qui conditionnent la propagation d'une onde radar dans un milieu complexe comme le sous-sol. La fréquence (ou longueur d'onde) influe directement sur la résolution et la pénétration : des fréquences plus élevées permettent une meilleure résolution mais une moindre pénétration. La permittivité diélectrique du milieu joue également un rôle fondamental en affectant la vitesse de propagation, l'impédance et la réfraction de l'onde. Nous avons également introduit la notion de zone de Fresnel, qui dépend de la distance de propagation et de la longueur d'onde centrale, et qui délimite la région principalement responsable des réflexions. Enfin, la polarisation du signal radar conditionne les types de réflexions observées, en fonction de la géométrie des interfaces rencontrées. Ces paramètres sont essentiels pour comprendre la manière dont les signaux radar sont formés et interprétés.

1.2.4 Polarité et réflexions aux interfaces

La polarité du signal électromagnétique joue un rôle important dans les interactions entre le signal et le milieu géologique. Dans le contexte du GPR, la polarisation peut être verticale, horizontale ou circulaire, affectant différemment la propagation et la réflexion des ondes [23] :

- **Réflectivité des diffuseurs :** La sensibilité des diffuseurs au signal dépend de leur orientation et de la polarisation du signal. Par exemple, certains diffuseurs peuvent être plus détectables avec une polarisation verticale qu'avec une polarisation horizontale.
- **Effet aux interfaces :** À chaque interface entre deux couches de permittivités différentes, une partie du signal est réfléchie et une autre est transmise. Ce phénomène est décrit par les coefficients de Fresnel, qui dépendent de l'angle d'incidence et des propriétés électromagnétiques des couches. Le coefficient de réflexion, noté R, exprime la proportion d'onde réfléchie et indique si l'onde conserve ou inverse sa polarité :

$$R = \frac{\sqrt{\varepsilon_1} - \sqrt{\varepsilon_2}}{\sqrt{\varepsilon_1} + \sqrt{\varepsilon_2}} \tag{1.15}$$

Si $R > 0$, l'onde réfléchie conserve sa polarité ; si $R < 0$, elle inverse sa polarité. Par exemple, lors du passage d'une onde de l'air ($\varepsilon_r \approx 1$) à l'eau ($\varepsilon_r \approx 81$), il y a un fort contraste de permittivité, pouvant entraîner une inversion de polarité. Le coefficient de transmission T, quant à lui, détermine la proportion d'énergie transmise et est donné par :

$$T = \frac{2\sqrt{\varepsilon_1}}{\sqrt{\varepsilon_1} + \sqrt{\varepsilon_2}} \tag{1.16}$$

La Figure 1.8 nous montre un exemple.

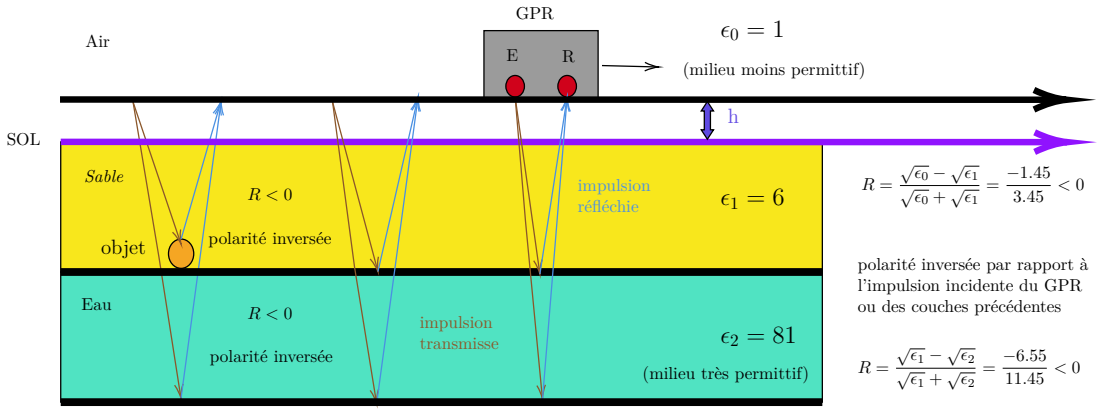


FIGURE 1.8 – Exemple sur les réflexions aux interfaces et la polarité.

Dans cette section, nous avons exploré les principes physiques de la propagation des ondes électromagnétiques dans le cadre du GPR. Ces bases nous permettent maintenant de passer à la modélisation du signal, qui repose directement sur ces phénomènes de propagation.

1.3 Modélisation du signal reçu réfléchi sur les sols et cibles

Dans cette section, nous développons le modèle de propagation des ondes dans un radar à pénétration de sol (GPR), en tenant compte des propriétés du sol, de la polarisation du signal, et nous donnerons des exemples concrets d'images GPR. Nous présentons ensuite l'expression complète du signal total reçu par le radar.

1.3.1 Modélisation du signal réfléchi par une interface de couche continue

Lorsqu'une onde émise par le GPR rencontre une interface continue, telle que la limite entre deux couches homogènes dans le sol, la réflexion de l'onde produit une signature particulière sur le radargramme. En effet, une couche étendue génère un écho qui apparaît sous la forme d'un **trait continu horizontal**. Ce trait est le résultat de la réflexion uniforme de l'onde le long de la surface de l'interface.

Pour une interface homogène parallèle au sol :

- Si la permittivité des deux couches diffère significativement, l'onde est partiellement réfléchie avec une amplitude importante, formant une ligne horizontale intense sur le radargramme.
- Si la couche est inclinée ou irrégulière, le trait apparaîtra incliné ou déformé suivant la géométrie de l'interface.

Ainsi, les stratifications du sol apparaissent souvent sous forme de traits continus sur le radargramme, facilitant l'identification des différentes couches. Il s'agit notamment de l'onde directe se propageant le long de chaque interface homogène dans le sous-sol.

1.3.2 Modèle de propagation des ondes sur les cibles

Dans cette étude, nous considérons un radar GPR (Ground Penetrating Radar) évoluant selon une configuration spécifique : le radar se déplace le long de l'axe u, parallèle au sol (axe y), mais à une hauteur h significative au-dessus de la surface du sol.

Cette disposition dite « surélevée » n'est pas classique en GPR, où les antennes sont généralement placées au contact direct du sol. Ce choix expérimental, qui sera justifié par la nature des mesures et des contraintes du site, a le mérite d'être souligné dès le départ car il influence notablement la modélisation

de la propagation des ondes et l'interprétation des données.

À chaque position u_m, le radar émet un signal $e(t)$ dans le sol. Ce signal est réfléchi par P diffuseurs correspondant à nos cibles enfouies, chacune caractérisée par un coefficient de réflexion $a_p \in [-1;1], p \in [1;P]$, représentant la part du signal rétrodiffusé vers le radar.

Nous considérons que le radar traverse L couches dans le sol, ainsi qu'une couche d'air au-dessus. Les étapes de ce processus sont les suivantes :

- **Écho du signal réfléchi :** L'écho du signal provenant d'un diffuseur p, situé en (y_p, z_p) et reçu par le radar, s'exprime comme une version retardée dans le temps du signal émis :

$$s_{m,p}(t) = a_p e\left(t - \tau_m\left(y_p, z_p, \epsilon_l\right)\right), \tag{1.17}$$

  où τ_m est le temps de propagation du signal entre le radar et le diffuseur, et retour.

- **Temps de propagation :** Pour une configuration générale où le sol est composé de L couches, chacune de constante diélectrique ϵ_l, et où le radar n'est pas en contact direct avec le sol, $\tau_m(y,z)$ est donnée par :

$$\tau_m(y,z) = \frac{2}{c_0}\left(d_{m,\text{air}}(y,z) + \sum_{l=1}^{L}\sqrt{\epsilon_l} d_{m,l}(y,z)\right), \tag{1.18}$$

  où $d_{m,l}$ et $d_{m,\text{air}}$ représentent respectivement les distances parcourues (aller simple) par l'onde à travers la couche l et dans l'air et où c_0 est la vitesse de la lumière dans le vide, approximativement 3×10^8 m/s.
  Remarque : Cette expression permet de prendre en compte explicitement les changements de milieu à travers les différentes couches du sol. En effet, les constantes diélectriques ϵ_l influencent la vitesse de propagation de l'onde dans chaque couche via la relation $v_l = \dfrac{c_0}{\sqrt{\epsilon_l}}$. Ainsi, le modèle rend compte des variations du temps de trajet dues à la nature hétérogène du sous-sol.

- **Signature des cibles ponctuelles (hyperboles) :** Les objets ponctuels ou petits obstacles enfouis, tels que des tuyaux ou des roches, produisent des hyperboles sur le radargramme. Ces hyperboles sont dues à la variation de la distance entre l'antenne et la cible en fonction de la position horizontale du radar u_m. Plus la cible est éloignée, plus le sommet de l'hyperbole est situé en profondeur dans le radargramme, reflétant l'éloignement en distance.

- **Signal total**

  Le signal total reçu par le radar à la position u_m est la somme de tous les échos provenant des P diffuseurs, incluant à la fois les réflexions continues et les réflexions ponctuelles. Il s'exprime comme suit :

$$s_m(t) = \sum_{p=1}^{P} s_{m,p}(t) = \sum_{p=1}^{P} a_p\, e\left(t - \tau_m\left(y_p, z_p, \epsilon_l'\right)\right), \tag{1.19}$$

  où le premier terme représente les contributions des réflexions continues (couches de sol) et le second terme regroupe les réflexions ponctuelles, produisant des hyperboles dans le radargramme. Ainsi, à chaque position u_m, le radar reçoit la somme de tous les échos provenant des P diffuseurs, chacun ayant son propre retard temporel et son coefficient de réflexion, tout en tenant compte des différentes couches dans le sol et de l'air au-dessus. Un exemple est illustré sur la Figure 1.9.

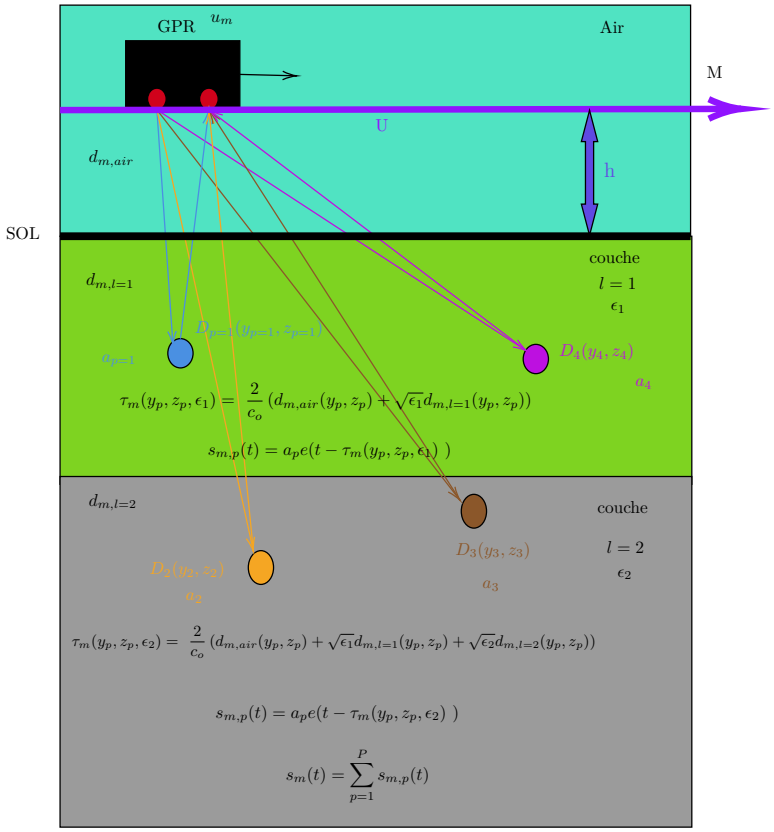


FIGURE 1.9 – Schéma des trajets des ondes radar dans le sous-sol. Les signaux en mauve correspondent aux ondes directes, en vert aux réflexions sur des cibles ponctuelles (produisant des hyperboles sur le radargramme) et en rouge aux réflexions sur des interfaces planes inclinées (produisant des traits continus).

En notant donc $s(x,t)$ le signal total $s_m(t)$, le modèle complet du signal inclut également le désordre $(cl(x,t))$ et le bruit $(n(x,t))$, selon l'expression :

$$y(x,t) = s(x,t) + cl(x,t) + n(x,t), \tag{1.20}$$

où :

- $s(x,t)$ regroupe les contributions des cibles souterraines, modélisées comme des hyperboles avec le coefficient associé déterminant leur amplitude pour chaque diffuseur p
- $cl(x,t)$ représente les réflexions des couches de sol et autres artefacts d'acquisition ;
- $n(x,t)$ correspond au bruit aléatoire des mesures.

Explication de l'expression du signal total

- Chaque terme dans la somme représente un écho provenant d'un diffuseur p, situé en (y_p, z_p), et se manifestant par une version retardée du signal émis $e(t)$.
- Le retard temporel $\tau_m(y_p, z_p)$ dépend des distances dans chaque couche ainsi que des vitesses de propagation dans chaque milieu.
- Le coefficient a_p module l'amplitude de chaque écho en fonction des propriétés du diffuseur.

Ainsi, à chaque position u_m, le radar reçoit la somme de tous les échos provenant des P diffuseurs, chacun avec son propre retard temporel et son coefficient de réflexion, en tenant compte des différentes couches dans le sol et de l'air au-dessus.

Passage à la pratique : discrétisation temporelle du signal En pratique, le signal $s_m(t)$ n'est pas observé de manière continue dans le temps, mais est **échantillonné** à une fréquence d'acquisition donnée, notée f_s. Ainsi, pour chaque position u_m du radar, on obtient un *vecteur de mesures discrètes* :

$$s_m[n] = s_m(nT_s) = \sum_{p=1}^{P} a_p \, e(nT_s - \tau_m(y_p, z_p, \epsilon'_l)), \tag{1.21}$$

où :

- $T_s = 1/f_s$ est la période d'échantillonnage ;
- $n \in \{0, \ldots, N-1\}$ désigne l'indice temporel des échantillons.

Ce passage du signal continu au signal discrétisé est fondamental pour le traitement numérique des données. En effet, les signaux échantillonnés $s_m[n]$ pour les différentes positions u_m sont ensuite assemblés pour former une image bidimensionnelle appelée *radargramme*, où l'axe horizontal correspond à la position du radar et l'axe vertical au temps d'acquisition (lié à la profondeur apparente).

Remarque : La fréquence d'échantillonnage f_s doit être suffisamment élevée pour satisfaire le critère de Nyquist [1], c'est-à-dire permettre une représentation correcte du signal réfléchi sans aliasing.

1.3.3 Modélisation du signal total sous forme vectorielle et matricielle

Dans le cadre de l'imagerie GPR, nous cherchons à reconstruire une scène souterraine à partir des signaux mesurés par un radar à pénétration de sol. La scène imagée est divisée en $N_y \times N_z$ pixels, chacun étant un diffuseur potentiel. Les signaux reçus sont observés sur N_T échantillons de temps discrets $t_i, i \in [\![1; N_T]\!]$.

Pour modéliser cette scène, nous définissons un vecteur $\boldsymbol{a} \in \mathbb{R}^{N_y N_z}$ contenant tous les coefficients de réflexion des pixels, ordonnés lexicographiquement :

$$\boldsymbol{a} = \left[a_1, \ldots, a_p, \ldots, a_{N_y N_z}\right]^T.$$

Forme vectorielle du signal reçu.

Pour chaque position u_m du radar, le signal échantillonné $\boldsymbol{s}_m$ est relié au vecteur $\boldsymbol{a}$ via une matrice de propagation $\boldsymbol{H}_m \in \mathbb{R}^{N_T \times N_y N_z}$, définie par :

$$[\boldsymbol{H}_m]_{ip} = e\left(t_i - \tau_m\left(y_p, z_p\right)\right), \tag{1.22}$$

où $\tau_m(y_p, z_p)$ représente le temps de retard pour une onde réfléchie par le point (y_p, z_p). Le signal reçu à la position m est donné par :

$$\boldsymbol{s}_m = \boldsymbol{H}_m \boldsymbol{a}. \tag{1.23}$$

En regroupant les matrices $\boldsymbol{H}_m$ de (1.23) et les vecteurs $\boldsymbol{s}_m$ sur toutes les positions u_m, nous obtenons une matrice globale $\boldsymbol{H} \in \mathbb{R}^{N_T M \times N_y N_z}$ et un vecteur global $\boldsymbol{s} \in \mathbb{R}^{N_T M}$:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_1^T & \cdots & \boldsymbol{H}_M^T \end{bmatrix}^T, \quad \boldsymbol{s} = \begin{bmatrix} \boldsymbol{s}_1^T & \cdots & \boldsymbol{s}_M^T \end{bmatrix}^T. \tag{1.24}$$

Ainsi, le problème peut s'écrire de manière compacte :

$$\boldsymbol{s} = \boldsymbol{H}\boldsymbol{a}. \tag{1.25}$$

Puis en incluant les couches et le bruit, le modèle peut s'écrire sous forme vectorielle :

$$\underline{y} = \underline{\boldsymbol{H}\boldsymbol{a}} + \underline{l} + \underline{n} \tag{1.26}$$

1. Le critère de Nyquist stipule que la fréquence d'échantillonnage f_s doit être au moins deux fois supérieure à la plus haute fréquence présente dans le signal, soit $f_s \geq 2f_{\max}$. Si cette condition n'est pas respectée, des phénomènes d'*aliasing* peuvent survenir : des hautes fréquences se replient et apparaissent comme de fausses basses fréquences dans le signal échantillonné, rendant l'interprétation des données erronée.

Forme matricielle globale (modèle convolutionnel)

Les modèles précédents représentent fidèlement le processus physique de propagation et de réflexion des ondes radar à travers une scène discrétisée. Toutefois, dans une optique de traitement du signal et d'inversion, ces modèles restent coûteux à manipuler numériquement, notamment à cause de la complexité des temps de propagation $\tau_m(y_p, z_p)$ pour chaque paire radar-pixel.

Pour simplifier l'analyse tout en capturant les phénomènes essentiels visibles sur les données (notamment les hyperboles), une approximation convolutionnelle est souvent adoptée. Celle-ci repose sur deux hypothèses principales :

- les cibles ponctuelles (diffuseurs) génèrent des signatures locales similaires, centrées à des positions différentes (translation spatiale) ;
- la forme des hyperboles peut être représentée à l'aide d'un dictionnaire d'atomes $\mathbf{H}_k$, chacun étant associé à une cible potentielle.

Sous ces hypothèses, les données radar peuvent être modélisées comme une superposition de motifs hyperboliques convolués avec des cartes de présence des cibles :

$$\mathbf{Y} = \sum_{k=1}^{K} \mathbf{C}_k \circledast \mathbf{H}_k + \mathbf{L} + \mathbf{N}, \tag{1.27}$$

où :

- $\mathbf{Y} \in \mathbb{R}^{N_T \times M}$ représente les données radar (B-scan) ;
- chaque $\mathbf{H}_k \in \mathbb{R}^{N_T \times M}$ est une forme hyperbolique typique centrée à une position canonique ;
- $\mathbf{C}_k \in \mathbb{R}^{N_T \times M}$ est une carte de présence indiquant où et avec quelle intensité cette hyperbole apparaît ;
- $\circledast$ représente la convolution 2D ;
- $\mathbf{L}$ regroupe les contributions du sol, généralement modélisées comme de faible rang (réflexions stratifiées) ;
- $\mathbf{N}$ est le bruit résiduel, supposé aléatoire et additif.

Cette formulation offre un cadre très adapté pour appliquer des techniques d'apprentissage ou de factorisation, où l'on cherche à retrouver les activations $\mathbf{C}_k$ à partir des données mesurées $\mathbf{Y}$, tout en séparant les composantes parasites $\mathbf{L}$ et $\mathbf{N}$.

1.4 Traitements classiques liés au radargramme

Les données acquises par radar à pénétration de sol (GPR) sont souvent affectées par diverses formes de bruit, telles que le bruit thermique, les réflexions multiples ou les interférences électromagnétiques d'origine instrumentale et environnementale. Ces perturbations nuisent à la lisibilité du radargramme et compliquent la détection ou la classification d'objets enfouis.

Pour répondre à ces difficultés, plusieurs méthodes de traitement classiques ont été proposées dans la littérature. Elles visent typiquement à améliorer le rapport signal-bruit (RSB), à corriger les distorsions du signal, ou à extraire des caractéristiques discriminantes. Parmi les plus couramment utilisées, on peut citer l'amélioration du RSB, l'égalisation d'histogramme, la réduction de dimensionnalité par SVD, l'inversion parcimonieuse, ainsi que la détection statistique par GLRT (Generalized Likelihood Ratio Test).

Dans ce travail, ces approches sont présentées pour situer notre démarche dans le contexte des traitements classiques du GPR. Toutefois, elles ne sont pas utilisées dans notre pipeline, soit parce qu'elles reposent sur des hypothèses fortes (bruit gaussien, milieu homogène, etc.) difficilement vérifiées dans

nos données, soit parce qu'elles sont sensibles aux variations expérimentales (positionnement, élévation, météo) que nous cherchons justement à rendre robustes. Nous adoptons à la place une approche fondée sur l'apprentissage automatique, décrite dans les sections suivantes.

1.4.1 Amélioration du Rapport Signal-Bruit (RSB)

Une étape clé du traitement des données est l'amélioration du RSB (rapport signal/bruit). Cela inclut l'application de filtres (bandes passantes, filtrage de fréquence) ainsi que des méthodes de compensation pour atténuer les distorsions introduites par le milieu traversé. Plusieurs méthodes permettent d'améliorer le RSB, notamment l'égalisation d'histogramme et le moyennage spatial basé sur la décomposition en valeurs singulières (SVD) [25–27].

Égalisation d'histogramme

L'égalisation d'histogramme est une méthode de traitement d'image couramment utilisée pour améliorer le contraste d'un B-scan, représenté matriciellement par $\mathbf{Y}$. Elle redistribue les niveaux d'intensité en fonction de leur probabilité d'occurrence, afin d'étendre les niveaux de gris sur toute la plage dynamique disponible.

La méthode repose sur la fonction de distribution cumulative (Cumulative Distribution Function, CDF) de l'image, notée $C(r_k)$, calculée à partir de l'histogramme normalisé $p(r_k)$, qui donne la probabilité d'occurrence du niveau de gris r_k (avec $r_k \in \{0, \ldots, L-1\}$, et L le nombre de niveaux de gris (par exemple, 256).) La CDF est définie comme suit :

$$C(r_k) = \sum_{j=0}^{k} p(r_j)$$

La nouvelle valeur du pixel est ensuite calculée par la transformation :

$$s_k = (L-1) \cdot C(r_k)$$

où s_k est la nouvelle intensité attribuée au pixel d'intensité initiale r_k. Cette transformation permet une répartition plus uniforme des niveaux de gris, ce qui améliore la visibilité des structures enfouies.

En appliquant cette opération à tous les pixels de $\mathbf{Y}$, on obtient une version du B-scan dont les contrastes sont rehaussés, facilitant ainsi l'interprétation visuelle et la détection manuelle ou automatique des cibles [28].

Moyennage Spatial du B-scan par Décomposition en Valeurs Singulières (SVD)

La décomposition en valeurs singulières (SVD) est une méthode puissante pour analyser et filtrer les données des B-scans en isolant les composantes principales associées aux structures dominantes, comme les couches linéaires du sol, tout en supprimant le bruit et d'autres interférences. Soit $\mathbf{Y}$ la matrice représentant le B-scan, de dimensions $N_t \times N_x$, où N_t correspond au nombre de positions du radar et N_x au nombre de points dans le temps. La SVD permet de décomposer $\mathbf{Y}$ selon l'expression :

$$\mathbf{Y} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T,$$

où :

- $\mathbf{U} \in \mathbb{R}^{N_t \times N_t}$ contient les vecteurs singuliers à gauche ;
- $\mathbf{\Sigma} \in \mathbb{R}^{N_t \times N_x}$ est une matrice diagonale contenant les valeurs singulières σ_i, classées par ordre décroissant ;
- $\mathbf{V} \in \mathbb{R}^{N_x \times N_x}$ contient les vecteurs singuliers à droite.

Les réflexions dues aux couches linéaires du sol sont principalement capturées par les premières valeurs singulières $\sigma_1, \sigma_2, \ldots$, qui définissent les sous-espaces dominants, car plus forts en amplitude. Pour supprimer ces contributions, on calcule une version de rang réduit de l'image, $\mathbf{Y}_r$, en conservant uniquement les r premières composantes :

$$\mathbf{Y}_r = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T,$$

où r est typiquement choisi comme 1 ou 2, suffisant pour représenter les couches dominantes. La version débruitée $\mathbf{Y}_{\text{deb}}$ du B-scan est obtenue en retirant cette contribution à l'image d'origine :

$$\mathbf{Y}_{\text{deb}} = \mathbf{Y} - \mathbf{Y}_r.$$

Ainsi, $\mathbf{Y}_{\text{deb}}$ met en évidence les cibles souterraines d'intérêt tout en supprimant les artefacts linéaires associés aux couches et les interférences.

En résumé, l'application de la SVD au B-scan permet d'extraire et de supprimer efficacement les réponses dominantes des couches linéaires du sol, offrant une version débruitée où les cibles significatives sont plus visibles. Cette technique, décrite dans [27] et [29], est particulièrement utile pour les applications GPR où le traitement des couches de sol est essentiel.

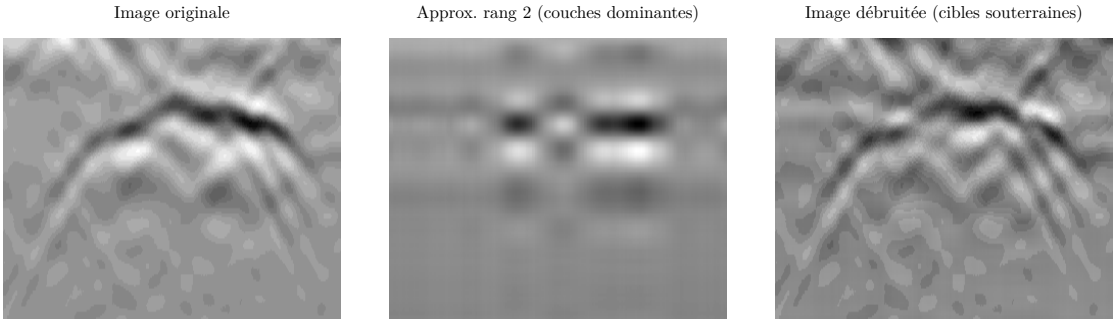


FIGURE 1.10 – Exemple B-scan SVD.

1.4.2 Méthodes d'inversion

Dans la sous-section 1.3.2, nous avons présenté un modèle du signal GPR comprenant un signal utile, des éléments de *clutter*[2], ainsi que du bruit. Le signal utile est modélisé par un codage parcimonieux des signatures des cibles dans un dictionnaire paramétré par les caractéristiques physiques de la scène, telles que la permittivité, la profondeur ou la géométrie des objets enfouis.

À partir de ce modèle, il est pertinent de formuler le problème inverse : estimer les coefficients $\{\mathbf{C}_k\}_{k=1,\ldots,K}$ et la matrice de *clutter* $\mathbf{L}$ afin de décomposer l'image GPR en ses différentes composantes. Cette estimation permet de détecter et caractériser les cibles enterrées tout en isolant la structure des couches souterraines, étape cruciale pour une interprétation fiable des données.

Plusieurs approches peuvent être envisagées pour résoudre ce problème d'inversion. Dans ce travail, nous privilégions une formulation par optimisation convexe, combinant fidélité aux données, parcimonie des coefficients, et modélisation du *clutter* par une contrainte de faible rang. Nous nous intéressons donc à l'estimation simultanée des contributions des cibles et des couches horizontales du sous-sol contenues dans la matrice de *clutter*.

2. Le *clutter* désigne les réflexions issues des couches horizontales du sous-sol, responsables d'échos multiples pouvant compliquer l'analyse.

Formulation du problème d'inversion

Pour inverser le modèle GPR, le problème d'optimisation suivant est proposé dans [4] :

$$\begin{aligned}(\{\mathbf{C}_k\}, \mathbf{L}) = \operatorname{argmin}_{\{\mathbf{C}_k\},\mathbf{L}} & \underbrace{\|\mathbf{Y} - \sum_{k=1}^{K} \mathbf{C}_k * \mathbf{H}_k - \mathbf{L}\|_2^2}_{\text{fidélité aux données}} \\ & + \underbrace{\lambda_1 \sum_{k=1}^{K} \|\mathbf{C}_k\|_1}_{\text{promotion de la parcimonie}} + \underbrace{\lambda_2 \|\mathbf{L}\|_*}_{\text{minimisation du rang}},\end{aligned} \tag{1.28}$$

où λ_1 et λ_2 sont des paramètres de régularisation. Cette fonction coût est conçue pour prendre en compte trois effets :

- un terme de fidélité aux données qui mesure la différence entre le modèle et le signal mesuré,
- un terme de promotion de la parcimonie via une norme L_1 sur chaque carte de coefficients $\mathbf{C}_k$,
- un terme de minimisation du rang via une norme nucléaire sur la matrice de clutter, favorisant les solutions à faible rang tout en conservant la convexité [30].

En complément de cette approche par optimisation convexe, des méthodes statistiques peuvent également être mobilisées pour tester la présence d'objets enfouis, en s'appuyant sur des hypothèses probabilistes sur le signal mesuré.

1.4.3 Méthodes statistiques [1]

Nous revenons ici à la représentation vectorielle du signal, introduite dans la sous-section 1.3.3. À partir des données radar $\boldsymbol{r}$, un vecteur $\boldsymbol{x}_{\epsilon',y,z}$ est construit à l'aide de la transformation $T_{\epsilon',y,z}$, qui extrait et aligne les valeurs du signal radar le long d'une hyperbole théorique correspondant à un objet enfoui de permittivité ϵ' situé en (y, z). Ce vecteur sert de base pour tester la présence d'une cible à cette position. Le problème peut alors être formulé comme un test d'hypothèse :

$$\begin{cases} H_0 : & \boldsymbol{x}_{\epsilon',y,z} = \boldsymbol{n}, \\ H_1 : & \boldsymbol{x}_{\epsilon',y,z} = a_{\epsilon',y,z}\boldsymbol{p} + \boldsymbol{n}, \end{cases} \tag{1.29}$$

où :

- $\boldsymbol{x}_{\epsilon',y,z}$ est le vecteur construit à partir des données radar brutes à l'aide de la transformation $T_{\epsilon',y,z}$, défini par :
$$T_{\epsilon',y,z}(\boldsymbol{r})[m] = \boldsymbol{r}_m\left(\tau_m(y, z, \epsilon')\right),$$
où :
  - r_m : trace temporelle (ou signal) enregistrée par l'antenne m,
  - $r_m(\tau_m)$: valeur de cette trace au temps correspondant à la propagation aller-retour entre l'antenne m et la position (y, z),
  - $\tau_m(y, z, \epsilon') = \dfrac{2}{c_0\sqrt{\epsilon'}}\sqrt{(x_m - y)^2 + z^2}$: temps de propagation associé.
- ϵ' est la permittivité diélectrique supposée du sol, influençant les temps de propagation dans le modèle ;
- $a_{\epsilon',y,z}$ est un coefficient d'amplitude à estimer ;
- $\boldsymbol{p}$ est le vecteur directeur représentant la forme théorique d'une cible enfouie (par exemple, un motif Ricker aligné sur une hyperbole) ;

- $\boldsymbol{n}$ est le bruit de fond, supposé gaussien, centré, avec une matrice de covariance Σ.

Pour résoudre ce problème de détection, on peut faire appel à une approche statistique fondée sur le test de rapport de vraisemblance généralisé (GLRT pour Generalized Likelihood Ratio Test), qui compare la vraisemblance des deux hypothèses :

$$\max_{\epsilon' \in \mathbb{R}^+} \frac{\left|\boldsymbol{p}^T \Sigma^{-1} \boldsymbol{x}\right|^2}{\left(\boldsymbol{p}^T \Sigma^{-1} \boldsymbol{p}\right)\left(\boldsymbol{x}^T \Sigma^{-1} \boldsymbol{x}\right)} \underset{H_1}{\overset{H_0}{\lessgtr}} \eta, \tag{1.30}$$

où :

- Σ est la matrice de covariance du bruit estimée à partir des données,
- $\boldsymbol{p}$ est défini à partir du motif d'onde de référence (ex. onde Ricker), centré sur une hyperbole théorique,
- η est un seuil de décision, fixé pour garantir une probabilité de fausse alarme P_{fa} donnée.

Le seuil η est déterminé en fonction du taux de fausses alarmes P_{fa}. Dans le cas particulier d'un bruit blanc et d'un *filtre adapté normalisé* (*Normalized Matched Filter*, **NMF**), ce seuil peut être exprimé analytiquement par :

$$\eta = 1 - P_{fa}^{\frac{1}{N-1}}, \tag{1.31}$$

où N est la taille du vecteur $\boldsymbol{x}$.

Estimation de la covariance du bruit En pratique, la matrice de covariance du bruit Σ est souvent inconnue et doit être estimée. Une estimation courante est donnée par :

$$\hat{\Sigma} = \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{x}_k \boldsymbol{x}_k^T, \tag{1.32}$$

où $\{\boldsymbol{x}_k\}_{k=1}^K$ sont des données secondaires collectées à des positions voisines de (y, z), mais suffisamment éloignées pour éviter toute contamination par un signal provenant de cibles potentielles. On effectuera une opération de régularisation sur la matrice pour éviter d'avoir des matrices singulières.

Résumé Le problème de détection est formalisé par l'équation (1.29) et le critère (1.30). Il repose sur la construction d'un vecteur directeur $\boldsymbol{p}$, l'estimation de la matrice de covariance $\hat{\Sigma}$, et l'évaluation du rapport de vraisemblance Λ par rapport à un seuil η. Ce détecteur permet de vérifier la présence d'un objet enfoui à une position donnée dans les données radar (pour plus de détails voir [1]).

Une fois les hyperboles détectées à l'aide des méthodes précédentes, il devient pertinent de les analyser plus finement afin de les classer selon leur nature. En effet, toutes les hyperboles ne correspondent pas nécessairement à des objets d'intérêt : certaines peuvent résulter de structures naturelles ou de réflexions parasites. L'étape de classification vise donc à distinguer automatiquement, parmi les hyperboles détectées, celles correspondant à des cibles pertinentes (objets enfouis) de celles issues du *clutter* ou d'autres artefacts.

Pour cela, nous faisons appel à des méthodes d'apprentissage supervisé, notamment des approches récentes basées sur le *deep learning*, capables de tirer parti des caractéristiques géométriques et spectrales des signatures hyperboliques pour en extraire des descripteurs discriminants.

1.5 Classification des Hyperboles Radar

La classification des hyperboles détectées sur les radargrammes est une étape essentielle dans l'analyse des données radar, permettant d'identifier et de différencier divers éléments souterrains. Ces hyperboles, préalablement détectées et localisées à partir des données radar traitées, nécessitent une analyse approfondie afin d'en déterminer la nature.

L'approche retenue repose sur une méthode supervisée, qui peut être mise en œuvre selon deux stratégies complémentaires. Cette méthode est particulièrement adaptée dans le contexte de la présente étude, où des données partiellement labellisées sont disponibles, et où l'objectif est d'obtenir une classification explicite des cibles identifiées dans les radargrammes.

1.5.1 Classification supervisée indépendante

La première stratégie consiste en une classification supervisée indépendante, où chaque hyperbole détectée est analysée séparément pour être assignée à une catégorie spécifique, comme des objets, des réseaux ou des discontinuités, en s'appuyant sur des données préexistantes et étiquetées. Cette approche permet de dissocier clairement les étapes de détection et de classification, facilitant l'analyse des performances de chaque composant du pipeline.

1.5.2 Classification intégrée à la détection

La seconde stratégie vise une classification intégrée à la détection, permettant de réaliser ces deux tâches simultanément. Cette approche améliore la précision globale tout en réduisant le nombre d'étapes intermédiaires dans le processus d'analyse. Dans les deux cas, la forme spécifique des hyperboles constitue un critère déterminant, servant de base pour extraire des imagettes, sous-segments des radargrammes centrés sur chaque hyperbole, qui fournissent des exemples visuels essentiels à l'entraînement des modèles de classification. Les modèles doivent être performants tout en restant robustes au faible nombre de données annotées, aux conditions expérimentales (différents sols, élévations, ...) et à une possible mauvaise labellisation.

Pour effectuer ces traitements, il est nécessaire de disposer d'un jeu de données annotées de bonne qualité. Nous proposons de construire cette base de données d'entraînement à partir des données obtenues par une campagne réalisée par la société Géolithe. Cette campagne est décrite dans la section suivante. On présente ensuite le jeu de données qui sera utilisé dans la suite de la thèse ainsi que les prétraitements qui ont été développés.

1.6 Dataset Géolithe

Nous présentons donc dans cette section la campagne de mesures effectuées par la société géolithe.

1.6.1 Présentation du site

Localisation de la zone d'étude

La campagne d'acquisition s'est déroulée dans la carrière GRANULATS VICAT de Barraux (Isère, 38), voir la Figure 1.11. Une large zone de la carrière a été réservée pour nos manipulations du 14 septembre 2020 au 16 février 2021. Les mesures ont été effectuées au niveau de deux tranchées creusées côte à côte, puis remblayées après la pose des différents objets.

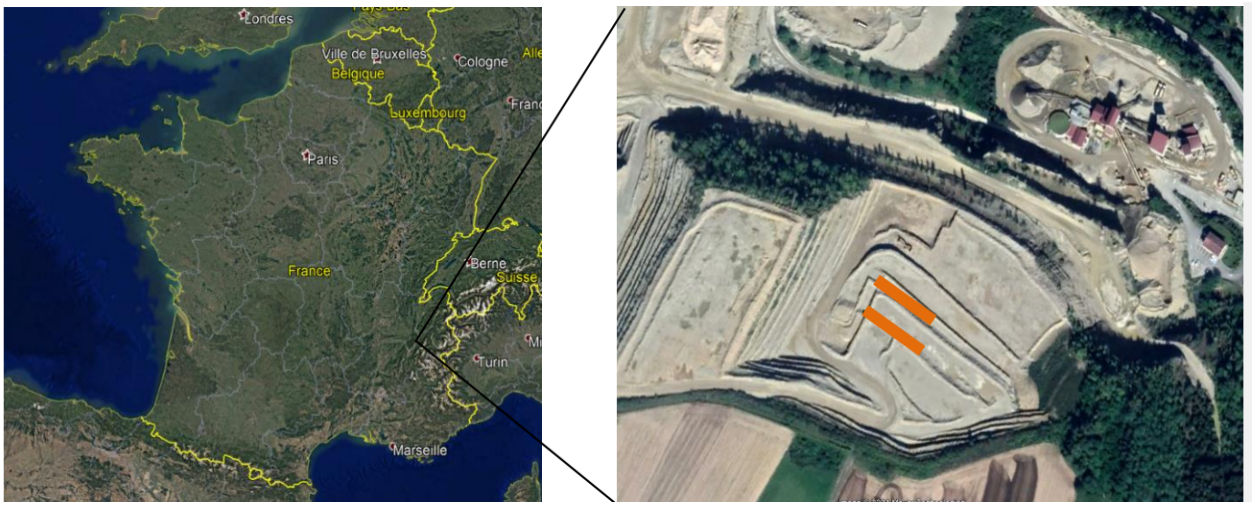


FIGURE 1.11 – Localisation de la zone d'étude : carrière de Barraux (Isère)

Cette campagne, préparée en amont de cette thèse, a nécessité une organisation logistique importante. Trois personnes ont travaillé en moyenne six jours pour installer les dispositifs de mesure, puis deux à trois opérateurs ont assuré les acquisitions pendant vingt jours. Une journée supplémentaire a été consacrée à la remise en état du site.

Les mesures GPR ont été réalisées par une équipe pluridisciplinaire composée de Nickolas Stelzenmuller (docteur en hydrologie et encadrant de cette thèse), Alexandra Royer (docteure en géophysique), Aline Esparel (ingénieure géophysicienne) et Nicolas Fleury (géologue). Leur expertise a permis d'assurer la qualité des données, qui constituent la base empirique du travail de classification présenté dans ce manuscrit.

Plan des tranchées

Deux tranchées ont été creusées côte à côte, puis remblayées l'une avec de la grave, l'autre avec du sable sec. Au fur et à mesure du remblaiement, des objets cibles ont été disposés de manière identique dans chaque tranchée. Les tranchées mesurent environ 50 m de long, 6,4 à 13 m de large et 2,2 à 4 m de profondeur (on montre les plans des tranchées sur les Figures 1.12 et 1.13). Elles sont orientées NW-SE et ont été conçues de manière à :

- Disposer les cibles centrées sur la tranchée, dans le sens de la longueur, à l'exception du téléphone portable (placé sur un côté de la tranchée) ;
- Garder environ un mètre de matériau de remplissage sous chaque cible, à l'exception du plus gros abri, posé sur la base de la tranchée ;
- Permettre de distinguer les cibles avec une antenne radar élevée à 2 m, en supposant un cône de radiation de 45°.

Les plans d'une des tranchées (en carte et en coupe) sont visibles sur les Figures 1.12 et 1.13 ci-dessous.

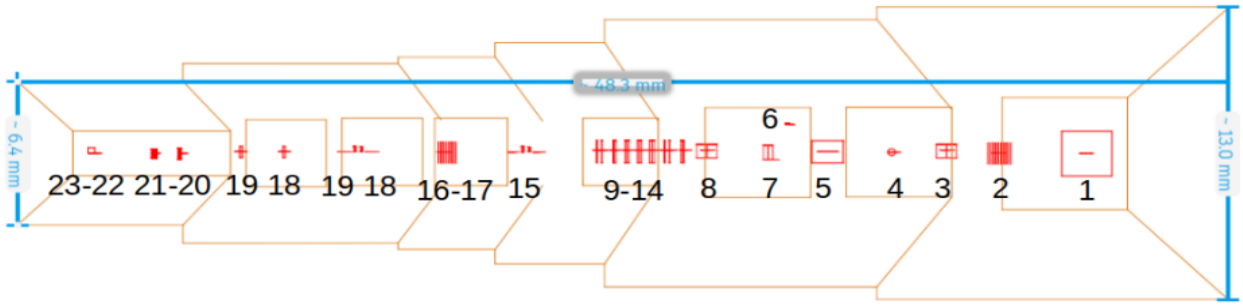


FIGURE 1.12 – Plan d'une tranchée en carte. Les bords de la tranchée sont représentés en orange, et les cibles enfouies en rouge.

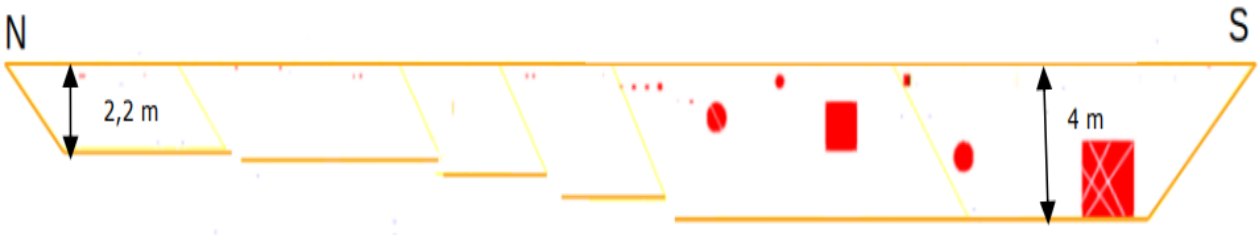


FIGURE 1.13 – Vue en coupe d'une tranchée. Les bords de la tranchée sont représentés en jaune, et les cibles enfouies en rouge.

Objets cibles

Le tableau 1.3 récapitule les objets enfouis et leur positionnement dans les tranchées de grave humide et de sable sec.

TABLE 1.3 – Récapitulatif des objets cibles enfouis dans les tranchées de grave humide et de sable sec et leur distance par rapport au départ des lignes d'acquisition

No	**Objet**	**Distance (m)**	**Profondeur du toit (m)**	**Distance (m)**	**Profondeur du toit (m)**
1	Grand abri	5,3	1,8	5	1,6
2	Fusils factices	7,9	0,9	8,7	1,2
3	Buse	10,1	1,7	12,1	1,5
4	Bidon vide	11,9	0,3	15,2	0,3
5	Petit abri	14,8	0,8	17,4	1
6	Téléphone	17,9	0,1	19,5	0,2
7	Fût	16,5	0,4	20,7	0,1
8	Buse	20,7	0,5	23	0,4
9	PEHD (Polyéthylène haute densité) D11cm	22	0,6	24,4	0,7
10	PEHD D3,6cm	22,6	0,9	25,2	0,8
11	PVC (Polychlorure de vinyle) D10 vide	23,2	0,3	25,9	0,2
12	PVC D10 cm plein	24	0,3	26,7	0,2
13	PVC D16 cm vide	24,9	0,2	27,6	0,2
14	PEHD D11 cm	25,7	0,3	28,6	0,4

No	Objet	Distance (m)	Profondeur du toit (m)	Distance (m)	Profondeur du toit (m)
15	Fusils factices	27,3	0,2	30,1	0,3
16	Mine factice en résine	29,3	0,3	31,9	0,4
17	Planche en bois	31,5	0,1	33,8	0,3
18	Gros obus factices	33,4	0,1	36,2	0,2
19	Petits obus factices	35,9	0,2	38,9	0,2
20	Mines métalliques factices	38,6	0,3	40,1	0,4
21	Mines factices en pavés de résine et cire	41,3	0,3	42	0,3
22	PEHD D3,6 cm vide	43,8	0,4	43,5	0,2
23	PEHD D3,6 cm plein	45,9	0,4	44,6	0,5

Conditions sur le terrain

Les journées d'acquisition se sont déroulées par temps couvert à ensoleillé et ont été annulées en cas de prévision d'intempéries.

Les acquisitions prévues ont été effectuées dans la grave et le sable, en conditions humides et sèches. La tranchée de grave était humide au début de la campagne d'acquisition mi-octobre, comme en témoigne la présence de flaques sur et à proximité de la tranchée, ainsi que de la boue au début de celle-ci (cf Figure 1.14, photo ci-dessous, prise le 28 octobre 2020). La tranchée s'est naturellement asséchée durant le mois de novembre, permettant ainsi des acquisitions dans des conditions sèches.

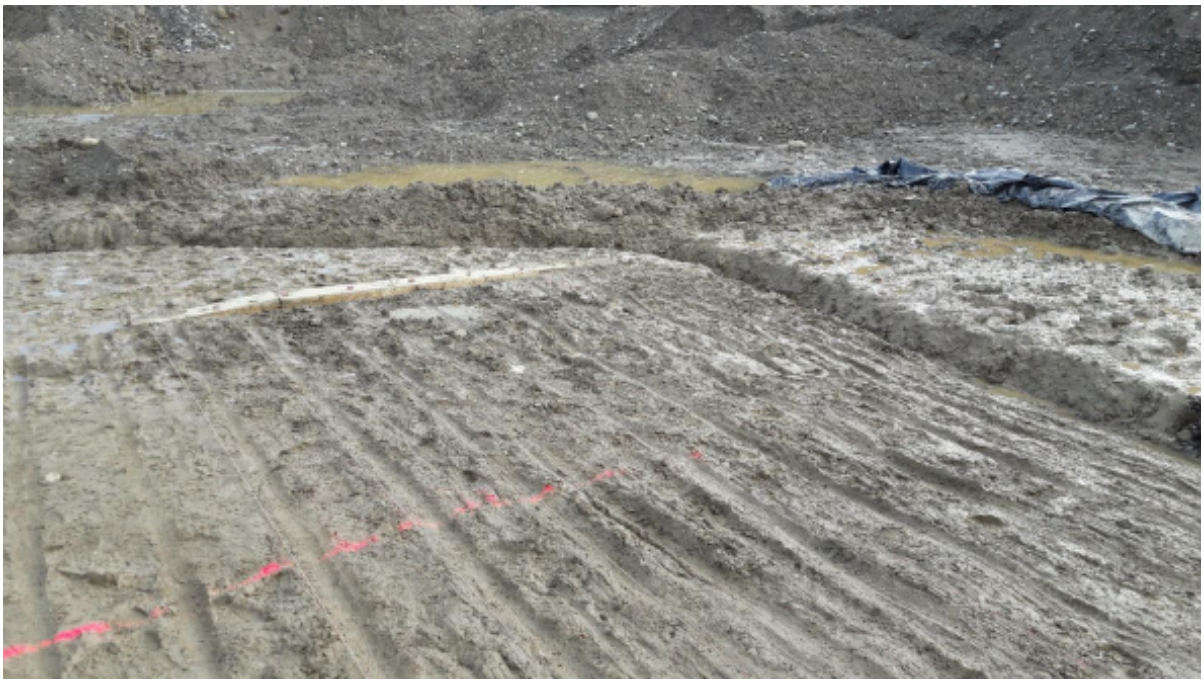


FIGURE 1.14 – Début de la tranchée de grave argileuse le 28/10/20, en conditions humides

Le temps étant globalement sec tout au long du mois de novembre, la tranchée de sable sec a été arrosée pour atteindre des conditions humides, cf Figure 1.15. Entre 30 et 40 m^3 d'eau ont été utilisés au moyen d'une citerne et de tuyaux, dont un tuyau de 50 m percé pour permettre un arrosage tout au long de la tranchée.

FIGURE 1.15 – Système d'arrosage de la tranchée de sable sec : tuyau percé et citerne de 17 m^3

1.6.2 Antennes GSSI utilisées

Matériel d'acquisition GSSI

Deux géoradars de la marque américaine GSSI (pour Geophysical Survey Systems, Inc.) ont été utilisés pour cette mission : les géoradars 200 MHz HyperStacking et 350 MHz UtilityScan Pro. Ces antennes radar à technologie HyperStacking permettent d'acquérir plusieurs échantillons par impulsion (pulse), au lieu d'un unique échantillon par impulsion pour des antennes classiques, ce qui améliore la profondeur d'investigation en augmentant le rapport signal sur bruit.

FIGURE 1.16 – Dispositif d'acquisition des radargrammes au sol : antenne 200 MHz (gauche) et 350 MHz (droite) avec roue codeuse et GPS (Global Positioning System), connectés au SIR 4000

Ces antennes étaient couplées à l'unité de contrôle SIR 4000, et le positionnement des profils a été assuré par un GPS SP90 RTK (Spectra Precision 90 Real-Time Kinematic) ainsi qu'une roue codeuse.

Un chariot en fibre de verre et en bois a été conçu pour surélever ces antennes à une hauteur constante tout au long des profils, minimisant ainsi les interférences électromagnétiques grâce à l'absence d'éléments métalliques.

Protocole de mesure GSSI

L'antenne de 200 MHz, adaptée à l'imagerie des objets de grande dimension et à grande profondeur, a été utilisée uniquement dans la première partie de la tranchée. Les profils couvraient une longueur de 25 m et une largeur de 3,5 m.

Avec l'antenne de 350 MHz, tous les objets de la tranchée ont été imagés sur une longueur de 47 m et une largeur de 3,2 m. Cependant, certains profils ont dû être raccourcis à cause d'inondations. Le schéma des profils est illustré sur la Figure 1.17.

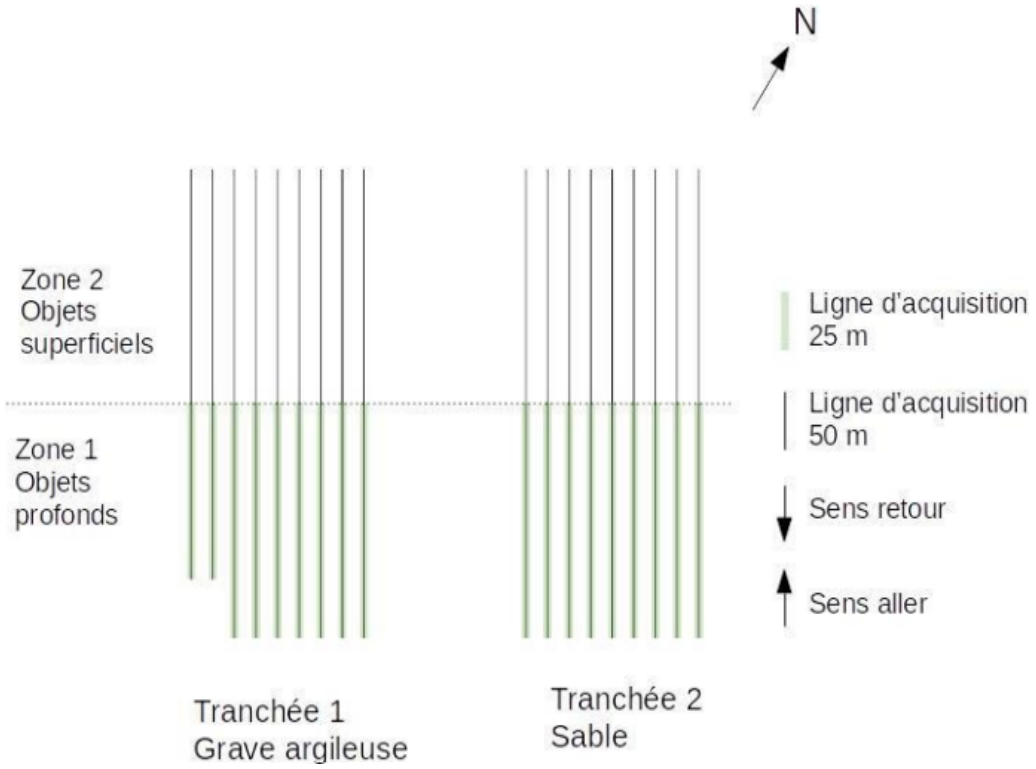


FIGURE 1.17 – Schéma des profils réalisés avec les antennes 200 MHz et 350 MHz (vu de dessus)

Traitement des données GSSI

Les radargrammes acquis avec les antennes de 350 MHz et de 200 MHz HS de GSSI ont été traités avec le logiciel open source GPRPY [31]. Une chaîne de traitement identique a été appliquée aux radargrammes présentés dans ce rapport. Un exemple d'application de ces traitements est montré à titre pédagogique sur la Figure 1.18.

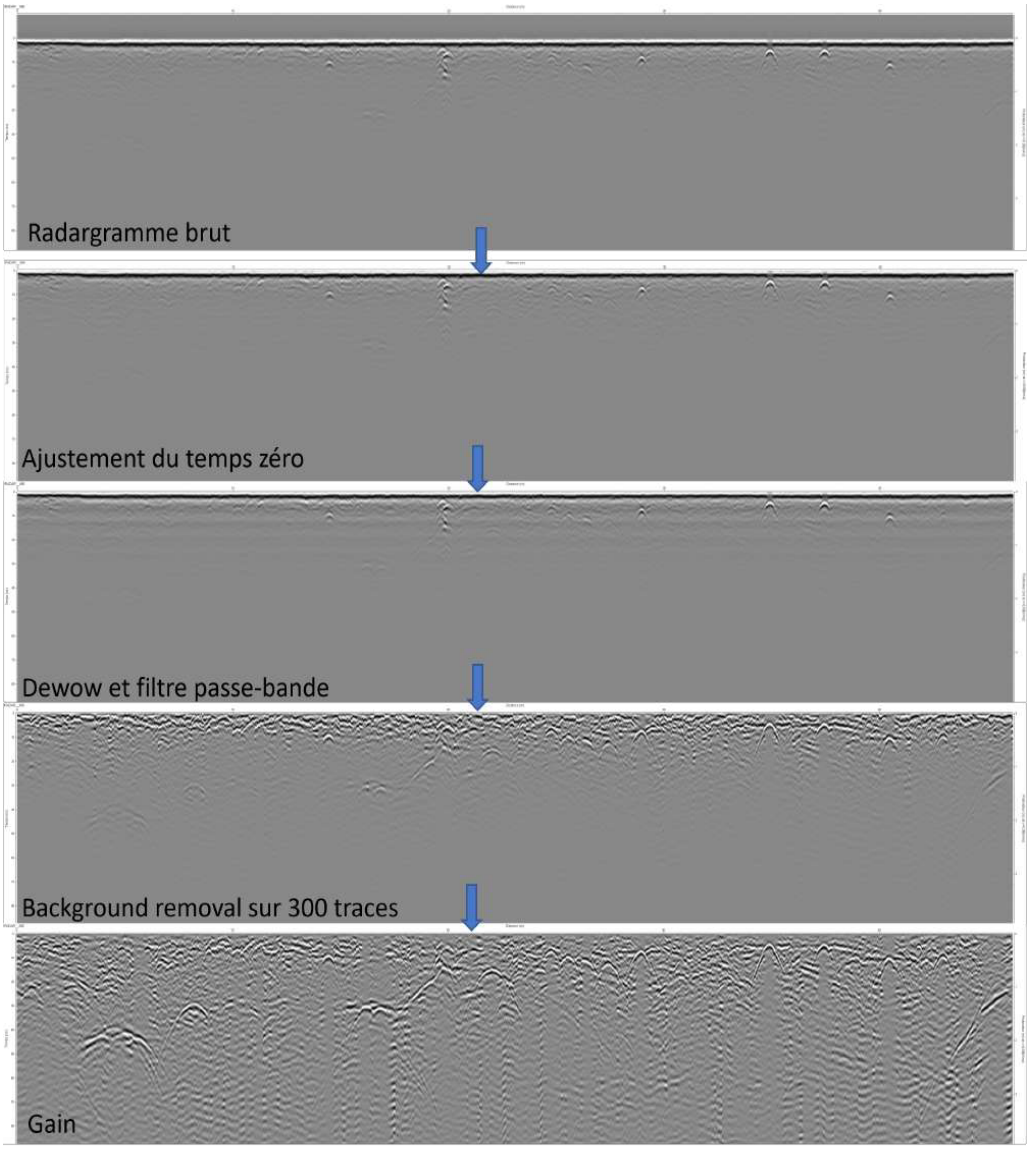


FIGURE 1.18 – Exemple des différentes étapes de la chaîne de traitement sur un radargramme acquis dans le sable sec, avec l'antenne plaquée au sol.

La chaîne de traitement utilisée avec les antennes de 200 et 350 MHz est la suivante :

- Correction du temps zéro (temps correspondant à la pénétration des ondes radar dans le sol) ;
- « Dewow » avec une fenêtre de 20 traces, une opération de filtrage temporel qui consiste à supprimer la composante continue ou basse fréquence (offset ou dérive lente) présente dans chaque trace radar. Cette étape permet de recentrer le signal autour de l'amplitude zéro, facilitant ainsi l'analyse des réflexions radar utiles en éliminant les variations lentes dues au système ou au milieu ;
- Filtre passe-bande de Butterworth avec les fréquences de coupure autour de la fréquence centrale f_c ($1/2\ f_c$ et $2\ f_c$) pour éliminer le bruit haute fréquence (interférences EM) et le bruit basse fréquence (lignes haute tension, etc.) ;
- « Background removal » sur une fenêtre glissante de 300 traces, réalisée en calculant la moyenne ou la médiane des traces sur cette fenêtre, puis en soustrayant cette estimation du bruit de fond

continu (incluant le bruit stationnaire, le ringing antenne-sol, etc.) de chaque trace individuelle. Cette méthode permet d'atténuer efficacement les composantes statiques et répétitives présentes dans les données radar. ;

- « Gain power » 1,5 pour compenser la perte d'intensité du signal avec la profondeur.

Pour les profils en surélévation, les mêmes traitements ont été effectués, à l'exception du retrait de fond (background removal), qui est effectué en premier afin de mieux choisir le temps zéro.

Pour la conversion temps/profondeur, les vitesses des ondes électromagnétiques ont été choisies sur la base de la profondeur du grand abri. Les vitesses sont les suivantes :

- Grave sèche : $v = 0,075$ m/ns, $\varepsilon_r = 16$;
- Grave humide : $v = 0,075$ m/ns, $\varepsilon_r = 16$;
- Sable humide : $v = 0,087$ m/ns, $\varepsilon_r = 11,9$;
- Sable parasite : $v = 0,095$ m/ns, $\varepsilon_r = 10$;
- Sable sec : $v = 0,09$ m/ns, $\varepsilon_r = 11$.

Élévations des antennes GSSI

Un chariot en fibre de verre et en bois, visible sur la Figure 1.19, a été conçu pour surélever les antennes de 200 et 350 MHz à une hauteur constante tout au long d'un profil . Le choix des matériaux de fabrication, en limitant l'usage d'éléments métalliques, permet de minimiser les interférences électromagnétiques. Le cadre et les roues du chariot sont positionnés à une distance suffisante des antennes afin de réduire ces interférences.

FIGURE 1.19 – Dispositif d'acquisition des radargrammes en élévation

Les hauteurs d'élévation retenues furent de 25 cm, 50 cm, 75 cm, 100 cm et 150 cm. Ces choix ont été déterminés en fonction de plusieurs critères :

- La nécessité d'une hauteur minimale de 25 cm pour une éventuelle acquisition par drone dans des conditions idéales ;

- Le temps alloué à la campagne expérimentale ainsi que la durée nécessaire pour chaque acquisition ;

- La limite de 100 cm d'élévation, identifiée par les experts de Géolithe, a été établie afin de garantir des résultats exploitables avec ce type de géoradar.

Dans la suite de ce travail, notre objectif est d'automatiser la reconnaissance des structures souterraines à partir des radargrammes GPR. Cependant, la complexité physique de la propagation des ondes, la diversité des configurations du sol, ainsi que la présence de nombreux artefacts dans les données rendent extrêmement difficile une modélisation explicite des cibles. Cette variabilité limite la généralisation des approches fondées uniquement sur des modèles physiques ou des règles déterministes. Pour pallier ces limitations, l'approche retenue dans cette thèse repose sur l'apprentissage supervisé à partir de données réelles annotées. Avant de pouvoir entraîner de tels modèles, il est essentiel de bien comprendre la signature typique des objets recherchés dans les données radar.

1.6.3 Caractérisation de quelques objets recherchés

Avant d'envisager des approches de classification automatique, notamment par apprentissage supervisé, il est important de comprendre comment se manifestent les cibles recherchées dans les données radar. Cette étape permet d'identifier les caractéristiques discriminantes exploitables par les algorithmes d'apprentissage et de mieux appréhender les limites des méthodes classiques fondées sur des règles heuristiques ou une inspection manuelle. En effet, la diversité des formes de réponses radar, leur dépendance aux conditions expérimentales (fréquence d'antenne, élévation, nature du sol) et la présence de signaux parasites rendent difficile la mise en place de règles fixes et robustes.

Dans cette optique, nous caractérisons ici la signature radar de plusieurs objets types susceptibles d'être rencontrés dans nos scénarios. Nous examinons notamment leur géométrie (forme de la réflexion, présence de multiples, etc.), ainsi que l'intensité et la polarité des signaux associés.

Le logiciel *gprMax* [32] a été utilisé pour simuler des radargrammes correspondant à différents modèles de sous-sol. Les résultats simulés pour chaque objet, obtenus avec l'antenne 350 MHz dans la tranchée de sable sec, sont confrontés aux radargrammes réels afin d'illustrer visuellement les similarités et les écarts entre les cas idéalisés et les mesures expérimentales. Cette comparaison constitue un support pédagogique pour mieux comprendre les défis de la classification automatique à partir de ces signatures.

Sur la Figure 1.20, seuls les objets situés sur la cinquième ligne d'acquisition sont numérotés pour faciliter la comparaison entre radargramme simulé et radargramme réel. Les résultats seront détaillés objet par objet dans la suite de cette partie. Pour chaque objet, des extraits de radargrammes seront comparés dans le sable humide (meilleure détection et résolution), pour chaque antenne (dans la mesure du possible). Regardons quelques objets séparement.

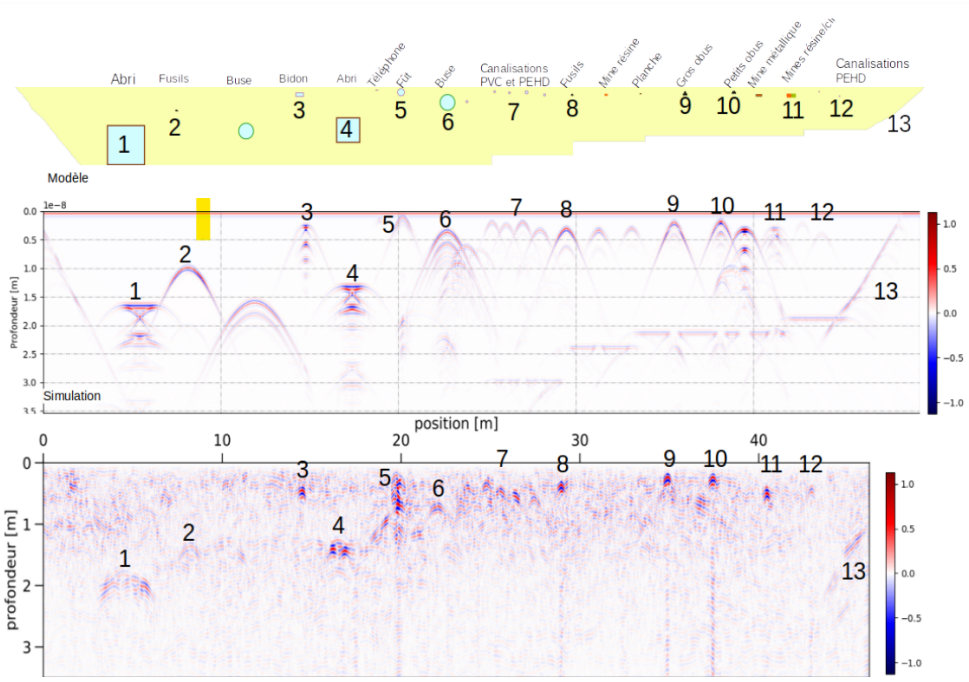


FIGURE 1.20 – Modèle de la tranchée de sable sec (en haut), radargramme simulé avec gprMax (au milieu) et réel (en bas). Pour plus de visibilité, les radargrammes sont ici affichés avec un ratio de 4. Un gain power 2 a été appliqué à ces deux radargrammes (plus la chaîne de traitement évoquée plus haut pour le radargramme réel).

Abris en bois

Les abris en bois sont cubiques, de 2 m (abri 1) et de 1,25 m de côté (abri en Figure 1.21 ci-dessous).

FIGURE 1.21 – Photographie d'un abri en bois en cours de transport

Sur le radargramme simulé, visible sur la Figure 1.22 ci dessous,

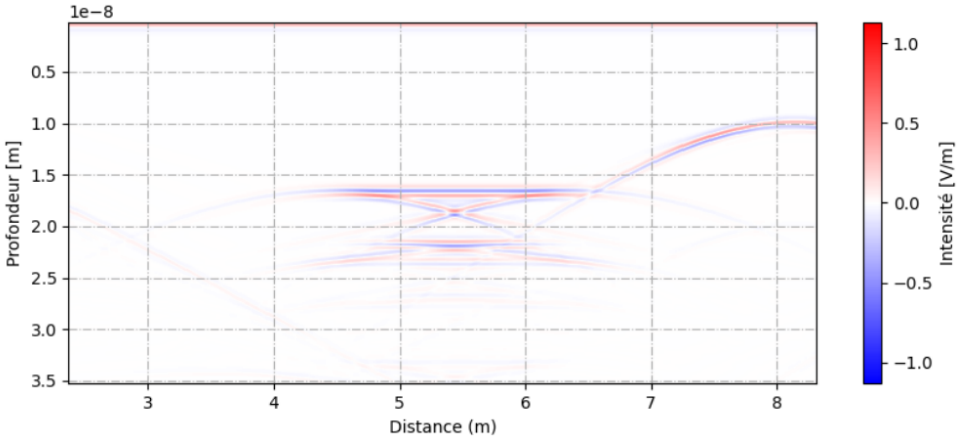


FIGURE 1.22 – Grand abri modélisé avec gprMax dans la tranchée de sable humide, avec une antenne de 350 MHz

la signature de l'abri est composée :

- Une large hyperbole correspondant au toit de l'abri (surface plane). La largeur de la partie aplatie, de forte intensité, correspond à la largeur de l'abri (2 m). Cette réflexion correspond plus précisément à l'interface sol–air (et non sol–bois), en raison du fort contraste de permittivité diélectrique entre l'air et le sol et de la faible épaisseur du bois. La polarité est normale (+-+ ici) : il n'y a pas de changement de polarité au contact sol–air (diminution de la permittivité diélectrique).

- Deux hyperboles adjacentes situées sous la première réflexion. Elles correspondent aux réflexions sur les coins de l'abri et présentent une polarité inverse.

- Une large hyperbole située sous les hyperboles précédentes, correspondant à la base de l'abri. Cette réflexion apparaît à une profondeur moindre que la réalité, du fait de la propagation plus rapide des ondes électromagnétiques dans l'air contenu dans l'abri. La réflexion est inverse (contact air–sol), et la largeur de la partie aplatie correspond, comme précédemment, à la largeur de la base de l'abri.

- Plusieurs réflexions supplémentaires visibles en augmentant le gain : les mêmes réflexions se répètent avec une intensité moindre.

Sur les radargrammes « réels », le grand abri (cf Figure 1.23) apparaît le plus souvent sous la forme d'une large hyperbole, dont la partie sommitale subhorizontale atteint environ 2 m. On distingue parfois les deux hyperboles adjacentes situées en dessous de celle-ci.

Le deuxième abri (radargrammes en Figure 1.24) est mieux dessiné, avec la réflexion plane (toit de l'abri) reposant sous les deux hyperboles adjacentes (côtés) et sur une seconde réflexion plane (base de l'abri). La partie plane de la première réflexion et l'écart entre les sommets des hyperboles adjacentes donnent la largeur de l'abri : 1,25 m.

Comme attendu, le toit des abris est le plus souvent observé avec une polarité normale (+-+ ici), les branches des hyperboles apparaissent généralement avec une polarité inverse (-+- ici), et la base présente également une polarité inverse (-+- ici).

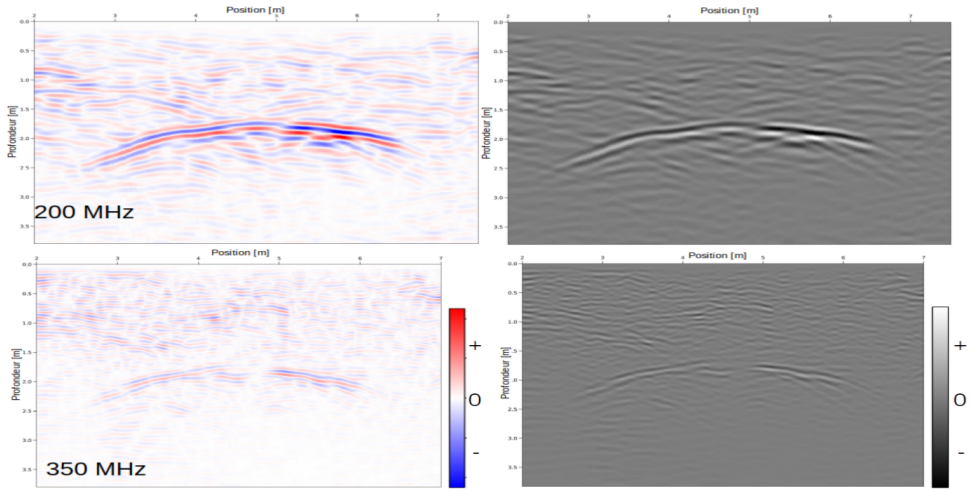


FIGURE 1.23 – Grand abri enfoui à 1.6 m de profondeur dans le sable humide pour deux échelles de couleurs différentes, à gauche le radargramme simulé et à droite le radargramme mesuré

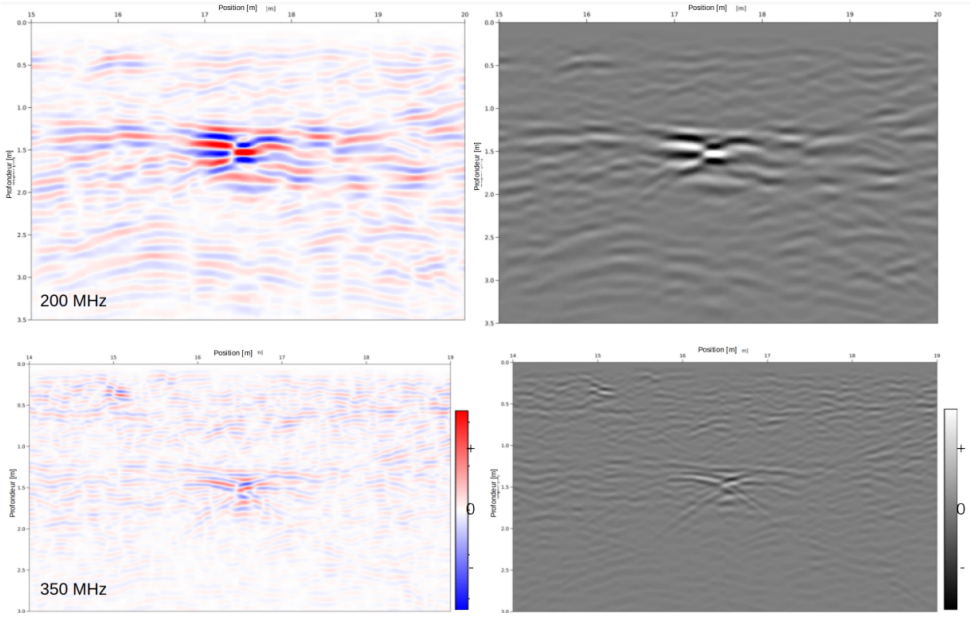


FIGURE 1.24 – Petit abri enfoui à 1 m de profondeur dans le sable humide, à gauche le radargramme simulé et à droite le radargramme mesuré

Fusils métalliques

Des barres métalliques de 80 cm de long ont simulé des fusils (Figure 1.25). Elles étaient disposées dans les tranchées par paquets de cinq et enveloppées de tissu.

FIGURE 1.25 – Photographie des barres métalliques utilisées pour simuler des fusils

Sur la simulation, les fusils se repèrent par une large hyperbole de forte intensité et de polarité inverse (-+- ici). L'hyperbole s'élargit avec la profondeur d'enfouissement (Figure 1.26 vs Figure 1.28).

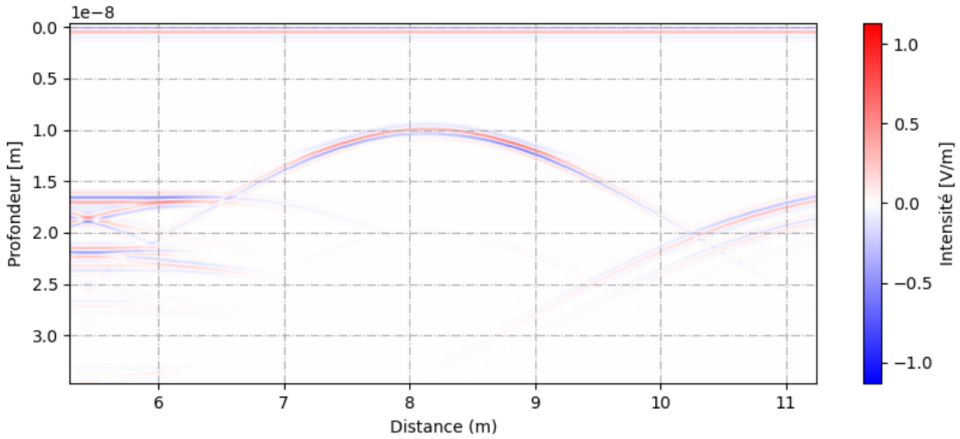


FIGURE 1.26 – Amas de fusils enfoui à 1 m modélisé avec gprMax dans la tranchée de sable humide, avec une antenne de 350 MHz

On note la présence d'un multiple sur la Figure 1.27. Ces caractéristiques s'expliquent par la nature métallique de ces simulâcres.

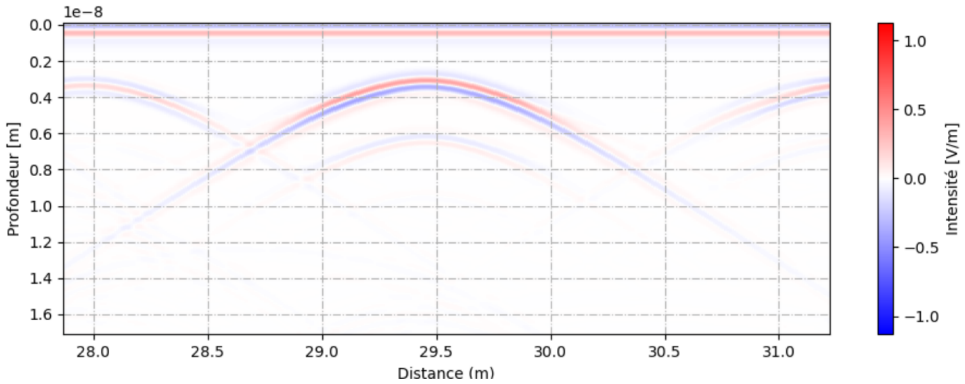


FIGURE 1.27 – Amas de fusils enfoui à 30 cm modélisé avec gprMax dans la tranchée de sable humide, avec une antenne de 350 MHz

Les paquets de fusils se présentent sur les radargrammes (Figure 1.28, Figure 1.27) sous la forme d'hyperboles parfaitement dessinées, de forte intensité et de polarité inverse (-+- ; sauf quelques rares

exceptions, qui peuvent être dues à une erreur de lecture). Comme attendu, la largeur de l'hyperbole croît avec la profondeur d'enfouissement des fusils.

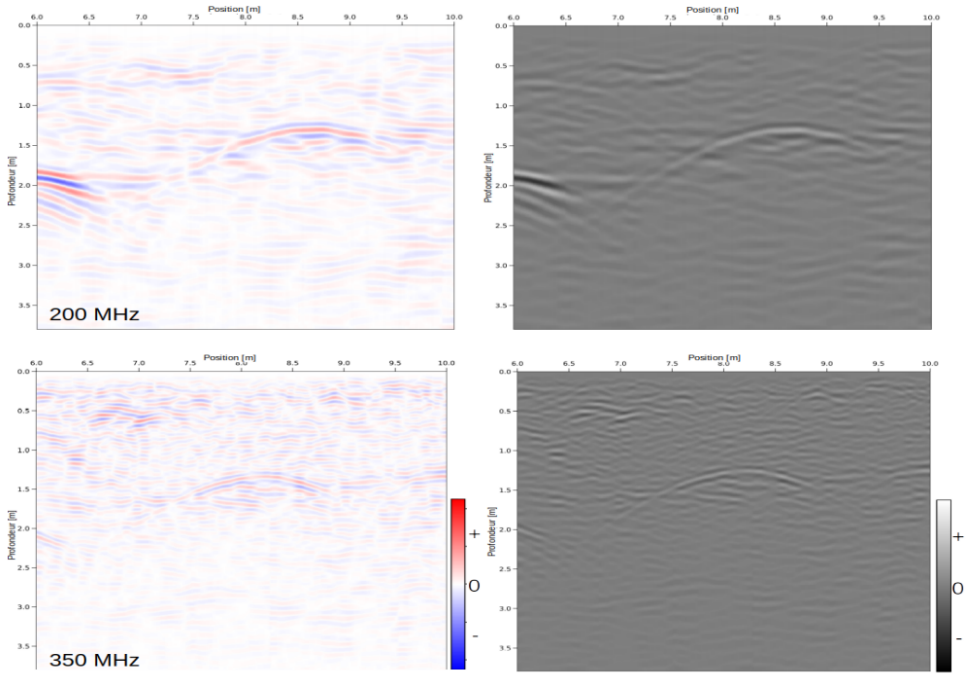


FIGURE 1.28 – Fusils enfouis à 1,2 m de profondeur dans le sable humide, à gauche le radargramme simulé et à droite le radargramme mesuré

De petites réflexions régulières sont parfois visibles sous la réflexion principale correspondant au tas de fusils (cas de l'antenne de 350 MHz dans la Figure 1.27). Il s'agit d'un courant induit effectuant un mouvement de va-et-vient sur les objets métalliques.

Bilan des caractéristiques des objets

Le tableau 1.4 récapitule les caractéristiques des objets tels qu'observés sur les radargammes issus de cette campagne expérimentale :

TABLE 1.4 – Caractéristiques des objets cibles observés lors de la campagne expérimentale

Objet	**Géométrie**	**Polarité**	**Intensité**
Abri en bois, vide	Réflexion plane + 2 hyperboles adjacentes + éventuelle réflexion plane en dessous	Normale au sommet, inverse pour les coins, inverse pour la base	Forte
Fusils (barres métalliques)	Hyperbole bien dessinée + éventuel multiples / ringing	Inverse	Forte
Obus (tubes métalliques vides)	Hyperbole bien dessinée + éventuels multiples / ringing	Inverse	Forte
Fût métallique vide	Hyperbole bien dessinée (si non endommagé) + éventuels multiples / ringing	Inverse	Forte
Mine cylindrique en résine	Hyperbole	Lecture difficile, théoriquement normale	Faible à moyenne

Objet	Géométrie	Polarité	Intensité
Mine cylindrique métallique remplie de cire ou de résine	Hyperbole + multiples / ringing	Inverse	Forte
Mines pavés en cire et résine	Hyperbole	Lecture difficile, théoriquement normale	Faible à moyenne
Canalisation PEHD ou PVC vide	Hyperbole bien dessinée	Normale	Faible
Canalisation PEHD ou PVC remplie	Hyperbole bien dessinée	Variable, théoriquement inverse	Moyenne à forte
Buses en béton	Hyperbole large bien dessinée	Normale	Faible à moyenne
Bidon d'essence vide	Hyperbole	Lecture difficile, théoriquement normale	Faible à moyenne
Téléphone portable éteint	Hyperbole resserrée + multiples	Inverse	Forte
Planche en bois	Hyperbole légèrement aplatie	Lecture difficile, théoriquement normale	Faible
Planche en bois enrobée de caoutchouc	Hyperbole légèrement aplatie	Lecture difficile, théoriquement normale normale	Faible à moyenne

Les objets peuvent être regroupés en quatre grandes catégories à partir de leur caractéristique sur les radargammes :

- Les objets métalliques : réflexions de forte intensité, polarité inverse, présence fréquente de multiples et réverbérations.
- Les canalisations : réflexion parfaitement hyperbolique, intensité moyenne, polarité difficilement lisible et dépendant de la nature de la canalisation.
- Les abris : hyperbole dont le toit est aplati, base et bord parfois visibles, forte intensité et polarité normale.
- Les petits objets dont la permittivité diélectrique relative est proche de celle du sol (plastique ou bois notamment) : petite hyperbole de faible intensité, polarité difficilement lisible.

L'analyse du tableau 1.4 met en évidence la grande diversité des signatures radar selon les types d'objets. On observe des variations notables en termes de **forme** (géométrie), de **polarité** et d'**intensité**, même au sein d'une même catégorie (par exemple, les mines en résine versus celles en métal). Ces réponses dépendent non seulement des **propriétés physiques** des objets (forme, matériau, remplissage), mais aussi de nombreux **facteurs contextuels** tels que la fréquence de l'antenne, l'élévation, la composition du sol, ou encore la profondeur d'enfouissement.

Dans ce contexte, la mise en place d'un **modèle mathématique explicite** capable de décrire exhaustivement l'ensemble de ces cas apparaît comme extrêmement difficile, voire irréaliste. Chaque catégorie nécessiterait un traitement spécifique, et les nombreuses **exceptions** ou cas limites (par exemple, objets dont la polarité est « difficilement lisible » ou l'intensité « variable ») compromettent la **généralisation** d'un tel modèle.

C'est pourquoi l'approche adoptée dans cette thèse repose sur des méthodes d'**apprentissage supervisé**, qui permettent de tirer parti de l'information contenue dans les données elles-mêmes. Le machine learning (ML) offre une solution pragmatique et puissante pour capturer la complexité inhérente à ces réponses radar, sans nécessiter une modélisation analytique complète. Les caractéristiques décrites dans ce tableau constituent ainsi la base des **éléments discriminants** que les modèles d'apprentissage chercheront à exploiter.

Avant d'aborder les choix méthodologiques pour la classification automatique, il est essentiel de présenter les conditions d'acquisition des données et leur organisation, car elles influencent directement la qualité et la diversité de l'information exploitable.

1.6.4 Organisation des campagnes terrain et structuration des données

Deux à trois personnes ont été mobilisées sur le site pour les acquisitions, avec la présence continue d'une personne tout au long de la campagne. Cette continuité a permis de maintenir un **protocole rigoureux** lors des différentes sessions d'acquisition.

Après quelques jours d'ajustement, il a été décidé de réaliser les mesures pour toutes les **élévations**, sur un type de sol donné et avec une **antenne spécifique**, en une seule journée (par exemple : acquisition dans la tranchée de sable humide avec l'antenne de 350 MHz, aux élévations de 0 à 150 cm, sur les neuf lignes d'acquisition en aller-retour).

Des **fiches récapitulatives** (paramètres d'acquisition, élévation, type de sol, schéma des profils, numéro de fichier) ont été complétées pour chaque type d'acquisition. Les données ont été triées et vérifiées dès que possible après chaque journée sur le terrain, avec un **contrôle qualité hebdomadaire**.

Les données issues des antennes GSSI de 200 MHz et 350 MHz sont enregistrées au format propriétaire `.DZT`, accompagnées de fichiers `.DZX` (métadonnées) et `.DZG` (lorsque l'acquisition est géoréférencée par GPS). Au total, les données brutes collectées représentent **1172 B-Scans**, soit environ 15 Go.

Ces fichiers ont été automatiquement indexés dans un `DataFrame` à l'aide de la bibliothèque `pandas`, ce qui a permis une gestion efficace des métadonnées et des profils associés. Parmi les 1172 fichiers disponibles, **731 contiennent des masques exploitables**, correspondant à des profils annotés ou annotables.

Ce prétraitement constitue une étape clé pour structurer les données en vue des phases ultérieures d'analyse et d'apprentissage. Afin de mieux interpréter les signaux radar et de justifier certains choix de traitement, il est également indispensable de comprendre les caractéristiques du matériel GPR utilisé. C'est l'objet de la section suivante.

Caractéristiques physiques du GPR utilisé

Prenons le fichier **Iradar_005.h5** dont les informations se trouvent dans le tableau 1.5.

name	n_mask	height	width	freq	elev	sol	mask_cat	mask_subcat	mask_cat_conf	mask_subcat_conf
IRADAR_005.h5	Mask_001	913	2594	200 MHz	0	sable sec	OBJET	ABRI_EN_BOIS	5	5
IRADAR_005.h5	Mask_002	913	2594	200 MHz	0	sable sec	OBJET	ABRI_EN_BOIS	3	5
IRADAR_005.h5	Mask_003	913	2594	200 MHz	0	sable sec	RESEAU	BETON	5	5
IRADAR_005.h5	Mask_004	913	2594	200 MHz	0	sable sec	OBJET	FUSILS	2	2
IRADAR_006.h5	Mask_001	913	2685	200 MHz	0	sable sec	OBJET	ABRI_EN_BOIS	5	5

TABLE 1.5 – Tableau des sous-masques et catégories (compact)

Nous pouvons observer le radargramme sur la Figure 1.29 et le masque associé sur la Figure 1.30 ci-dessous.

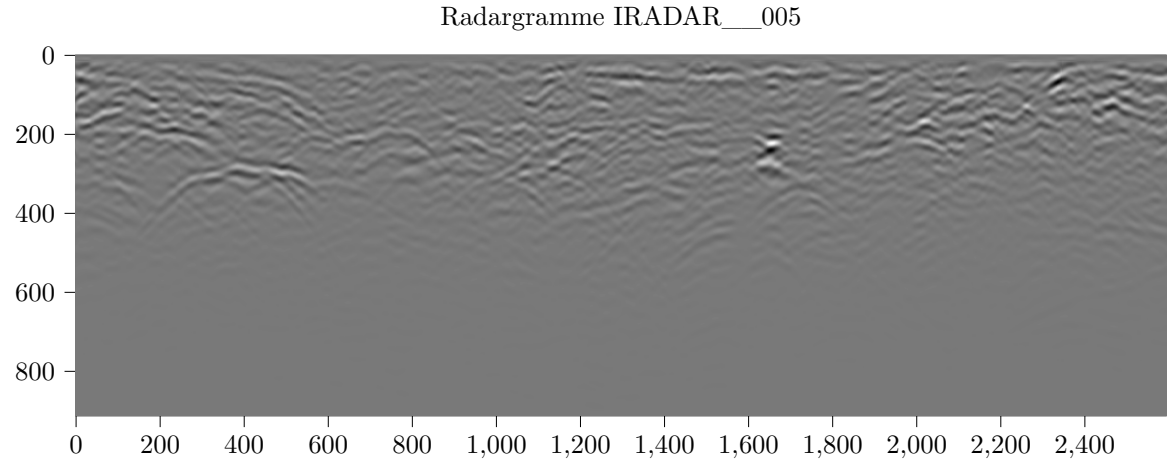


FIGURE 1.29 – Radargramme du Iradar005.

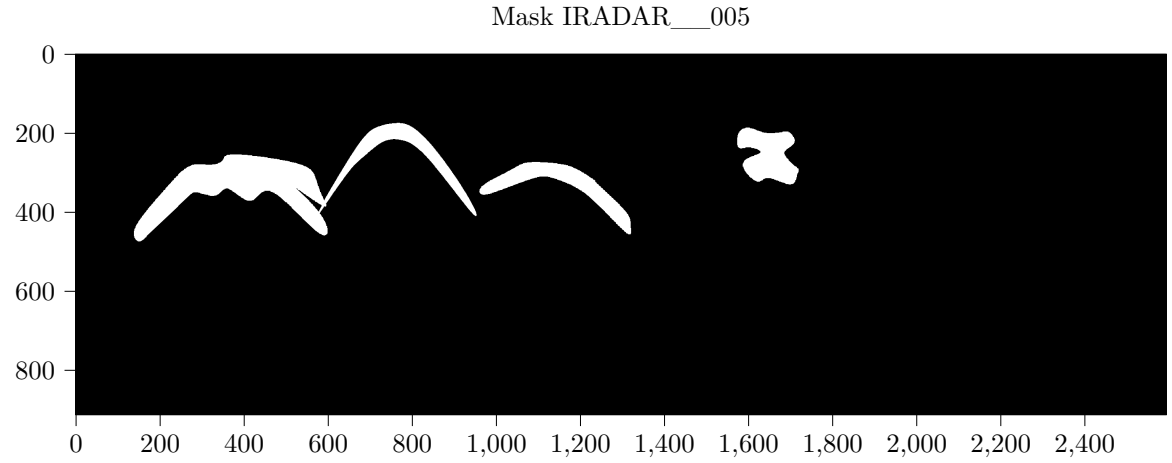


FIGURE 1.30 – Masque du Iradar005.

Chaque objet du masque, que l'on nomme **Mask**, est séparé en un ou plusieurs sous-masques. Ici, nous en avons quatre : **Mask__001**, **Mask__002**, **Mask__003** et **Mask__004**, correspondant à nos quatre objets, comme illustré dans la Figure 1.31. Le masque, ainsi que ses sous-masques, est une image binaire, avec le noir et le blanc comme seules couleurs présentes.

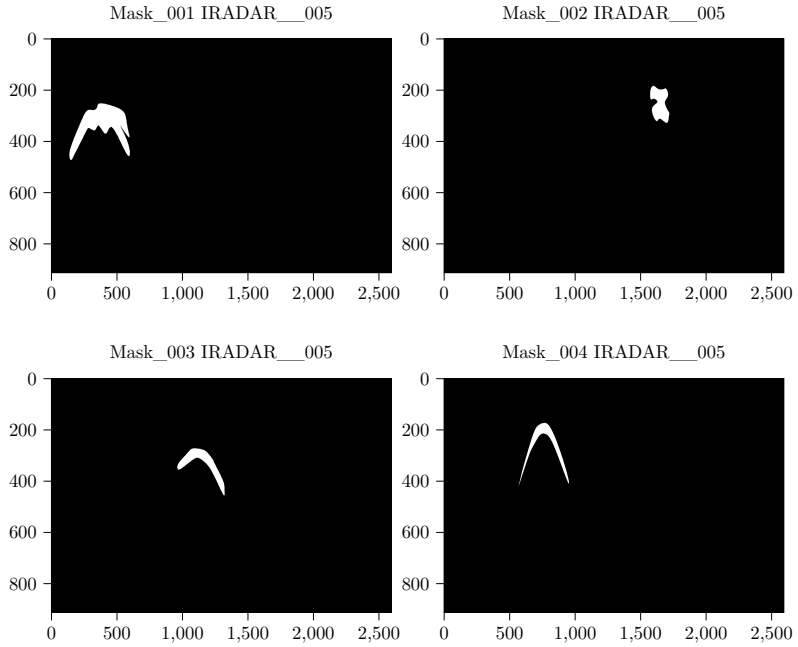


FIGURE 1.31 – Sous-masque du Iradar005.

Les hyperboles présentes dans chaque sous-masque de la Figure 1.31 correspondent à une sous-catégorie, comme décrite dans le tableau 1.5, sans tenir compte de leur nombre. Dans la Figure 1.31, chaque sous-masque contient une hyperbole, mais il se pourrait qu'on en ait plusieurs, comme on le verra plus tard avec la Figure 1.32.

1.6.5 Répartition des sous-catégories dans les sous-masques

Le tableau 1.6 illustre la répartition des sous-catégories annotées pour chacune des trois catégories : **Objet**, **Réseau** et **Discontinuité**, dans l'ensemble des *sous-masques* du jeu de données fourni par Géolithe. Il est important de souligner que cette représentation ne tient pas compte du nombre réel d'hyperboles contenues dans chaque sous-masque : un même masque peut encapsuler plusieurs objets ; c'est le cas lorsque plusieurs hyperboles se chevauchent, comme on le verra plus tard en 1.7.2, tandis qu'un autre n'en contient qu'un seul.

TABLE 1.6 – Répartition des sous-classes par classe

Sous-classe	**OBJET**	**RÉSEAU**	**DISCONTINUITÉ**
ABRI EN BOIS	750		
BETON	257		
FUSILS		355	
VIDE NON METALLIQUE	209		
FUT METALLIQUE		266	
BIDON PLASTIQUE	129		
TELEPHONE	28		
MINE METALLIQUE CYLINDRIQUE	370		
MINE PLASTIQUE CYLINDRIQUE	93		
STRATIGRAPHIE_D			1

Sous-classe	OBJET	RÉSEAU	DISCONTINUITÉ
PLANCHE EN BOIS ENROBEE DE PNEUMATIQUE	48		
MINE PLASTIQUE PAVE	51		
OBUS	182		
REMPLI NON METALLIQUE		20	
PLANCHE EN BOIS	23		
UNKNOWN_O	2		
INTERFACE PEDOLOGIQUE_D			61
UNKNOWN_R		1	
TOTAL	**2142**	**642**	**62**

On observe par ailleurs un déséquilibre important entre les grandes catégories, notamment entre les classes **Objet** et **Discontinuité**, cette dernière étant largement sous-représentée. Ce déséquilibre, conjugué à l'approximation induite par un comptage basé uniquement sur les sous-masques, pose un réel problème pour l'entraînement de modèles d'apprentissage supervisé robustes.

Dans ce contexte, nous proposons de reconstruire une nouvelle base de données à partir de celle existante, en tenant compte du **nombre effectif d'hyperboles** contenues dans chaque sous-masque, tout en opérant un **regroupement cohérent des sous-catégories** en classes principales équilibrées. Cette nouvelle organisation constitue la base du dataset utilisé pour les expériences de classification menées dans cette thèse.

1.7 Construction du dataset de la thèse

1.7.1 Choix de nouvelles classes

Une nouvelle base de données a été créée en regroupant certaines sous-catégories d'objets en trois grandes classes principales, définies comme suit :

- **Classe 1 : Abri en bois**
  - Abri en bois
- **Classe 2 : Métallique**
  - Mine métallique cylindrique
  - Obus
  - Fût métallique
  - Fusils
- **Classe 3 : Non métallique**
  - Mine plastique pavé
  - Béton
  - Vide non métallique
  - Planche en bois
  - Bidon plastique
  - Planche en bois enrobée de pneumatique
  - Téléphone
  - Rempli non métallique
  - Mine plastique cylindrique
- **Classe 4 : Empty**

Pour cette base de données, la catégorie Discontinuité, présente uniquement pour l'élévation 0, n'a pas été prise en compte.

Bien avant de définir nos nouvelles classes, nous avons trouvé intéressant d'ajouter une quatrième catégorie à nos trois catégories, qui sont : Objet, Réseau et Discontinuité. Cette catégorie, que l'on nommera **Empty**, désigne les zones du masque et du radargramme où aucun objet n'est visible.

L'ajout de la catégorie Empty s'avère nécessaire afin de **renforcer la robustesse et la fiabilité du processus d'apprentissage**. En effet, dans les données GPR, un grand nombre d'imagettes extraites ne contiennent aucun objet enfoui identifiable. Ces zones dites « vides » peuvent néanmoins présenter des motifs ou des réflexions parasites susceptibles d'induire en erreur le classifieur si elles ne sont pas explicitement représentées. L'introduction de la classe Empty permet donc de distinguer les véritables signatures d'objets des structures de fond ou du bruit, évitant ainsi les fausses détections et améliorant la capacité du modèle à généraliser sur des données réelles hétérogènes.

Pour identifier les zones sans objets de la catégorie Empty, nous suivons plusieurs étapes. D'abord, pour chaque fichier Iradar, nous extrayons les positions des rectangles qui délimitent les objets sur le radargramme. Chaque rectangle est défini par ses coins, c'est-à-dire par ses coordonnées **x_min, y_min, x_max et y_max**. Ensuite, pour définir la taille des rectangles de la catégorie Empty, nous calculons la longueur et la largeur moyennes de tous les rectangles qui contiennent des hyperboles.

Ensuite, nous générons 64 points uniformément répartis sur le radargramme, organisés en 8 lignes et 8 colonnes. À partir de chaque point, nous dessinons quatre rectangles (en haut à droite, en haut à gauche, en bas à droite et en bas à gauche) de manière à ce qu'ils ne se chevauchent pas, ni entre eux, ni avec les rectangles déjà présents sur le radargramme. Pour chaque point, un seul rectangle est sélectionné afin de garantir une diversité des emplacements. Un paramètre est défini pour limiter le nombre maximal de rectangles de la catégorie Empty sur un radargramme, qui est fixé à 5. Ainsi, pour chaque fichier Iradar, nous identifions au maximum cinq rectangles appartenant à la catégorie Empty.

1.7.2 Prétraitements

Avant de construire notre nouvelle base de données, un prétraitement des données fournies par Géolithe était nécessaire. Les différentes étapes sont décrites ci-dessous.

Description de la pipeline

Dans notre objectif d'entraîner des modèles de classification automatique d'hyperboles, il est essentiel de disposer d'exemples visuels bien localisés et individualisés. Les annotations initiales sont fournies sous forme de *sous-masques binaires*, chacun représentant une ou plusieurs hyperboles appartenant à une même catégorie.

Cependant, ces sous-masques ne permettent pas d'isoler chaque hyperbole de manière indépendante. Or, pour construire une base de données adaptée à l'apprentissage supervisé, nous souhaitons que chaque exemple corresponde à une seule hyperbole clairement délimitée. De plus, cette structuration est également indispensable pour les travaux de détection présentés au chapitre 4, où la localisation précise des objets sous forme de coordonnées est requise.

Pour cette raison, nous proposons de transformer chaque sous-masque en un ou plusieurs **rectangles englobants** ou *Regions of Interest (RoIs)*, chacun entourant une hyperbole distincte, bien que des chevauchements puissent exister. Ces RoIs sont ensuite utilisés pour extraire des imagettes centrées sur chaque hyperbole, qui serviront d'exemples individuels dans notre base de données de classification.

Les étapes pour extraire chaque objet à partir des RoIs sont les suivantes :

1. Récupération du masque pour chaque fichier Iradar.

2. Traitement indépendant de chaque sous-masque associé.

3. Application de l'algorithme Density-Based Spatial Clustering of Applications with Noise **DBSCAN** pour regrouper les points représentant chaque objet du sous-masque en un ou plusieurs **clusters**. Lorsqu'un sous-masque contient une seule hyperbole, le clustering est simple. Cependant, dans le cas de chevauchements (comme dans la Figure 1.32), une **érosion maximale** est parfois nécessaire pour séparer correctement les hyperboles. Après cette érosion maximale, le nombre de clusters est déterminé par **DBSCAN**, et les **centroïdes** des clusters sont obtenus par **K-means**. Lorsque le nombre de clusters de l'image avec érosion maximale est le même que celui de la véritable image, on utilise directement **DBSCAN** et **K-means** sur celle-ci.

4. Traçage d'un rectangle autour de chaque cluster en utilisant les points les plus extrêmes (haut, bas, gauche, droite) du cluster, avec une marge pour englober les objets (dilatés lorsqu'on aura d'abord effectué une érosion maximale).

5. Une fois que tous les rectangles et leurs positions issus du masque ont été obtenus, nous les projetons sur l'image du radargramme, de même taille, afin d'extraire chaque hyperbole du radargramme qui constituera notre jeu de données.

Érosion maximale et dilatation

Avec **DBSCAN**, il est plus facile de détecter les hyperboles dans les sous-masques. Cependant, même dans ces sous-masques, certaines hyperboles peuvent se chevaucher (voir la Figure 1.32). Une **érosion maximale** est utilisée pour séparer les hyperboles qui se touchent.

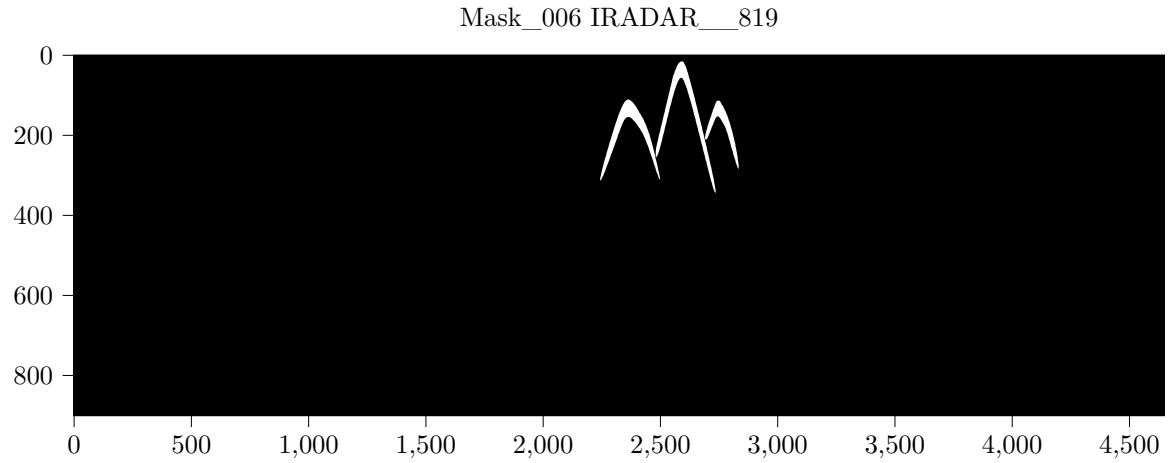


FIGURE 1.32 – Sous-masque 006 du fichier Iradar819. On peut observer trois hyperboles.

Sur la Figure 1.32, trois hyperboles (catégorie **Réseaux**, sous-catégorie *Vide non métallique*) sont très proches. **DBSCAN** regroupe ces hyperboles en un seul cluster (figure 1.33), et le rectangle obtenu englobe toutes les hyperboles à la fois (figure 1.34). Une **érosion maximale** est appliquée pour séparer ces hyperboles (figure 1.35). Une fois séparées, une dilatation est effectuée sur les résultats de l'érosion maximale afin de restaurer la taille initiale des hyperboles, tout en conservant les clusters distincts (figure 1.36).

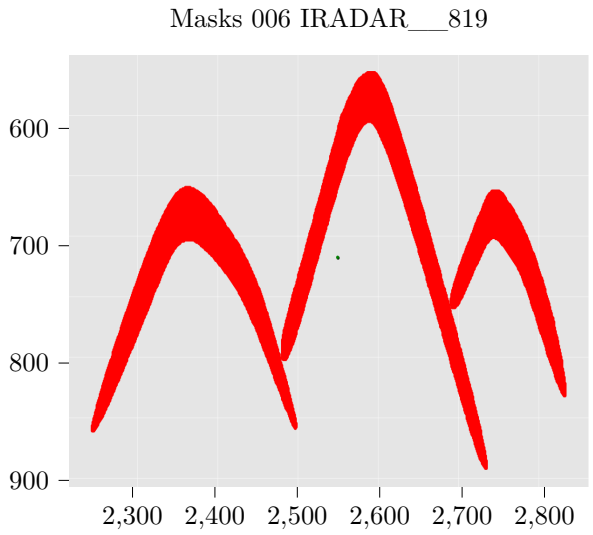


FIGURE 1.33 – Clustering avant érosion maximale du sous-masque Mask_006 d'Iradar819.

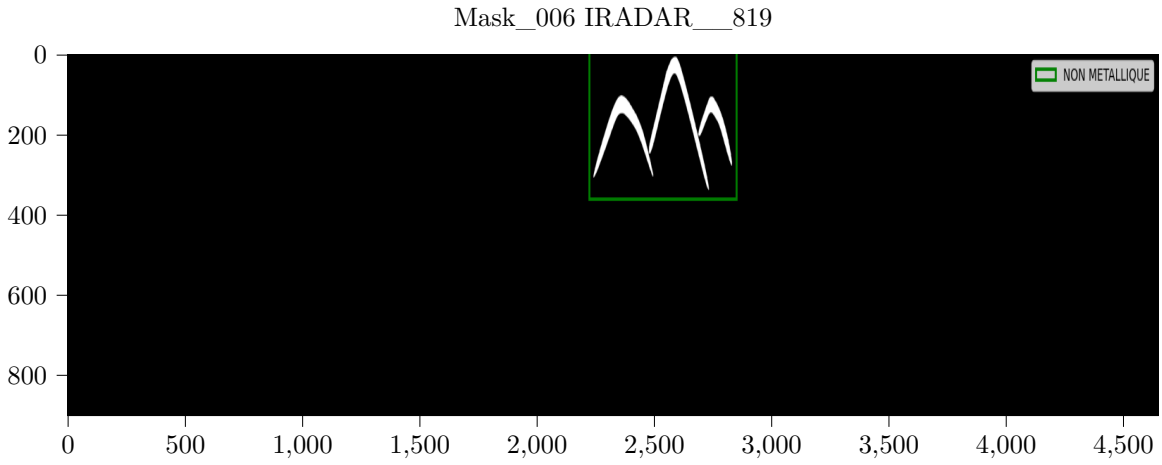


FIGURE 1.34 – Rectangle avant érosion maximale du Mask 006 d'Iradar819.

Après érosion maximale (figure 1.35), trois clusters distincts sont identifiés par **DBSCAN**. Une dilatation est ensuite appliquée pour revenir à la forme initiale des hyperboles (figure 1.36).

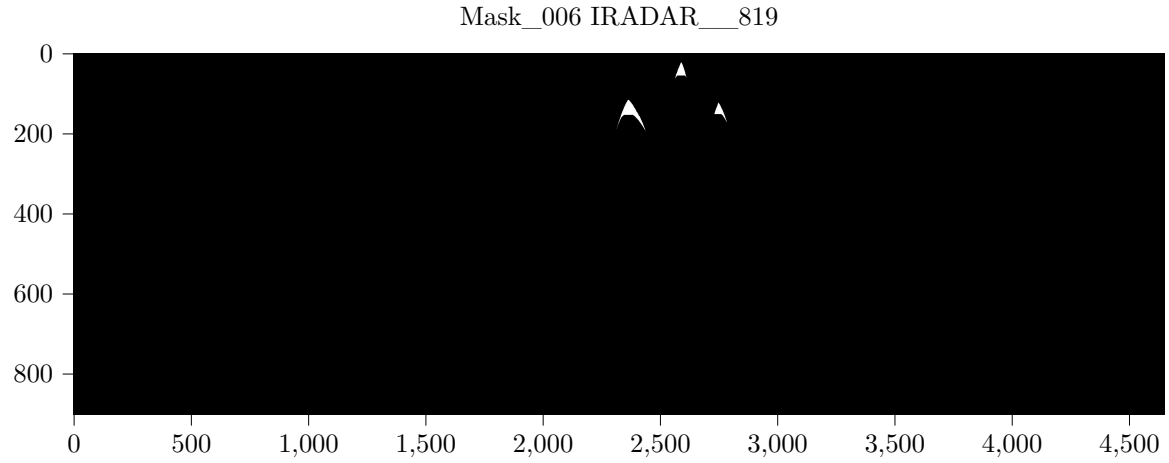


FIGURE 1.35 – Érosion maximale pour obtenir le véritable nombre de clusters.

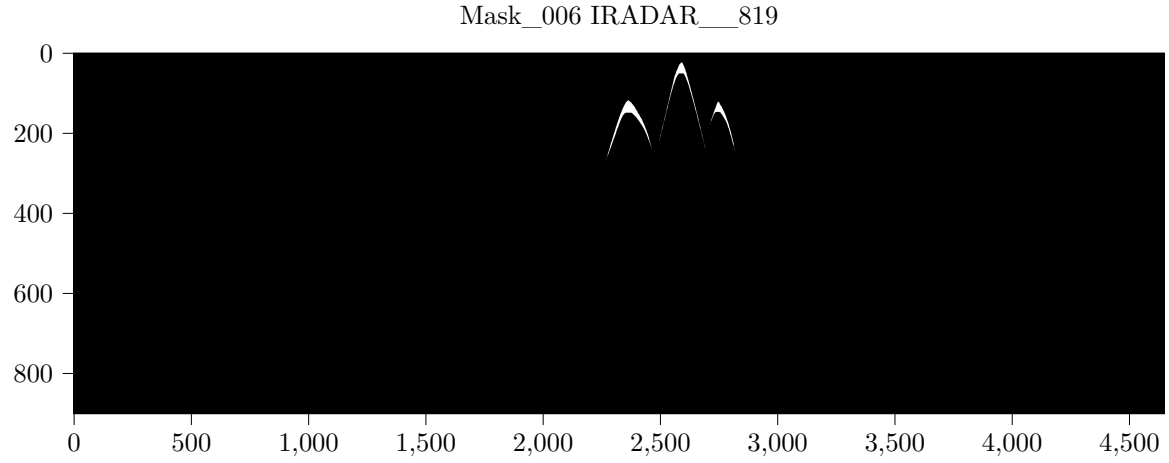


FIGURE 1.36 – Dilatation des résultats de l'érosion maximale pour **DBSCAN**.

Enfin, les rectangles finaux sont ajustés pour englober correctement les objets (figure 1.37).

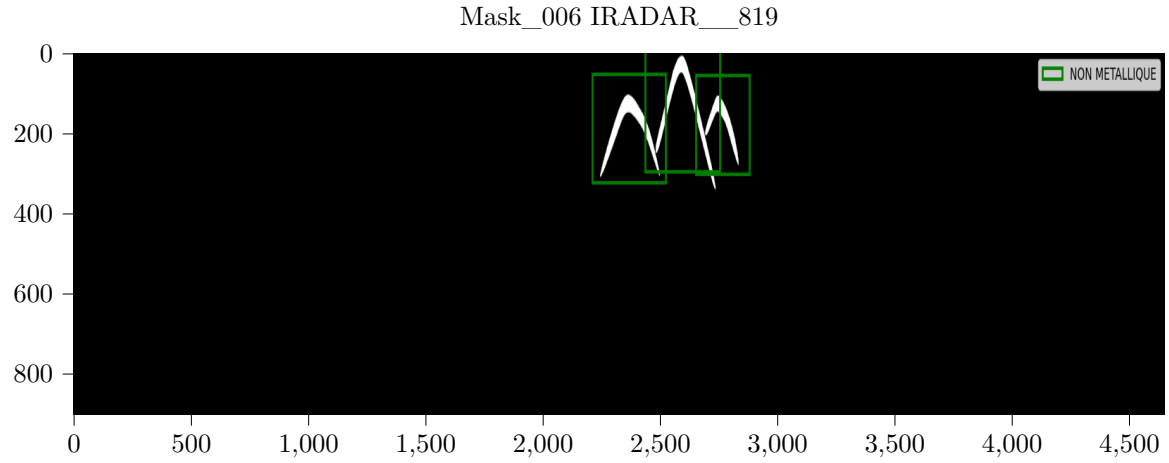


FIGURE 1.37 – Rectangles finaux pour le sous-masque 006 d'Iradar819 après érosion maximale et dilatation.

Résumé L'érosion maximale est appliquée pour identifier le nombre de clusters. Si ce nombre diffère de celui détecté par **DBSCAN** sur l'image originale, les résultats de l'érosion maximale sont dilatés jusqu'à restaurer les dimensions originales des hyperboles. Les rectangles finaux sont ensuite tracés autour des clusters dilatés.

Projections sur le radargramme

Les Figures 1.38 et 1.39 présentent respectivement le masque d'un fichier et le radargramme associé. Une fois tous les rectangles du masque tracés, y compris ceux appartenant à la catégorie **Empty**, leurs positions sont projetées sur le radargramme illustré à la Figure 1.39. Cette projection permet d'extraire chaque rectangle du radargramme, constituant ainsi les éléments de notre nouvelle base de données.

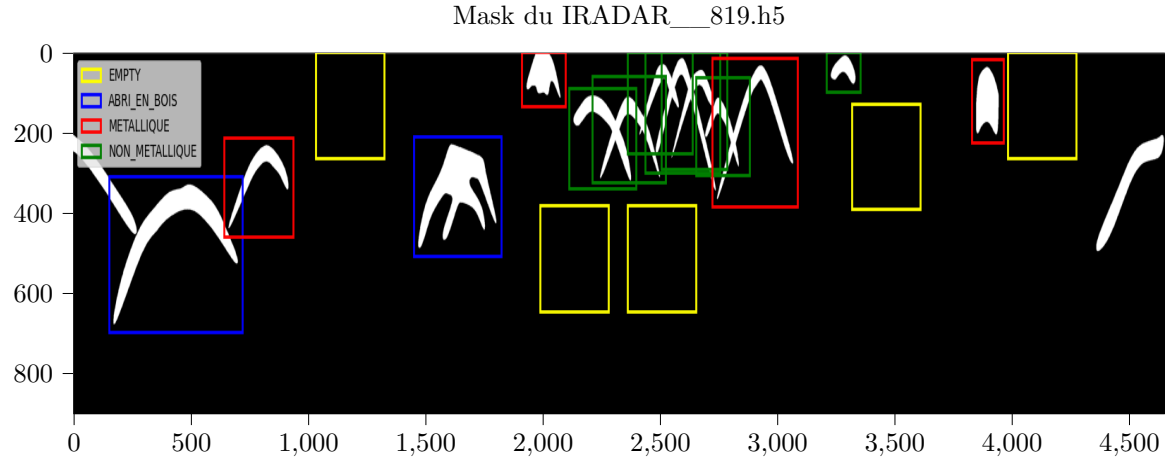


FIGURE 1.38 – Rectangles obtenus pour le radargramme de l'iradar 819 après érosion et DBSCAN , incluant la catégorie **Empty**. Les deux objets situé tout à droite et à gauche n'ont pas été considérées elles font parti de la catégorie Discontinuité.

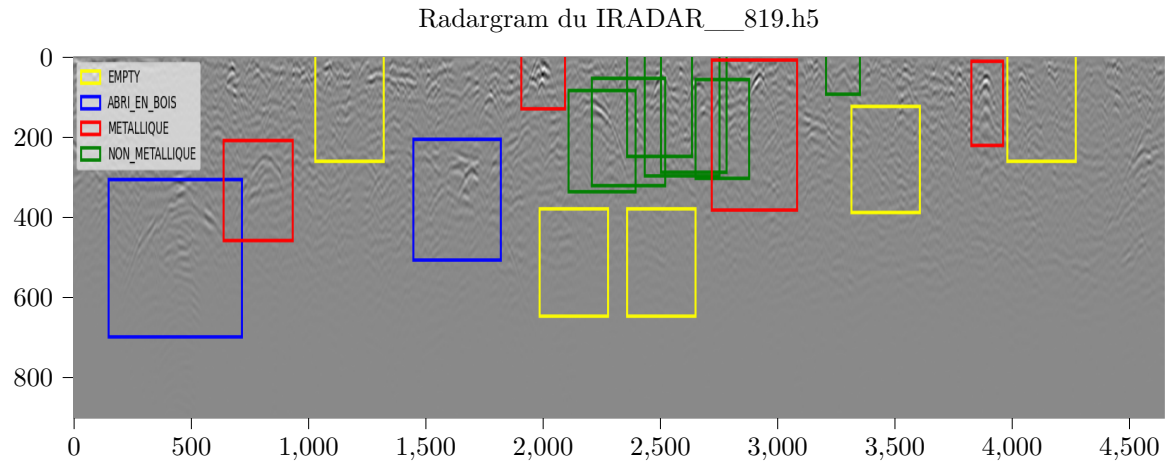


FIGURE 1.39 – Radargramme associé au fichier masqué de l'iradar 819, après érosion et DBSCAN, incluant la catégorie **Empty**.

Les tableaux suivants, 1.7 et 1.8, montrent la distribution des élévations spécifiques à la catégorie **Empty**, en fonction du type de sol et de la fréquence radar.

	Empty						**Total**
Sol/Élévation (cm)	**0**	**25**	**50**	**75**	**100**	**150**	
grave humide	95	110	110	95	75	65	**550**
grave sèche	150	105	105	110	105	100	**675**
sable sec	274	280	280	250	275	260	**1619**
sable humide	170	130	130	135	120	120	**805**
Total	**689**	**625**	**625**	**590**	**575**	**545**	**3649**

TABLE 1.7 – Distribution de chaque élévation en fonction du type de sol filtré pour la classe Empty.

	Empty						**Total**
Frq/Élévation (cm)	**0**	**25**	**50**	**75**	**100**	**150**	
200	294	260	270	265	255	260	**1604**
350	395	365	355	325	320	285	**2045**
Total	**689**	**625**	**625**	**590**	**575**	**545**	**3649**

TABLE 1.8 – Distribution de chaque élévation en fonction de la fréquence filtrée pour la classe Empty.

Resize

Une fois que nous avons récupéré chaque image de rectangle pour l'ensemble de nos quatre classes **Objet, Réseau Discontinuité et Empty**, nous devons les prétraiter afin de pouvoir entraîner un modèle de CNN .

Les étapes du prétraitement sont les suivantes :

1. Tout d'abord, en nous inspirant de l'article [33], chaque image sera redimensionnée à une hauteur de 112 et une largeur de 60, car le CNN ne peut traiter que des images d'une même taille.
2. Ensuite, nous normaliserons nos images afin de faciliter l'apprentissage des modèles CNN.

La Figure ci-dessous nous montre un exemple de prétraitement d'une image

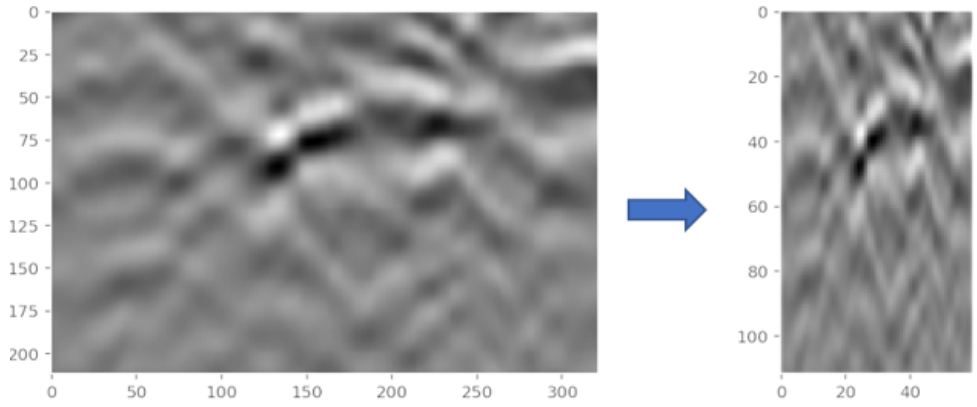


FIGURE 1.40 – Exemple Radargramme redimensionné en 112 x 60 qu'on a ensuite normalisée. Cette normalisation consiste en un centrage-réduction, c'est-à-dire que l'intensité de chaque pixel est recentrée autour de 0 (valeur moyenne 0.5) et mise à l'échelle par un facteur de 0.5, afin de stabiliser l'apprentissage et d'accélérer la convergence du modèle.

1.7.3 Répartition de la Base de données de Thèse

Une fois que nous avons extrait l'ensemble des hyperboles, nous obtenons la répartition suivante de nos données, présente dans les Tableaux 1.9 et 1.10

	Abris en bois						Métallique						Non Métallique						Empty						**Total**
Soil/Elevation (cm)	**0**	**25**	**50**	**75**	**100**	**150**	**0**	**25**	**50**	**75**	**100**	**150**	**0**	**25**	**50**	**75**	**100**	**150**	**0**	**25**	**50**	**75**	**100**	**150**	
grave humide	18	33	23	21	11	16	28	31	36	26	24	10	36	25	25	16	4	0	95	110	110	95	75	65	**933**
grave sèche	34	31	31	24	27	18	61	29	27	30	23	26	51	22	21	19	10	0	150	105	105	110	105	100	**1159**
sable sec	64	67	72	72	79	74	80	97	102	99	116	108	132	90	80	64	48	21	274	280	280	250	275	260	**3084**
sable humide	34	37	35	34	37	33	66	49	49	54	59	60	113	45	44	47	37	8	170	130	130	135	120	120	**1646**
Total	150	168	161	151	154	141	235	206	214	209	222	204	332	182	170	146	99	29	689	625	625	590	575	545	**6822**

TABLE 1.9 – Distribution de notre base de données pour chaque élévation en fonction du type de sol (filtrée pour Empty).

	Abris en bois						Métallique						Non Métallique						Empty						**Total**
Fréquence/Élévation (cm)	**0**	**25**	**50**	**75**	**100**	**150**	**0**	**25**	**50**	**75**	**100**	**150**	**0**	**25**	**50**	**75**	**100**	**150**	**0**	**25**	**50**	**75**	**100**	**150**	
200 MHZ	84	87	89	87	83	82	96	52	48	48	47	60	135	54	46	37	20	1	294	260	270	265	255	260	**2760**
350 MHZ	66	81	72	64	71	59	139	154	166	161	175	144	197	128	124	109	79	28	395	365	355	325	320	285	**4062**
Total	150	168	161	151	154	141	235	206	214	209	222	204	332	182	170	146	99	29	689	625	625	590	575	545	**6822**

TABLE 1.10 – Distribution de notre base de données pour chaque élévation en fonction de la fréquence (filtrée pour Empty).

Base de données utilisées Pour les expérimentations présentées dans le chapitre 2, nous avons donc choisi d'exclure les hyperboles de l'élévation 0, qui ne tiennent pas compte de la hauteur et qui comprenaient des hyperboles de la classe *Discontinuité*, ainsi que celles de l'élévation 150 cm, peu représentées par rapport aux autres élévations, comme le montre le tableau 1.10, notamment pour la catégorie *Non Métallique*. La répartition complète de cette base de données est présentée dans les tableaux 1.11 et 1.12.

	Abris en bois				Métallique				Non Métallique				Empty				**Total**
Soil/Elevation (cm)	**25**	**50**	**75**	**100**	**25**	**50**	**75**	**100**	**25**	**50**	**75**	**100**	**25**	**50**	**75**	**100**	
grave humide	20	16	12	4	13	17	6	11	16	14	8	4	20	12	18	12	**203**
grave sèche	24	18	17	19	18	14	16	6	14	15	14	10	15	19	19	20	**258**
sable sec	37	44	48	54	48	40	55	50	40	44	44	48	45	50	45	43	**735**
sable humide	18	21	22	22	20	28	22	32	29	26	33	37	19	18	17	24	**388**
Total	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	**1584**

TABLE 1.11 – Distribution de notre base de données pour chaque élévation en fonction du type de sol (filtrée pour Empty").

	Abris en bois				Métallique				Non Métallique				Empty				**Total**
Frequency/Elevation (cm)	**25**	**50**	**75**	**100**	**25**	**50**	**75**	**100**	**25**	**50**	**75**	**100**	**25**	**50**	**75**	**100**	
200 MHZ	53	46	54	49	24	13	19	18	26	28	26	20	40	46	45	40	**547**
350 MHZ	46	53	45	50	75	86	80	81	73	71	73	79	59	53	54	59	**1037**
Total	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	**1584**

TABLE 1.12 – Distribution de notre base de données pour chaque élévation en fonction de la fréquence (filtrée pour Empty").

Nous disposons donc au total de 1584 imagettes d'hyperboles. Cette base de données sera utilisée pour les tâches de classification de la section suivante et sera divisée en trois ensembles : entraînement, test et validation, conformément aux pratiques en apprentissage automatique. Le tableau 1.13 montre la répartition de ces trois ensembles en fonction du ratio des données d'entraînement sur nos 1584 images.

Ratio Train	Nb Images Train	Nb Images Val	Nb Images Test
0.1	158	713	713
0.2	317	633	634
0.3	475	554	555
0.4	634	475	475
0.5	792	396	396
0.6	950	317	317
0.7	1108	238	238
0.8	1267	158	159
0.9	1425	79	80

TABLE 1.13 – Répartition des images selon le ratio d'entraînement

1.8 Conclusion

Ce premier chapitre a posé les fondations nécessaires à la compréhension des enjeux liés à l'analyse automatique des données issues du radar à pénétration de sol (*Ground Penetrating Radar*, GPR) dans le contexte spécifique de la détection et de la classification d'objets enfouis.

Nous avons d'abord rappelé les principes physiques de propagation des ondes électromagnétiques dans les milieux diélectriques, ainsi que les spécificités de la réponse radar d'objets enterrés. En particulier, nous avons mis en évidence les difficultés liées à l'interprétation visuelle des radargrammes, notamment en raison de la présence d'artefacts, de bruit, de réflexions multiples ou de clutter provenant des couches du sous-sol.

Ensuite, nous avons présenté en détail la campagne expérimentale menée dans la carrière de Barraux (Isère) par Géolithe. Ce site, instrumenté avec rigueur, a permis l'acquisition de données GPR riches et variées, en modulant plusieurs facteurs : types de sol (sec, humide, grave), élévation des antennes (de 0 à 150 cm) et fréquence d'émission (200 MHz, 350 MHz). La diversité des configurations a été volontairement recherchée afin de générer une base de données représentative des conditions rencontrées sur le terrain.

À partir de ces données, un travail important de préparation a été effectué pour aboutir à un jeu de données exploitable en apprentissage supervisé. Cela inclut :

- des étapes de prétraitement des radargrammes (dewow, suppression du fond, normalisation, etc.) ;
- un processus d'annotation par des experts via une interface dédiée, permettant d'identifier précisément les hyperboles caractéristiques de différents types d'objets (métalliques, non métalliques, structures, etc.) ;
- la construction d'un format de données structuré (`.h5`) intégrant les images, les métadonnées, les masques et les catégories ;
- l'extraction automatique de régions d'intérêt (Région of Interest, RoI) centrées sur les hyperboles annotées, incluant également des zones **Empty** pour modéliser l'arrière-plan ;
- une harmonisation des imagettes (redimensionnement, centrage, normalisation), permettant de constituer une base cohérente de 1584 images réparties en quatre grandes classes : **abri en bois**, **objets métalliques**, **objets non métalliques** et **zones vides**.

L'analyse fine des signatures radar des objets (géométrie, polarité, intensité) a révélé la complexité des réponses et la difficulté de concevoir un modèle analytique unique capable de les capturer toutes. Cela justifie pleinement le recours à des méthodes d'**apprentissage supervisé**, capables d'apprendre ces variations directement à partir des données.

La base de données ainsi constituée représente un socle solide pour explorer différentes approches de classification automatique. Elle sera utilisée dans les chapitres suivants pour comparer des méthodes classiques et des modèles d'apprentissage profond.

Dans le **chapitre 2** suivant, nous évaluons plusieurs stratégies de classification supervisée, des plus classiques (SVM, Random Forest) aux plus avancées (CNN, covariance pooling). L'objectif est de déterminer dans quelle mesure ces modèles peuvent exploiter les propriétés visuelles et statistiques des hyperboles afin d'améliorer la reconnaissance automatique des objets enfouis dans les images GPR.

Chapitre 2

Classification d'images de GPR : Approches classiques et modernes

La classification des données issues du géoradar (GPR) repose sur un éventail d'approches, allant de modèles classiques tels que les machines à vecteurs de support (SVM) aux réseaux de neurones convolutionnels peu profonds, désignés ici par S-CNN (Shallow Convolutional Neural Networks), jusqu'à des architectures plus profondes comme ResNet, exploitées avec ou sans transfert d'apprentissage. Toutefois, les travaux spécifiquement consacrés à ce domaine restent relativement rares dans la littérature GPR. Cette section examine successivement ces différentes approches, puis introduit une méthode avancée basée sur une représentation globale des images de notre jeu de données, obtenue par le calcul d'une matrice de covariance à partir des activations extraites d'un CNN.

Sommaire

2.1 Approches classiques

Les approches classiques de classification, telles que les Machines à vecteurs de support, ou SVM (pour *Support Vector Machine* en anglais), sont largement utilisées dans divers domaines, y compris pour la classification des images GPR [6,7] ainsi que les forêts aléatoires, ou RF (pour *Random Forests* en anglais) [34, 35]. Avant d'explorer des modèles plus complexes basés sur l'apprentissage profond, nous évaluons les performances de ces méthodes supervisées sur notre jeu de données. L'objectif est d'analyser leur capacité à distinguer les différentes classes et de comparer leurs performances avec celles d'architectures neuronales plus avancées.

2.1.1 Préparation des données

Afin de comparer les performances des méthodes classiques aux modèles d'apprentissage profond, nous avons défini un pipeline de traitement appliqué aux trois sous-ensembles : entraînement, validation et test. Chaque échantillon de notre jeu de données est une image GPR de taille 112×60, que nous avons prétraitée comme suit :

- **Vectorisation et normalisation (ou standardisation) des images :** Chaque image bidimensionnelle est vectorisée en un vecteur colonne $\mathbf{x}_i \in \mathbb{R}^{6720}$ (avec $6720 = 112 \times 60$), et l'ensemble des images est regroupé sous la forme d'une matrice $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^\top \in \mathbb{R}^{N \times 6720}$, où N est le nombre d'échantillons. Une standardisation est ensuite appliquée sur chaque colonne de $\mathbf{X}$, afin de centrer les données (moyenne nulle) et de les normaliser (variance unitaire).
- **Réduction de la dimensionnalité (ACP) :** Étant donné la haute dimension des vecteurs d'entrée, une Analyse en Composantes Principales (ACP) [36] est appliquée afin de projeter les données dans un sous-espace de dimension plus faible, tout en conservant un maximum d'informations.

  La matrice standardisée $\mathbf{X}$ est transposée, puis soumise à une décomposition en valeurs singulières (SVD) :

  $$\mathbf{X}^\top = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top,$$

  où les colonnes de $\mathbf{U} \in \mathbb{R}^{6720 \times d}$ représentent les directions principales (composantes), et les valeurs singulières dans $\mathbf{\Sigma}$ permettent de quantifier la variance expliquée.

  Pour évaluer la quantité d'information préservée par les k premières composantes principales, nous utilisons le *pourcentage cumulé de variance expliquée* (Cumulative Proportion of Explained Variance, CPEV), défini par :

  $$\text{CPEV}(\mathbf{U}_k) = \frac{\text{Tr}(\mathbf{U}_k^\top \mathbf{X}^\top \mathbf{X} \mathbf{U}_k)}{\text{Tr}(\mathbf{X}^\top \mathbf{X})},$$

  où $\mathbf{U}_k \in \mathbb{R}^{6720 \times k}$ contient les k premiers vecteurs propres. Cette quantité mesure la part de variance capturée par la projection dans le sous-espace défini par $\mathbf{U}_k$, relativement à la variance totale du jeu de données.

  Le nombre optimal de composantes k est alors déterminé comme le plus petit entier tel que $\text{CPEV}(\mathbf{U}_k) \geq 0.95$, c'est-à-dire que l'on conserve au moins 95 % de la variance initiale. Ce critère garantit que la réduction de dimension conserve l'essentiel de l'information, tout en limitant la complexité computationnelle.

  Une fois k déterminé, les données standardisées sont projetées dans ce sous-espace réduit :

  $$\tilde{\mathbf{X}} = \mathbf{X} \cdot \mathbf{U}_k,$$

  ce qui permet d'obtenir une représentation plus compacte, utilisée ensuite comme entrée pour les algorithmes de classification, dont SVM et les forêts aléatoires.

2.1.2 Support Vector Machines (SVM)

Les Machines à vecteurs de support, ou SVM (pour *Support Vector Machine* en anglais), sont des algorithmes d'apprentissage supervisé qui cherchent à séparer les échantillons de différentes classes à l'aide d'une frontière de décision maximisant la marge entre les points les plus proches de cette frontière [37]. Ce principe confère aux SVM une bonne capacité de généralisation, en particulier dans les cas où les classes sont difficilement séparables de manière linéaire [38].

Pour prendre en compte des données non linéairement séparables, les SVM peuvent être étendus à l'aide de fonctions de transformation, ou *noyaux*, qui projettent les données dans un espace de dimension supérieure où la séparation linéaire est possible (voir annexe A). Dans le cadre de problèmes de classification multiclasse, le SVM peut également être généralisé en entraînant un classifieur binaire pour chaque classe, en distinguant cette classe de l'ensemble des autres [39]. Chaque classifieur produit un score ou une probabilité d'appartenance à sa classe, et la classe finale attribuée à un échantillon est celle dont le score est maximal. Cette approche permet d'appliquer le SVM, initialement conçu pour la classification binaire, à des jeux de données comportant plusieurs catégories, comme c'est le cas pour nos imagettes d'hyperboles.

2.1.3 Forêts Aléatoires, Random Forests (RF)

Les forêts d'arbres décisionnels (ou forêts aléatoires, de l'anglais *Random Forest*, RF) ont été initialement proposées par Ho [40] en 1995, puis formellement développées en 2001 par Leo Breiman [41] et Adele Cutler. Elles font partie des techniques d'apprentissage automatique. Cet algorithme combine les concepts de sous-espaces aléatoires et de *bootstrap*, surnommé bagging par Leo Breiman lui-même. Ce sont des algorithmes qui combinent plusieurs arbres de décision pour améliorer la robustesse et la précision de la classification. Contrairement aux arbres de décision individuels, qui peuvent être sensibles au bruit et au surapprentissage, les RF exploitent la diversité de plusieurs arbres entraînés sur des sous-échantillons aléatoires des données. La prédiction finale est obtenue par un vote majoritaire des arbres, ce qui permet de réduire la variance du modèle.

2.1.4 Classification après analyse en composantes principales (ACP)

Dans cette approche, les données sont d'abord projetées dans un espace de dimension réduite via une *analyse en composantes principales (ACP)* afin de conserver l'information la plus pertinente tout en réduisant la complexité. Les composantes principales ainsi obtenues sont ensuite utilisées comme entrée pour un classifieur, soit un SVM, soit une forêt aléatoire (Random Forest). On parle donc de modèles combinés *ACP avec SVM* ou *ACP avec RF*.

La Figure 2.1 illustre le pipeline combinant réduction de dimension et classification.

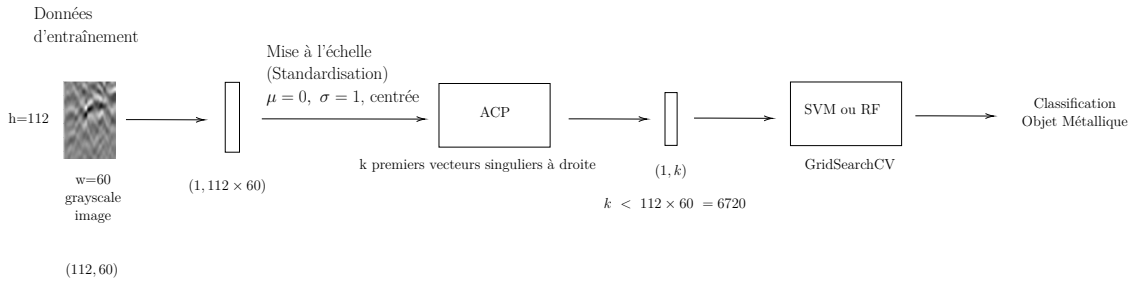


FIGURE 2.1 – Schéma du pipeline combinant réduction de dimension par ACP suivie d'une classification par SVM ou Random Forest (*ACP avec SVM* ou *ACP avec RF*). La recherche sur grille des hyperparamètres (`GridSearchCV`) est indiquée sur la Figure, mais sera détaillée dans la sous-section suivante.

2.1.5 Stratégies d'entraînement et optimisation des hyperparamètres

Afin de garantir une méthodologie rigoureuse et reproductible avec `scikit-learn`, les deux modèles classiques étudiés dans ce chapitre (ACP¨avec SVM et ACP avec RF) suivent les mêmes stratégies pour l'optimisation de leurs hyperparamètres et l'évaluation de leurs performances.

Recherche sur grille (*Grid Search*) L'optimisation des hyperparamètres a été réalisée à l'aide d'une recherche sur grille systématique, permettant d'explorer un ensemble discret de combinaisons possibles. Chaque configuration a été évaluée via une validation croisée à cinq plis (*5-fold cross-validation*) afin de sélectionner celle maximisant la précision moyenne sur les plis de validation [1]. Cette procédure a été implémentée avec la classe `GridSearchCV` de `scikit-learn`. Dans cette étude, elle a été appliquée aux modèles SVM et RF. Pour le SVM, on utilisera SVC (Support Vector Classifier), qui est la classe implémentant le SVM pour la classification dans `scikit-learn`.

- **SVC :** `kernel` : fonction de transformation des données (`rbf` utilisée ici ; d'autres noyaux sont présentés en annexe A). `gamma` : paramètre du noyau contrôlant l'influence d'un point d'entraînement ({1e-3, 1e-4}), `C` : paramètre de régularisation, compromis entre la maximisation de la marge et la minimisation des erreurs de classification ({1, 10, 100, 1000}) et `decision_function_shape` : stratégie multiclasse utilisée (`'ovr'` par défaut, correspondant à One-vs-All (OvA)).
- **RF :** `criterion` : critère de séparation des nœuds (**indice de Gini** [2]) , `n_estimators` : nombre d'arbres dans la forêt ({100, 300}), `min_samples_split` : nombre minimal d'échantillons pour diviser un nœud ({2, 4}) et `max_depth` : profondeur maximale des arbres ({50, 100, 200}).

La validation croisée à 5 plis (*5-fold cross-validation*) intégrée à la recherche sur grille permet d'évaluer la performance moyenne d'un ensemble de paramètres sur plusieurs sous-ensembles de l'ensemble d'entraînement. Cette étape ne constitue pas l'évaluation finale des modèles sur le jeu de test, mais garantit que les hyperparamètres sélectionnés sont à la fois robustes et généralisables.

Synthèse des modèles Cette méthodologie définit un cadre expérimental commun pour les modèles SVM et RF après réduction de dimension par ACP, garantissant des comparaisons équitables entre ces approches classiques et les architectures profondes. Les expériences finales sur les données de test, incluant les répétitions et la variation des partitions, sont présentées dans la section 2.1.6.

Les performances des deux modèles sont comparées à celles des autres méthodes étudiées afin d'évaluer leur pertinence pour la classification des images radar GPR. Dans les tableaux de résultats, les modèles sont désignés respectivement de la manière suivante :

$$SVC\_gs\_std\_pca \quad \text{et} \quad RF\_gs\_std\_pca.$$

Ces notations indiquent que les hyperparamètres ont été optimisés par recherche sur grille (`gs`) et que les données ont été standardisées (`std`) avant d'être projetées dans un sous-espace de dimension réduite via une analyse en composantes principales (`pca`), où la standardisation est effectuée à l'aide de la méthode `StandardScaler` [3] de `scikit-learn`

Les performances de ces modèles sont évaluées sur l'ensemble de test pour différents ratios de données d'entraînement, avec 100 graines aléatoires pour mesurer leur robustesse face aux variations de

1. Chaque pli est utilisé à tour de rôle comme ensemble de test, les autres servant à l'apprentissage.
2. L'indice de Gini mesure l'impureté d'un ensemble de données. Il est égal à 0 si tous les échantillons appartiennent à la même classe et atteint son maximum lorsque les classes sont équitablement réparties.
3. La méthode `StandardScaler` applique une transformation des données selon :

$$x_j^{\text{std}} = \frac{x_j - \mu_j}{\sigma_j}$$

où μ_j est la moyenne et σ_j l'écart-type de la j-ième variable sur l'ensemble d'apprentissage. Documentation : `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html`.

partitionnement. Les résultats obtenus permettront de comparer la précision de SVC_gs_std_pca et RF_gs_std_pca aux autres approches étudiées dans ce chapitre.

2.1.6 Expérimentations

Pour évaluer la robustesse des modèles, nous procédons comme suit :

- Pour chaque ratio d'entraînement $r \in [0.1, 0.9]$, les modèles SVC_gs_std_pca et RF_gs_std_pca sont entraînés sur **100 graines aléatoires** différentes.
- Les partitions sont stratifiées par classe afin de conserver la proportion des catégories.
- La précision des données de test est calculée pour chaque expérience réalisée.
- Les résultats sont présentés sous forme de moyennes et d'intervalles de confiance (5e–95e quantiles) pour évaluer la stabilité et la sensibilité des modèles aux variations de partition.

Cette procédure se poursuivra dans l'ensemble des prochaines sections de ce chapitre.

Les résultats obtenus avec ces deux méthodes classiques sont présentés dans la Figure 2.2. Les performances dépendent fortement de la qualité des caractéristiques extraites. Si ces descripteurs ne capturent pas efficacement l'information pertinente contenue dans les images GPR, les classifieurs risquent d'être limités dans leurs performances.

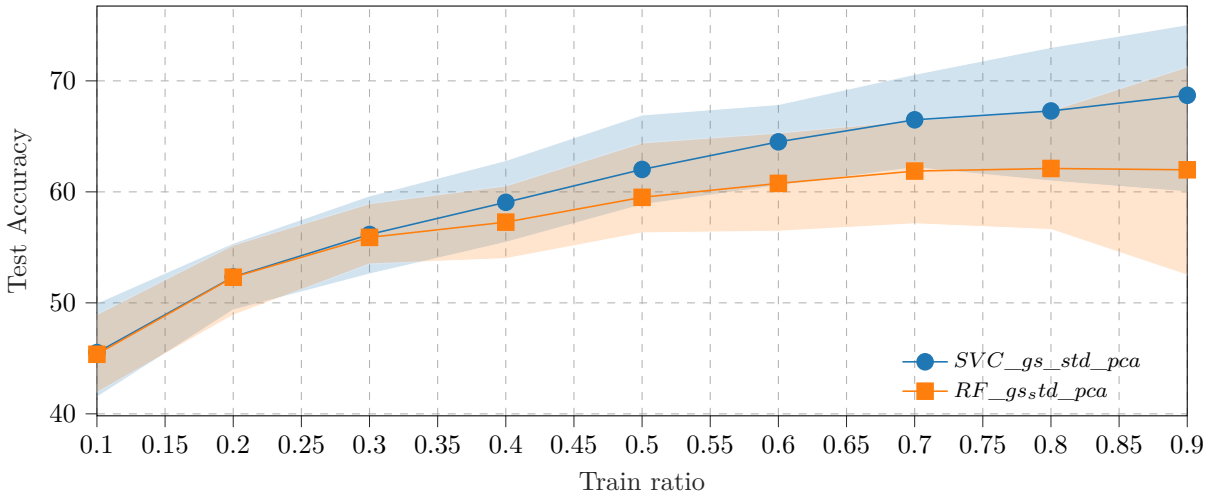


FIGURE 2.2 – Comparaison des résultats avec les modèles classiques d'apprentissage automatique. Précision sur les données test en fonction du ratio de l'ensemble de données d'entraînement. La ligne correspond à la moyenne sur 100 expériences, la zone remplie aux 5e et 95e quantiles.

Nous observons que, pour des ratios inférieurs à 0.4, les performances des deux modèles sont relativement comparables. Au-dessus de cette valeur, la méthode avec SVM devient plus performante, atteignant une précision de près de 70% sur les données de test lorsque l'ensemble d'entraînement représente environ 90% des données disponibles. Toutefois, même avec un grand volume de données d'entraînement, les performances restent limitées. Le modèle SVC_gs_std_pca, ayant obtenu les meilleurs résultats, est conservé pour la suite des comparaisons.

L'Analyse en Composantes Principales (ACP), suivie d'un classifieur SVM ou Random Forest, a été explorée dans un premier temps pour la classification des images GPR. Bien que cette approche permette de réduire la dimensionnalité des données et de capturer certaines variations principales, l'ACP reste sensible au bruit et peut mal représenter les structures complexes présentes dans les radargrammes.

Afin d'améliorer les résultats et de surmonter ces limitations, nous proposons d'explorer une stratégie alternative, fondée sur l'utilisation de descripteurs statistiques plus riches que la simple vectorisation des images. Nous allons nous intéresser aux matrices de covariance, connues pour leur robustesse face aux transformations géométriques. L'enjeu est de déterminer comment extraire ces matrices à partir des images GPR et d'évaluer leur pertinence pour améliorer la classification.

2.2 Méthode à partir des covariances

Les approches précédemment explorées, fondées sur l'ACP suivie d'un classifieur SVM ou de Random Forest, ont permis de réduire la dimensionalité et de capturer certaines variations principales des données. Toutefois, la sensibilité de l'ACP au bruit et sa difficulté à représenter des structures complexes limitent ses performances dans le contexte des radargrammes GPR.

Afin de dépasser ces limitations, nous proposons d'explorer une stratégie alternative basée sur des descripteurs statistiques de second ordre : les matrices de covariance. Ces représentations permettent de capturer les corrélations internes entre les caractéristiques et offrent une modélisation plus robuste que la simple vectorisation ou projection linéaire des données.

Cette approche exploite la géométrie riemannienne associée à l'espace des matrices SPD, offrant une meilleure expressivité, en particulier lorsque le nombre d'exemples d'entraînement est limité. Inspirée des travaux de [42] et [11], elle consiste à extraire des caractéristiques à l'aide d'un CNN, puis à en construire des matrices de covariance pour former des descripteurs fiables et performants. Elle s'est également révélée efficace dans d'autres contextes, tels que l'analyse des signaux EEG [43] ou l'imagerie multispectrale [44], démontrant sa polyvalence et son potentiel pour la classification des données GPR.

2.2.1 Principes généraux des CNN et motivations

Les réseaux de neurones convolutionnels (CNN) reposent sur l'opération de convolution, qui consiste à appliquer un filtre (*kernel*) sur l'image pour détecter des motifs locaux tels que des contours, des textures ou des variations d'intensité. Chaque filtre produit une carte de caractéristiques (*feature map*) mettant en évidence la présence de ces motifs.

Les couches convolutionnelles sont suivies de fonctions d'activation non linéaires et de couches de *pooling*, qui réduisent la dimension spatiale tout en préservant les informations importantes, rendant ainsi les représentations moins sensibles aux petites translations et aux variations locales causées par le bruit.

Dans cette étude, les CNN sont utilisés uniquement comme extracteurs de caractéristiques ou *backbone*. Les cartes issues des premières couches du réseau constituent les représentations initiales qui seront exploitées pour construire des descripteurs plus riches basés sur les matrices de covariance, comme détaillé dans la section suivante.

2.2.2 Modèles proposés

À partir des cartes de caractéristiques extraites par le CNN, nous détaillons ici les principales étapes du modèle basé sur les matrices de covariance. Contrairement aux approches précédentes reposant sur l'ACP suivie d'un SVM ou d'une Random Forest, la classification s'appuie désormais sur ces matrices plutôt que directement sur les images GPR prétraitées. Le processus de construction de ces matrices et leur utilisation pour la classification est présenté ci-après.

Extraction des caractéristiques par un CNN : Nous utilisons les l premières couches d'un CNN non préentraîné pour extraire des caractéristiques adaptées aux images GPR. Les résultats préliminaires

montrent que les performances sont meilleures avec un réseau entraîné from scratch qu'avec un modèle préentraîné sur des données de vision par ordinateur. [4]

Cette observation s'explique donc probablement par le fait que les premières couches des CNN capturent principalement des motifs simples, tels que les bords et les textures, qui sont directement exploitables pour les images GPR.

Chaque couche du réseau produit un ensemble de cartes de caractéristiques (*feature maps*), notées $\mathbf{I}_i \in \mathbb{R}^{h_i \times w_i}$, où i désigne l'indice du filtre. Ces cartes contiennent des informations locales essentielles pour l'analyse des motifs enfouis. Dans ce travail, nous partons d'images en niveaux de gris $\mathbf{I} \in \mathbb{R}^{h \times w}$, avec des dimensions initiales de $h = 112$ et $w = 60$, ce qui facilite la comparaison avec des études futures [8].

Lorsqu'un ResNet-34 est utilisé, nous limitons l'extraction aux $l = 8$ premières couches, chaque couche produisant 64 cartes de caractéristiques. Pour réduire la consommation de mémoire, le nombre de cartes conservées par couche est limité à $\bar{d} = 32$, tout en préservant les informations les plus discriminantes pour la classification.

Construction du tenseur des caractéristiques : Les étapes pour construire le tenseur des caractéristiques sont les suivantes :

- Pour chacune des l premières couches du CNN, nous conservons les $\bar{d}$ premiers filtres. Par conséquent, le nombre total de sorties des filtres est donné par $d = l \cdot \bar{d}$.
- Soit $\mathbf{I}_i \in \mathbb{R}^{h_i \times w_i}$ avec $i \in [1, d]$, représentant l'ensemble des sorties des filtres de toutes les couches.
- Définissons M_h comme la moyenne des h_i et M_w comme la moyenne des w_i.
- Redimensionnons chaque sortie de filtre $\mathbf{I}_i$ en $\tilde{\mathbf{I}}_i \in \mathbb{R}^{M_h \times M_w}$.
- Empilons ensuite les sorties redimensionnées $\tilde{\mathbf{I}}_i$ pour former un tenseur $\mathcal{T}$, défini comme $\mathcal{T} = \{\tilde{\mathbf{I}}_i\}_{i \in [1,d]} \in \mathbb{R}^{d \times M_h \times M_w}$.
- Enfin, remodelons $\mathcal{T}$ en une matrice $\mathbf{T}$, où $\mathbf{T} \in \mathbb{R}^{d \times M}$ avec $M = M_h \cdot M_w$.

Couche de Covariance Pooling (CovPool) [5]

Une fois le tenseur des caractéristiques $\mathbf{T}$ construit, une couche de pooling basée sur la covariance (*Covariance Pooling Layer*) est appliquée pour capturer les relations statistiques entre les caractéristiques. Ce processus est réalisé comme suit :

$$\mathbf{C} = \mathbf{T}\bar{\mathbf{I}}\mathbf{T}^{\mathrm{T}},$$

où :

- $\bar{\mathbf{I}} = \frac{1}{N}\left(\mathbf{I} - \mathbf{1}_N \mathbf{1}_N^{\mathrm{T}}\right)$ est une matrice de centrage qui normalise les données.
- $\mathbf{C} \in \mathbb{R}^{d \times d}$ est la matrice de covariance, appartenant au manifold des matrices symétriques définies positives (*Symmetric Positive Definite*, SPD).

Classification basée sur les matrices de covariance : Les matrices de covariance $\mathbf{C}$ obtenues servent de descripteurs robustes pour la classification à l'aide de la librairie `scikit-learn`. Après la vectorisation ou transformation appropriée, elles sont utilisées avec des algorithmes tels que :

4. Nous étudierons plus tard les différences entre un modèle préentraîné et un modèle initialisé aléatoirement, c'est-à-dire le transfert learning versus l'approche *from scratch*. Ici, pour l'extraction des caractéristiques, le CNN est entraîné *from scratch*.

5. Il s'agit juste de l'estimateur de covariance en gaussien, c'est-à-dire la Sample Covariance Matrix (SCM). On garde le terme CovPool qui est couramment utilisé en ML.

- **SVC_gs_covpool :** Les matrices, centrées puis vectorisées, sont utilisées pour entraîner un SVM en appliquant la même grille de recherche que celle présentée dans la section 2.1.5, afin d'identifier des frontières optimales entre les classes.
- **RF_gs_covpool :** Ces matrices, centrées puis vectorisées, sont également utilisées pour entraîner un Random Forest en appliquant la même grille de recherche que celle présentée dans la section 2.1.5, garantissant à la fois robustesse et performance.

Schéma des modèles : La Figure 2.3 illustre le processus complet de classification basé sur les matrices de covariance.

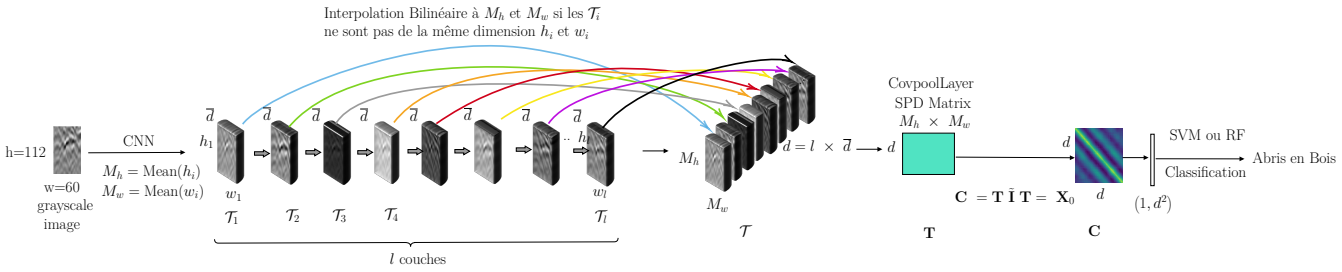


FIGURE 2.3 – Processus de pooling basé sur la covariance appliqué à un radargramme.

2.2.3 Expérimentations

Nous nous plaçons dans la même configuration que celle des expérimentations précédentes de 2.1.6. Sur la Figure 2.4, on observe que nos deux modèles basés sur la covariance obtiennent toujours de meilleurs résultats que ceux du SVC_gs_std_pca. En l'occurrence, SVC_gs_covpool atteint les meilleures performances et, donc, les covariances sont très intéressantes. On le conservera pour la suite.

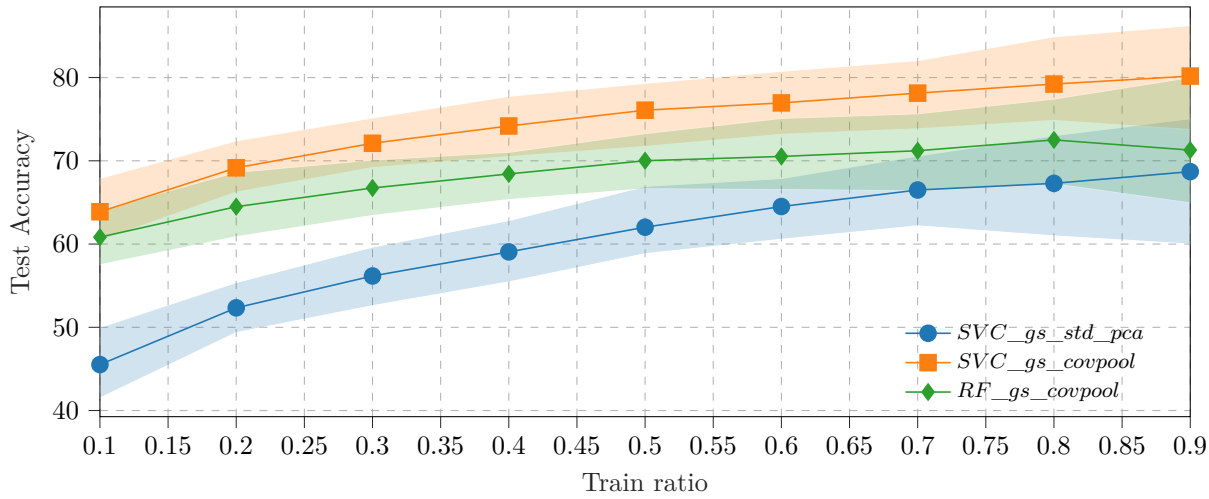


FIGURE 2.4 – Comparaison des résultats avec les modèles à partir des covariances (SVC_gs_covpool et RF_gs_covpool) auxquelles nous avons ajouté les résultats de SVC_gs_std_pca. Précision sur les données de test par rapport au pourcentage de l'ensemble de données d'apprentissage sur 100 graines différentes. Pour chaque méthode, la ligne correspond à la moyenne de la précision sur toutes les graines des données de test, et la zone remplie correspond aux 5ème et 95ème quantiles.

Les méthodes exploitant les matrices de covariance extraites des sorties intermédiaires d'un CNN entraîné *from scratch* présentent plusieurs avantages par rapport à l'approche ACP suivie d'un SVM.

Premièrement, elles permettent de capturer les dépendances spatiales et les corrélations complexes entre les différentes cartes de caractéristiques (*feature maps*) produites par les couches convolutives. Ces corrélations contiennent des informations discriminantes que l'ACP, centrée sur les composantes principales, ne peut pas toujours représenter, en particulier pour des structures non linéaires.

Deuxièmement, les matrices de covariance offrent une représentation plus riche et robuste. Contrairement à la simple vectorisation ou à l'ACP, elles conservent les interactions entre les caractéristiques extraites par le CNN, ce qui confère une meilleure résistance aux variations expérimentales fréquentes dans les données GPR, telles que le bruit, les différences d'élévation ou les conditions environnementales.

Enfin, l'entraînement *from scratch* du CNN permet aux filtres convolutifs de s'adapter aux spécificités des images GPR. Les matrices de covariance issues de ces filtres (couches de sortie de filtre) contiennent donc des informations hiérarchiques et non linéaires particulièrement adaptées à la tâche de classification, difficilement capturables par des méthodes linéaires classiques.

Conclusion. En pratique, l'exploitation des matrices de covariance dérivées d'un CNN entraîné *from scratch* a conduit à des performances supérieures par rapport à l'approche ACP avant SVM. Cette amélioration résulte de la richesse des représentations capturées par les covariances des cartes de caractéristiques ainsi que de la capacité du CNN à apprendre des motifs discriminants spécifiques aux images GPR.

Nous allons maintenant nous intéresser aux CNN pour la classification directe des images GPR, qui sont reconnus pour offrir de meilleures performances que les algorithmes classiques, tels que les SVM. Nous commencerons toutefois par étudier les architectures peu profondes (*shallow*).

2.3 S-CNN : Shallow Convolutional Neural Networks

Les *Shallow Convolutional Neural Networks* (S-CNN), proposés par [33], constituent une solution légère et efficace pour la classification des images GPR. Ces modèles sont conçus pour extraire rapidement les caractéristiques pertinentes tout en maintenant une faible complexité computationnelle.

2.3.1 Architecture de S-CNN

L'architecture de S-CNN est composée de :

- Deux premiers blocs comprenant une convolution, une normalisation (*Batch Normalization*), une activation ReLU (Rectified Linear Unit), une couche de *max pooling* et l'utilisation du *dropout*, voir plus loin 2.4.4.
- Un troisième bloc similaire sans *max pooling*, avec l'utilisation du dropout.
- Une couche finale entièrement connectée avec une fonction *softmax*.
- Environ **120 000** paramètres.

Les dimensions finales des images étaient 112×60 pixels, permettant ainsi de réduire la mémoire nécessaire au processus d'apprentissage sans compromettre la qualité des données.

Cette architecture sera implémentée à l'aide de la bibliothèque `torch`[6].

La Figure 2.5 illustre cette architecture.

6. `torch` est la bibliothèque principale de PyTorch, dédiée à la création et à l'entraînement de réseaux de neurones, avec une interface flexible pour les opérations sur tenseurs et l'automatisation du calcul différentiel. Plus d'informations sur : `https://pytorch.org`

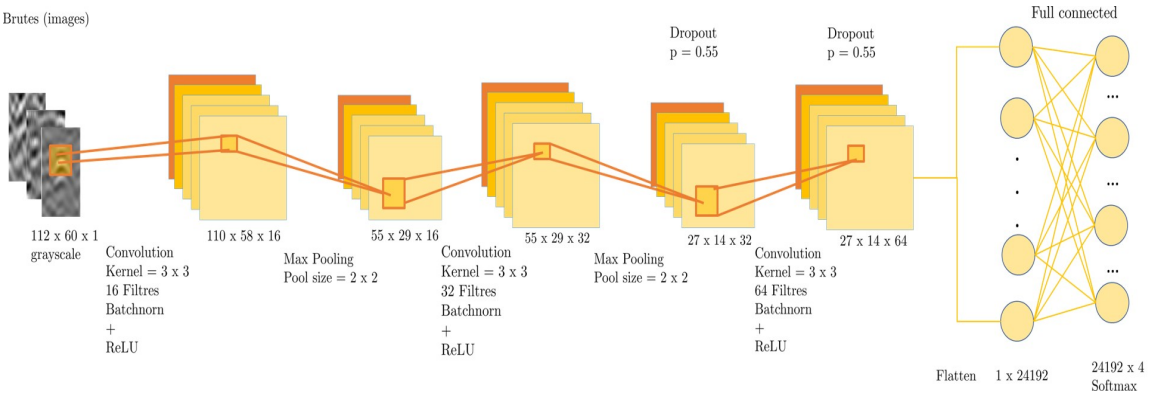


FIGURE 2.5 – Ci-dessus se trouve l'Architecture du modèle nommé CNN1 proposée dans [33] pour la classification des images GPR.

2.3.2 Avantages et inconvénients des S-CNN

Les *Shallow Convolutional Neural Networks* (S-CNN) constituent une solution efficace pour la classification d'images GPR dans des contextes à faibles ressources et à volume de données limité. Leur architecture simple présente plusieurs avantages, mais également certaines limites structurelles.

Avantages

- **Faible complexité computationnelle :** Grâce à leur profondeur réduite, les S-CNN demandent peu de mémoire et de puissance de calcul, ce qui les rend adaptés aux dispositifs embarqués souvent utilisés pour l'acquisition de données GPR.
- **Bonne performance avec peu de données :** Leur nombre limité de paramètres permet un entraînement efficace même sur des jeux de données restreints, comme c'est souvent le cas en GPR.
- **Moindre risque de surapprentissage :** La simplicité du modèle limite la spécialisation excessive, ce qui favorise une meilleure généralisation sur des ensembles bruités ou hétérogènes.

Inconvénients

- **Capacité d'expression limitée :** Leur faible profondeur empêche la modélisation de structures complexes, ce qui peut nuire à la reconnaissance de cibles enfouies mal définies ou faiblement contrastées.
- **Moins robustes au bruit :** L'absence de couches hiérarchiques profondes limite leur capacité à apprendre des représentations robustes aux perturbations fréquentes dans les données GPR.

2.3.3 Expérimentations

Comme nous pouvons le voir sur la Figure 2.6, les résultats obtenus sont globalement meilleurs pour chaque ratio de données d'entraînement avec la méthode basée sur la covariance, notamment pour SVC_gs_covpool, bien qu'ils restent proches de ceux obtenus avec le S-CNN. Toutefois, les performances du S-CNN s'avèrent insuffisantes au regard des résultats observés.

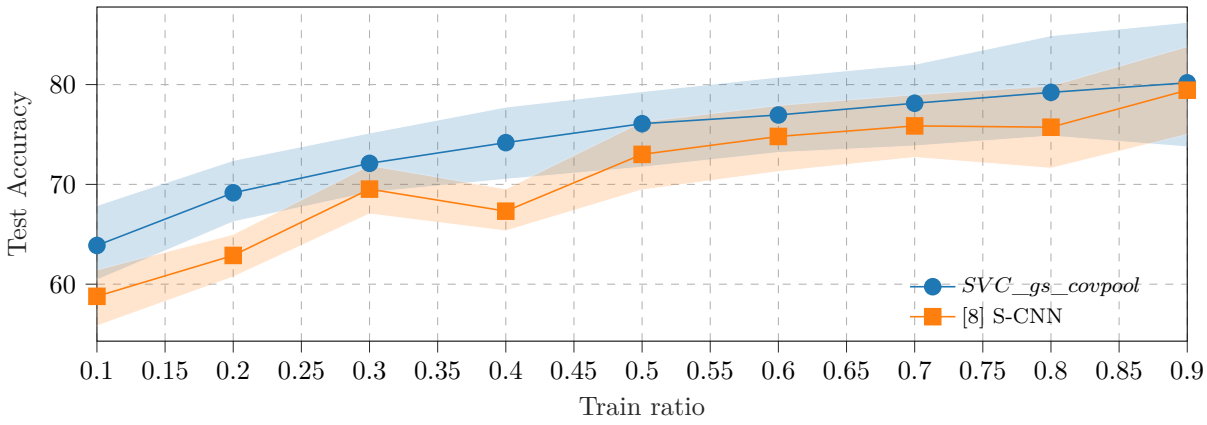


FIGURE 2.6 – Comparaison des résultats avec S-CNN et SVC_gs_covpool. Précision sur les données test par rapport au pourcentage de l'ensemble de données d'apprentissage sur 100 graines différentes. Pour chaque méthode, la ligne correspond à la moyenne de la précision sur toutes les graines, et la zone remplie correspond aux 5ème et 95ème quantiles.

Nous avons exploré l'utilisation de réseaux de neurones convolutifs peu profonds (S-CNN, pour *Shallow Convolutional Neural Network*) appliqués directement sur les images GPR, dans le but d'améliorer les résultats obtenus précédemment avec des descripteurs basés sur les matrices de covariance. Ces réseaux apprennent automatiquement des filtres de convolution afin d'extraire des motifs locaux caractéristiques, suivis de couches entièrement connectées pour effectuer la prédiction finale. Bien que cette approche permette de capturer certaines structures discriminantes dans les radargrammes, les performances obtenues restent limitées et sont sensibles aux variations expérimentales. L'utilisation de S-CNN n'a donc pas permis d'améliorer significativement la classification par rapport aux covariances.

Pour dépasser ces limitations, nous proposons désormais d'**explorer des réseaux de neurones beaucoup plus profonds**, notamment les réseaux neuronaux convolutionnels profonds (CNN), capables d'apprendre des représentations hiérarchiques plus riches et mieux adaptées à la complexité des images GPR. L'objectif est de tirer parti de cette profondeur pour améliorer la classification en capturant des structures discriminantes que les modèles peu profonds ne peuvent pas représenter efficacement.

2.4 Entraînement des CNN : From Scratch et Transfert d'apprentissage

Les réseaux neuronaux convolutionnels (*Convolutional Neural Networks*, CNN) représentent aujourd'hui l'une des architectures les plus performantes pour la classification d'images. Comme vu précédemment, leur efficacité repose sur leur capacité à extraire automatiquement des représentations hiérarchiques à partir des données brutes, apprenant successivement des caractéristiques locales (bords, textures), puis globales (formes, structures). Ces architectures ont une grande capacité à capturer des relations complexes et abstraites dans les données. Cependant, l'entraînement de ces modèles peut se heurter à des limitations, notamment à la rareté des données disponibles, ce qui est le cas dans notre étude. Dans ce travail, deux approches principales sont explorées : l'entraînement **from scratch** et le **transfert d'apprentissage**.

2.4.1 Adaptation aux images GPR

Tout d'abord, avant d'aborder les phases d'entraînement, nous devons présenter comment les modèles CNN classiques seront adaptés à nos images GPR. Comme cela a été fait en 2.3, les images GPR utilisées dans ce travail étant en niveaux de gris (grayscale), des ajustements sont nécessaires pour les adapter aux architectures CNN profondes classiques, comme **AlexNet** [45], **ResNet** [46], et **MobileNetV2** [47], conçues à l'origine pour des images RGB à trois canaux. La première couche convolutionnelle est donc modifiée afin d'accepter une entrée à un seul canal, ce qui réduit le nombre de poids dans cette couche (par exemple, les convolutions 3×3×3 deviennent 3×3×1) et modifie l'extraction des caractéristiques de bas niveau, telles que les contours et les textures.

Cette adaptation a un impact direct sur le fine-tuning du modèle (voir 2.4.3). En effet, les poids préentraînés de la première couche ne pouvant plus être utilisés, ils sont réinitialisés et doivent être réappris à partir des données GPR. Toutefois, les couches suivantes conservent leurs poids préentraînés sur ImageNet [48], ce qui permet de préserver les représentations intermédiaires et avancées, facilitant ainsi l'adaptation du modèle.

L'utilisation d'images en grayscale présente à la fois des avantages et des limites. D'un côté, la réduction du nombre de paramètres dans la première couche diminue la complexité du modèle et améliore sa robustesse face aux variations des données. De l'autre, cette simplification peut affecter la capacité à capturer certaines structures complexes initialement optimisées pour des images RGB. Néanmoins, grâce au fine-tuning, le réseau parvient progressivement à compenser cette différence et à exploiter efficacement les informations spécifiques aux données GPR. Ces architectures ont une grande capacité à capturer des relations complexes et abstraites dans les données.

2.4.2 Entraînement from scratch

L'entraînement *from scratch* consiste à initialiser un modèle avec des poids aléatoires et à l'entraîner entièrement sur des données spécifiques à la tâche. Cette méthode offre une grande flexibilité et permet de concevoir un modèle optimisé pour les caractéristiques propres aux données GPR.

Avantages de l'entraînement from scratch :

- **Adaptation optimale aux données spécifiques :** Le modèle est entièrement entraîné sur les images GPR, ce qui lui permet d'extraire des caractéristiques adaptées.
- **Flexibilité de l'architecture :** L'architecture du modèle peut être modifiée et optimisée selon les besoins spécifiques de la tâche, notamment en ajustant les dimensions d'entrée et le nombre de classes en sortie.

Limites de l'entraînement from scratch :

- **Exigences en données :** Lorsque l'on entraîne un réseau from scratch, tous les poids sont initialisés aléatoirement. Contrairement au transfert d'apprentissage, où les premières couches utilisent des caractéristiques générales déjà apprises sur de grands ensembles de données (comme ImageNet), le modèle from scratch doit apprendre à extraire des caractéristiques pertinentes directement à partir des données de la tâche. Cela nécessite beaucoup plus de données pour permettre au modèle d'identifier et de généraliser des motifs pertinents ; or, nous disposons d'un nombre limité d'exemples.
- **Coût computationnel élevé :** L'entraînement d'un modèle profond tel que ResNet est coûteux en termes de temps de calcul et de mémoire.

2.4.3 Transfert d'apprentissage

Le transfert d'apprentissage est une technique permettant d'adapter un modèle préentraîné, tel qu'un réseau entraîné sur ImageNet, à une nouvelle tâche avec un ensemble de données différent. Cette approche est particulièrement utile lorsque les données annotées sont limitées, car elle tire parti des connaissances

acquises sur un grand volume d'images pour faciliter l'apprentissage dans un domaine spécifique, comme la classification des images GPR.

Approches principales Deux stratégies principales sont utilisées pour l'adaptation d'un modèle préentraîné :

- **Feature extraction :** Les premières couches du réseau, qui capturent des caractéristiques génériques (bords, textures), sont gelées, et seules les couches finales sont ajustées à la nouvelle tâche. Cette méthode réduit les besoins en données et accélère l'entraînement.
- **Fine-tuning :** Toutes les couches du modèle préentraîné sont réajustées sur les nouvelles données. Cette approche est plus coûteuse en ressources, mais elle permet une meilleure adaptation aux spécificités des données GPR.

Avantages et inconvénients du transfert d'apprentissage Le transfert d'apprentissage présente plusieurs atouts :

- **Efficacité avec peu de données :** Il permet de réduire la quantité de données annotées nécessaire pour atteindre de bonnes performances.
- **Gain de temps :** L'utilisation d'un modèle préentraîné diminue significativement la durée de l'entraînement.
- **Adaptabilité :** Le modèle peut être rapidement ajusté à une nouvelle tâche en exploitant des caractéristiques générales préalablement apprises.

Cependant, il comporte aussi certaines limites, notamment :

- **Différence entre les images GPR et celles d'ImageNet :** Les réseaux préentraînés sur ImageNet ont appris des caractéristiques adaptées aux images naturelles en RGB, tandis que les images GPR sont en niveaux de gris et présentent des motifs très différents. Cette différence peut limiter l'efficacité du transfert d'apprentissage et nécessiter des ajustements spécifiques.
- **Transferts possibles d'informations non pertinentes :** Certaines caractéristiques apprises sur ImageNet peuvent ne pas être adaptées aux images GPR, introduisant ainsi du bruit dans les premières couches du réseau.
- **Coût du fine-tuning :** Ajuster l'ensemble des couches du réseau demande plus de données et de ressources de calcul, ce qui peut être un inconvénient si les ressources sont limitées.

Ainsi, bien que le transfert d'apprentissage soit une solution efficace pour la classification des images GPR, il doit être soigneusement ajusté pour prendre en compte les spécificités de ces données, notamment en adaptant les premières couches du modèle et en choisissant judicieusement entre l'extraction de caractéristiques (*feature extraction*) et l'ajustement fin (*fine-tuning*).

2.4.4 Modèles considérés

Entrée des réseaux : Il existe une grande variété d'architectures de réseaux de neurones convolutifs (CNN) dans la littérature, chacune présentant des compromis spécifiques en termes de complexité, de profondeur et de capacité de généralisation. Dans le cadre de nos expériences, nous avons retenu trois architectures couramment utilisées et disponibles dans la bibliothèque `torchvision`[7] : AlexNet [45], ResNet-34 [46] et MobileNetV2 [47]. Ces modèles ont été adaptés pour traiter des images en niveaux de gris et pour correspondre au nombre de classes visées dans notre tâche. L'objectif est de comparer leurs performances afin d'identifier l'architecture la plus efficace dans notre contexte d'application.

Dans la suite, nous décrivons plus en détail la structure de chacune de ces trois architectures utilisées.

7. `torchvision` est une bibliothèque Python faisant partie de l'écosystème PyTorch, qui fournit des jeux de données, des modèles préentraînés et des transformations utiles pour la vision par ordinateur. Plus d'informations sur : `https://pytorch.org/vision/`

AlexNet : AlexNet [45] est une architecture pionnière, composée des couches suivantes :

- **Couches convolutionnelles :** Cinq couches convolutionnelles avec des filtres de tailles variées, suivies de fonctions d'activation ReLU.
- **Max-pooling :** Réduction de la dimensionnalité à l'aide de couches de max-pooling.
- **Couches entièrement connectées :** Trois couches entièrement connectées, avec la dernière utilisée pour la classification.
- **Dropout** : cette technique permet de réduire le risque de sur-apprentissage, qui se manifeste lorsque le modèle apprend trop précisément les données d'entraînement, y compris les variations non significatives. Cela peut compromettre ses performances sur des données inédites. Le *dropout* consiste à désactiver aléatoirement certains neurones durant l'entraînement afin de rendre l'apprentissage moins dépendant de connexions spécifiques et ainsi favoriser une meilleure généralisation.

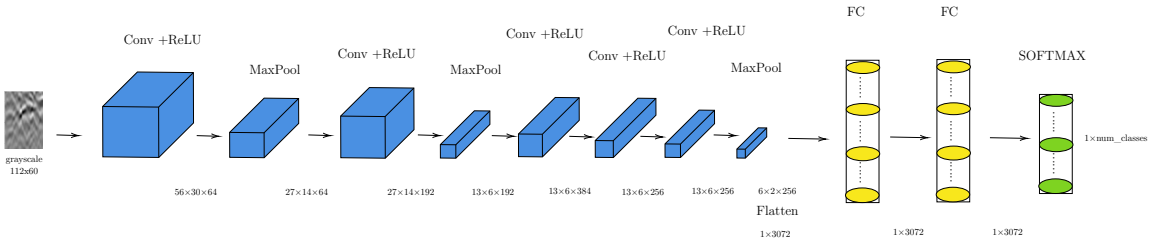


FIGURE 2.7 – Architecture AlexNet.

ResNet-34 : ResNet-34 (Residual Network) [46] est une architecture conçue pour résoudre les problèmes de dégradation des performances dans les réseaux profonds en utilisant des **connexions résiduelles** (aussi appelées *skip connections*). Cela permet de former des réseaux très profonds tout en maintenant une performance élevée en facilitant l'apprentissage des gradients.

Les principales couches de **ResNet-34** sont :

- **Blocs résiduels :** Les blocs résiduels dans ResNet-34 sont constitués de convolutions suivies de normalisation par lots (*Batch Normalization*) et d'activations *ReLU*. Un bloc résiduel prend en entrée un signal et lui applique une transformation, avant d'ajouter l'entrée au résultat de la transformation. La sortie d'un bloc résiduel est donc la somme de l'entrée et de la sortie de la transformation. Cette addition permet de maintenir un flux de gradients plus fluide pendant la rétropropagation, ce qui facilite l'apprentissage.
- **Global Average Pooling (GAP) :** Après avoir passé plusieurs blocs résiduels, un pooling global moyen est appliqué sur les cartes de caractéristiques afin de réduire la dimensionnalité avant la couche de classification. Cela permet de résumer les informations tout en conservant les caractéristiques essentielles.
- **Connexion résiduelle :** Cette connexion permet de transmettre les gradients directement entre les couches, en permettant à l'information d'être ajoutée directement à la sortie du bloc. Cela permet de réduire le problème du gradient disparu dans les réseaux profonds et d'améliorer la convergence du modèle.

Expression mathématique d'un bloc résiduel : Un **bloc résiduel** standard dans ResNet peut être formulé comme suit :

$$\text{Sortie} = F(x, \{W_i\}) + x$$

Où :

- $F(x, \{W_i\})$ est la transformation appliquée à l'entrée x, généralement une séquence de convolutions, de normalisation de lot et de fonction d'activation (ReLU).
- x est l'entrée du bloc, qui est ajoutée directement à la sortie de la transformation $F(x, \{W_i\})$.
- $\{W_i\}$ représente les poids associés à la transformation F.

Résumé de l'architecture de ResNet-34 : ResNet-34 repose sur des blocs résiduels classiques qui permettent d'ajouter les entrées aux sorties des transformations dans chaque bloc. Cela facilite le passage des gradients et permet un apprentissage plus stable et efficace. Une illustration de cette architecture est montrée dans la Figure 2.8

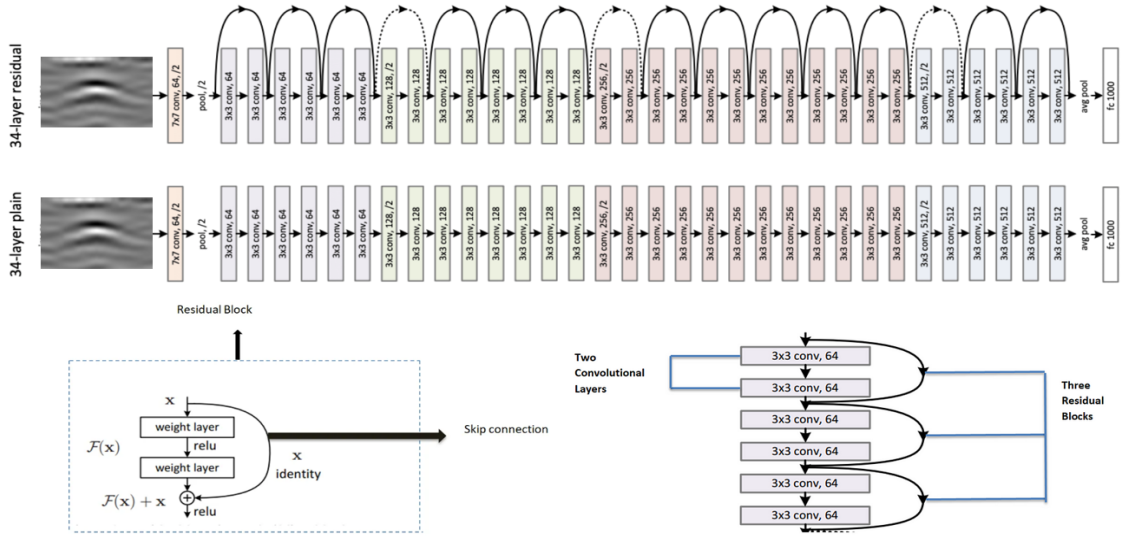


FIGURE 2.8 – Architecture de ResNet-34 avec les blocs résiduelle utilisée pour la classification des images GPR.

Utilisation des bottlenecks dans des ResNet plus profonds (ex. ResNet-50) : Dans les versions plus profondes de ResNet, telles que **ResNet-50**, l'utilisation des **blocs résiduels** et des **blocs bottleneck** (ou « goulot d'étranglement » en français) devient essentielle. Ces blocs permettent de rendre l'architecture plus efficace, tout en maintenant des performances élevées ; car ils permettent de gérer efficacement l'augmentation du nombre de canaux tout en minimisant le coût computationnel. En effet, les **bottlenecks** sont des structures où l'expansion des canaux est effectuée avant la réduction des dimensions dans les blocs résiduels. Cela permet de réduire le nombre de calculs tout en offrant une meilleure capacité de représentation.

Un **bloc résiduel bottleneck** est composé de trois étapes (cf Figure 2.9) :

- **Convolution 1×1 (expansion) :** Augmente le nombre de canaux.
- **Convolution 3×3 (depthwise) :** Effectue un filtrage des caractéristiques.
- **Convolution 1×1 (réduction) :** Réduit le nombre de canaux pour revenir à la dimension d'entrée.

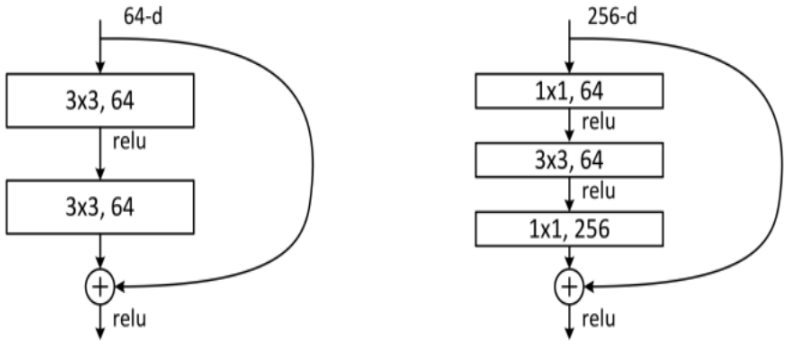


FIGURE 2.9 – Bloc résiduel standard (ResNet-18/34) à gauche comme dans la Figure 2.8 et bloc résiduel bottlenecks pour les ResNet plus profonds ResNet-50/101/152 à droite. [46]

La transformation dans un bloc résiduel avec **bottleneck** peut être formulée comme :

$$\text{Sortie} = F(x, \{W_i\}) + x$$

Où :

- $F(x, \{W_i\})$ correspond ici à la séquence de convolutions dans un **bottleneck** : expansion avec une convolution 1×1, suivie d'une convolution 3×3, puis réduction avec une autre convolution 1×1.
- x est l'entrée du bloc, qui est ajoutée à la sortie du bottleneck.

MobileNetV2 : une architecture optimisée pour l'efficacité MobileNetV2 [47] est une amélioration directe de l'architecture MobileNet V1 [49], conçue pour combiner l'efficacité computationnelle et la performance. La première version (V1) introduisait le concept de *convolutions séparables en profondeur* (*depthwise separable convolutions*), permettant de réduire considérablement le nombre d'opérations tout en maintenant une bonne capacité de représentation. Cette opération décompose la convolution standard en deux étapes successives :

- une convolution *depthwise*, appliquée indépendamment sur chaque canal d'entrée ;
- une convolution *pointwise* 1×1, qui combine les cartes de caractéristiques obtenues.

La Figure 2.10 ci-dessous illustre ces convolutions.

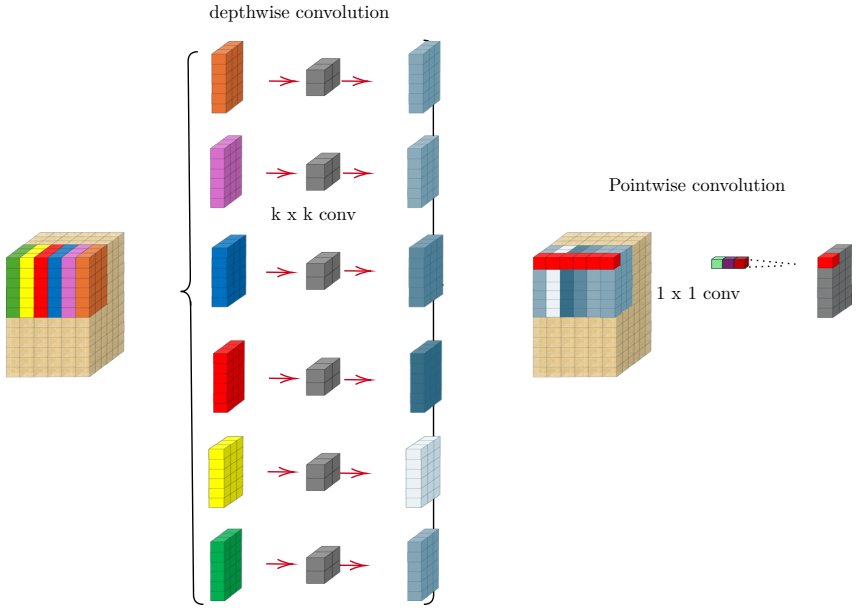


FIGURE 2.10 – Principe des convolutions séparables en profondeur introduites dans MobileNet V1 : combinaison d'une convolution *depthwise* et d'une convolution *pointwise* 1×1.

MobileNetV2 reprend ce principe fondamental et y apporte deux innovations majeures :

1. l'introduction de **blocs résiduels inversés** (*Inverted Residual Blocks*), qui facilitent la rétropropagation du gradient et améliorent la stabilité de l'apprentissage ;

2. l'utilisation de **goulets d'étranglement linéaires** (*linear bottlenecks*), qui préserve l'information tout en limitant la complexité du modèle.

Chaque bloc inversé résiduel comprend trois étapes principales :

- **Expansion** : une convolution 1×1 augmente la dimensionnalité de l'espace des caractéristiques ;
- **Convolution séparée en profondeur** : une convolution 3×3 agit sur chaque canal de manière indépendante ;
- **Projection linéaire** : une convolution 1×1 réduit la dimensionnalité à sa taille initiale, sans activation, afin d'éviter la perte d'information.

Lorsqu'un bloc conserve les mêmes dimensions d'entrée et de sortie, une connexion résiduelle (*skip connection*) est ajoutée, facilitant le passage des gradients et améliorant la généralisation. Le processus complet d'expansion et de compression linéaire au sein d'un bloc est illustré à la Figure 2.11.

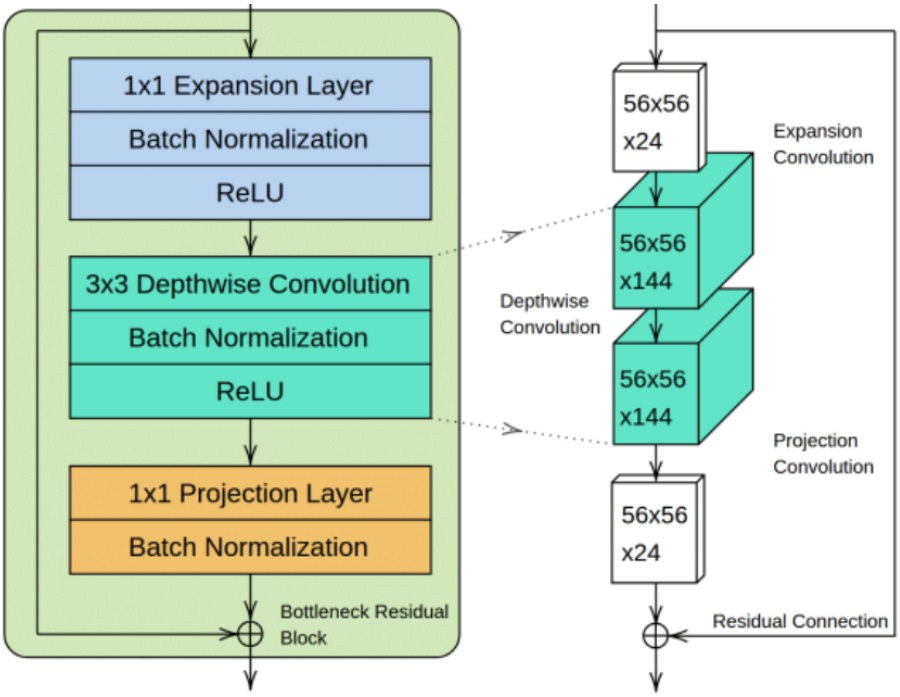


FIGURE 2.11 – MobileNetV2 avec blocs résiduels inversés. Processus pour la création de goulets d'étranglement (bottlenecks) linéaires avec l'augmentation de la carte des caractéristiques de 24 cartes à 144 cartes et la réduction de la carte des caractéristiques de 144 cartes à 24 cartes. [50]

Chaque couche est suivie d'une normalisation de lot, et la fonction d'activation utilisée est ReLU6, qui est plus stable numériquement que ReLU pour les calculs à faible précision.

Architecture complète L'architecture globale de MobileNetV2, illustrée Figure 2.12, comprend :

- une première convolution 3×3 standard ;
- une série de 17 blocs résiduels inversés ;
- une convolution 1×1 finale pour l'agrégation des caractéristiques ;
- un *pooling* global par moyenne (GAP) ;
- une couche de classification.

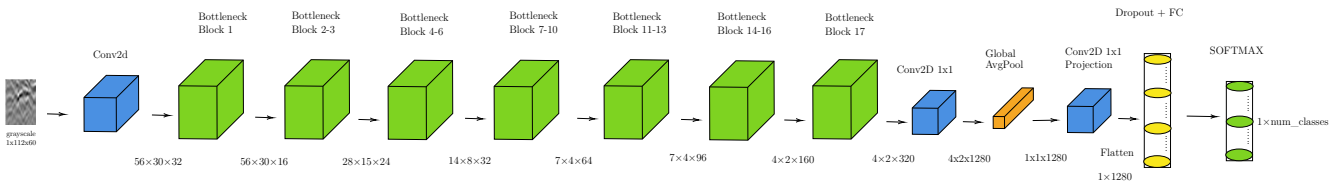


FIGURE 2.12 – Architecture complète de MobileNet V2 : succession de blocs résiduels inversés et convolutions séparables en profondeur.

Résumé En résumé, MobileNet V2 conserve la philosophie d'efficacité introduite par MobileNet V1, tout en améliorant la qualité des représentations internes grâce aux blocs inversés résiduels et à la projection linéaire. Cette combinaison permet d'obtenir un excellent compromis entre performance et coût

de calcul, rendant cette architecture particulièrement adaptée aux environnements contraints ou aux applications nécessitant des réseaux légers.

2.4.5 Résumé des modèles

Le tableau suivant compare les principales caractéristiques des modèles utilisés :

TABLE 2.1 – Comparaison des architectures AlexNet, ResNet-34 et MobileNetV2.

Modèle	Particularités	Paramètres	Adaptation à grayscale
AlexNet	Convolutions classiques + Dropout	∼60M	Modifiable facilement
ResNet-34	Connexions résiduelles + GAP	∼25M	Modifiable facilement
MobileNetV2	Blocs inversés résiduels + ReLU6	∼3.5M	Très adapté

2.4.6 Résultats des expérimentations et analyse des performances

Dans cette partie, tous les modèles sont utilisés avec les mêmes paramètres à chaque fois. Les trois architectures sont disponibles dans la bibliothèque `torchvision`[8]. L'optimiseur choisi est **SGD**(Stochastic Gradient Descent) avec un momentum de 0.9, une taille de batch de 8 et un taux d'apprentissage (learning rate) de 0.007.

Sur la Figure 2.13, on observe que les résultats sont généralement meilleurs avec le modèle ResNet-34 retrained ou from scratch (que l'on notera parfois **RRT**), que l'on dispose de peu ou de beaucoup de données d'entraînement. En comparaison avec MobileNetV2 et AlexNet, les performances sont meilleures avec MobileNetV2 lorsque le ratio des données d'entraînement est faible. En effet, pour AlexNet, les performances se dégradent particulièrement lorsque le ratio d'entraînement est inférieur à 0.3. Cependant, pour des ratios d'entraînement plus élevés, AlexNet présente des performances légèrement meilleures que MobileNetV2.

Sur la Figure 2.14, on observe également que les résultats sont globalement meilleurs avec ResNet-34 fine-tuned ou ajusté (que l'on notera parfois **RFT**), que l'on dispose de peu ou de beaucoup de données d'entraînement. Ici, MobileNetV2 donne de meilleurs résultats avec des ratios d'entraînement faibles, inférieurs à 60% des données de test. Les performances d'AlexNet sont plus mitigées et globalement moins bonnes que celles des deux autres modèles sur l'ensemble des ratios. De plus, une forte variabilité est observée selon les différentes graines utilisées, avec un modèle qui semble parfois être en sous-apprentissage (underfitting) lorsqu'il y a peu de données d'entraînement. Cela se manifeste par des résultats difficiles à expliquer et la présence de nombreuses valeurs aberrantes, nécessitant probablement une correction.

8. `torchvision` est une bibliothèque Python faisant partie de l'écosystème PyTorch, qui fournit des jeux de données, des modèles préentraînés et des transformations utiles pour la vision par ordinateur. Plus d'informations sur : `https://pytorch.org/vision/`.

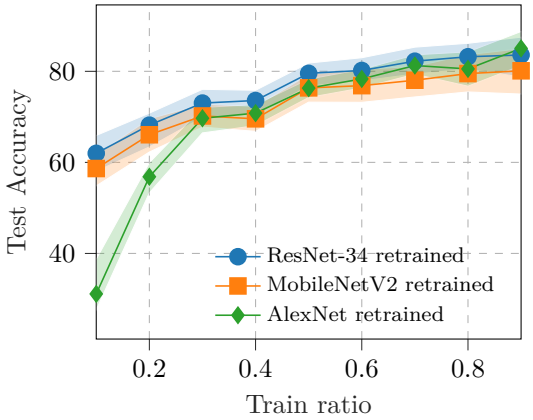


FIGURE 2.13 – Comparaison des trois CNN *retrained*. Précision sur les données test par rapport au pourcentage de l'ensemble de données d'apprentissage sur 100 graines différentes. Pour chaque méthode, la ligne correspond à la moyenne de la précision sur toutes les graines, et la zone remplie correspond aux 5ème et 95ème quantiles.

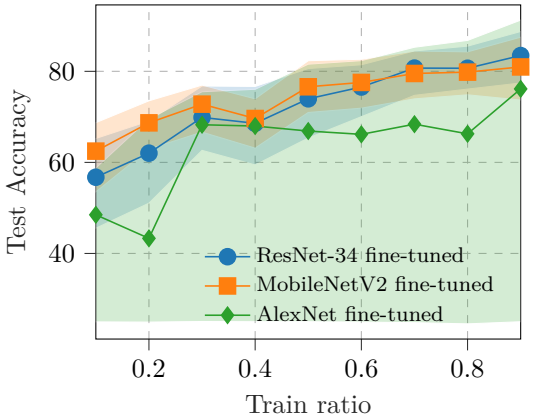


FIGURE 2.14 – Comparaison des trois CNN fine-tuned. Précision sur les données test par rapport au pourcentage de l'ensemble de données d'apprentissage sur 100 graines différentes. Pour chaque méthode, la ligne correspond à la moyenne de la précision sur toutes les graines, et la zone remplie correspond aux 5ème et 95ème quantiles.

Globalement, les expérimentations montrent que l'approche *from scratch* a permis d'obtenir de meilleures performances que le transfert d'apprentissage. Ces résultats peuvent s'expliquer par les facteurs suivants :

Analyses des résultats :

- Le modèle *from scratch* bénéficie d'une optimisation complète basée sur les données GPR, ce qui favorise une extraction de caractéristiques plus pertinentes.
- En revanche, les modèles préentraînés sur ImageNet, adaptés à des images RGB naturelles, sont moins efficaces pour des images GPR unidimensionnelles.
- Le transfert d'apprentissage est souvent limité par les caractéristiques générales apprises sur des bases de données génériques, qui ne sont pas toujours directement applicables aux spécificités des données GPR.
- L'utilisation d'images en grayscale a permis de réduire la complexité des modèles entraînés *from scratch*, les rendant plus performants dans ce contexte. Cependant, pour le transfert d'apprentissage, cette simplification a limité la capacité des premières couches à exploiter pleinement les poids initiaux.
- Les résultats obtenus avec AlexNet fine-tuned montrent une variabilité importante selon la graine d'initialisation, avec des performances allant d'un comportement quasi-aléatoire (25 %) à un apprentissage satisfaisant (92 %). Cette instabilité pourrait s'expliquer par plusieurs facteurs. D'une part, l'architecture d'AlexNet, dépourvue de mécanismes modernes de stabilisation tels que la Batch Normalization, est connue pour être sensible aux conditions d'initialisation et aux premiers mini-lots vus par le réseau. D'autre part, les filtres préentraînés sur ImageNet pourraient être mal adaptés aux caractéristiques particulières des données GPR, ce qui rend le modèle plus susceptible de converger vers des minima peu informatifs. Ainsi, la forte dépendance observée à la seed pourrait résulter d'une combinaison de ces éléments, même si d'autres facteurs liés au partitionnement aléatoire du jeu de données ou à la dynamique d'entraînement ne peuvent être entièrement exclus.

Conclusion des expérimentations : Bien que le transfert d'apprentissage soit une solution efficace pour des ensembles de données limités, l'entraînement *from scratch* a montré une meilleure capacité à s'adapter aux spécificités des images GPR et à offrir de meilleures performances de classification. Parmi nos trois CNN, ResNet-34 entraîné *from scratch* a montré les meilleures performances.

Comparaison avec le modèle basé sur les covariances : La Figure 2.15 ci-dessous montre que, pour de faibles ratios de données d'entraînement, le modèle SVC_gs_covpool obtient de meilleurs résultats que nos deux modèles ResNet-34, qu'il soit entraîné from scratch (retrained) ou fine-tuned.

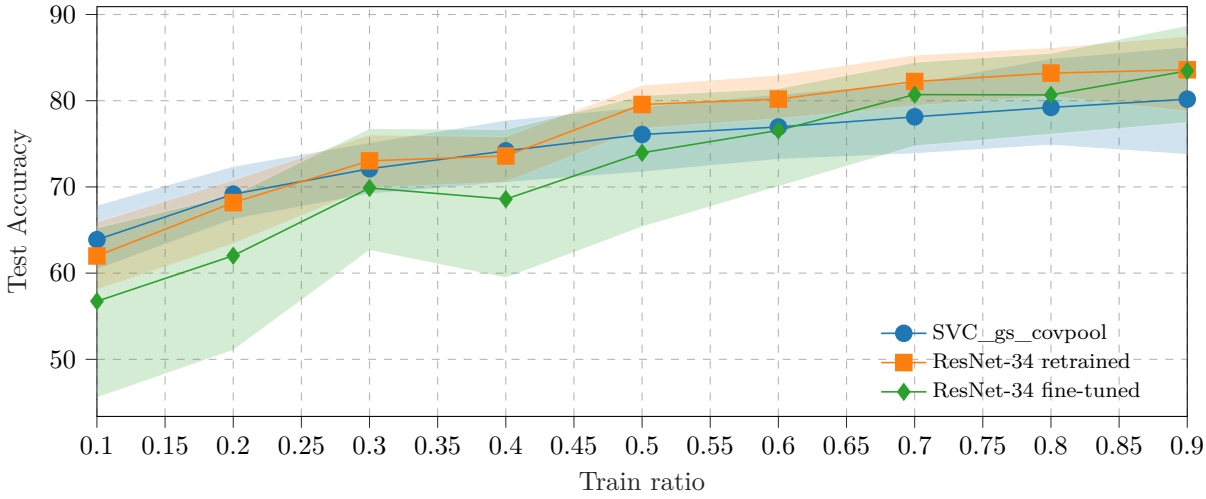


FIGURE 2.15 – Comparaison des performances de ResNet-34 dans les deux configurations et de SVC_gs_covpool. La précision sur les données de test est représentée en fonction du pourcentage des données d'entraînement, calculée sur 100 graines différentes. Pour chaque méthode, la ligne indique la moyenne des précisions sur toutes les graines, tandis que la zone remplie correspond aux 5ème et 95ème quantiles.

Bien que les modèles basés sur les covariances aient montré de meilleures performances que l'ACP ou les S-CNN, leurs performances ne surpassent pas celles du ResNet-34 from scratch (retrained) lorsque la quantité de données d'entraînement augmente. En effet, comme illustré sur la Figure 2.15, au-delà de 30% des données d'entraînement, le ResNet-34 obtient de meilleurs résultats. En revanche, dans un scénario où la quantité de données d'entraînement est plus faible, notre modèle basé sur les covariances semble mieux s'adapter que le ResNet-34 fine-tuned. **Il est donc pertinent de développer un modèle de réseau de neurones profond basé sur les matrices de covariance, capable de maintenir de bonnes performances, même avec un plus grand volume de données d'entraînement.**

Choix de L Pour le choix du nombre de couches L, nous avons comparé le modèle SVC_gs_covpool lorsque nous avons un ratio des données d'entraînement de 70%[9] en utilisant les trois CNN pour la construction de la covariance. Les résultats se trouvent dans la Figure 2.16

9. Plusieurs ratios ont été évalués. Les résultats étant similaires, nous retenons et présentons ceux correspondant à un ratio d'entraînement de 70 %

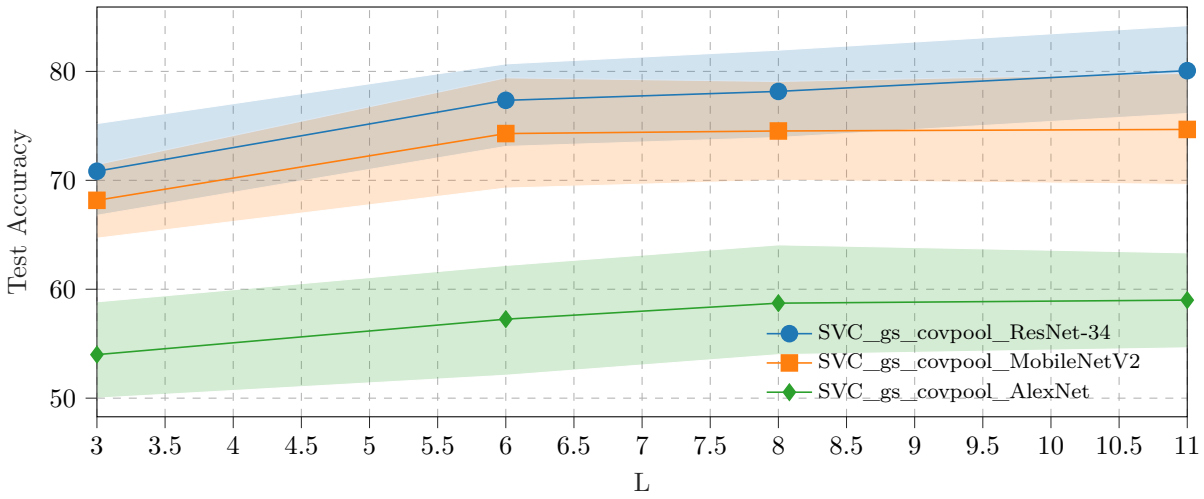


FIGURE 2.16 – Comparaison des performances de SVC_gs_covpool avec les backbones CNN pour le choix de L. La précision sur les données de test est représentée en fonction du pourcentage des données d'entraînement, calculée sur 100 graines différentes. Pour chaque méthode, la ligne indique la moyenne des précisions sur toutes les graines, tandis que la zone remplie correspond aux 5ème et 95ème quantiles.

Le choix s'était donc finalement porté sur ResNet-34, qui obtient de meilleurs résultats ici que MobileNetV2 et AlexNet. Au vu de ces résultats, nous avons fait le choix de $L = 8$, notamment parce que nous obtenons de bonnes performances comparables à celles obtenues lorsque $L = 11$ avec chacun des trois CNN. De plus, avec $L = 8$, l'entraînement des modèles prend moins de temps qu'avec $L = 11$.

2.5 Vers un modèle robuste pour les données GPR

Dans les sections précédentes, nous avons exploré différentes approches pour la classification des images GPR, allant des modèles profonds classiques, capables de traiter de grands ensembles de données, aux matrices de covariance, particulièrement efficaces lorsque les données sont limitées. Cependant, les données GPR présentent des caractéristiques spécifiques et des variabilités (type de sol, élévation, fréquence, etc.) qui posent des défis supplémentaires. Dans cette section, nous cherchons à développer un modèle qui soit non seulement performant dans différents contextes, mais également robuste face aux invariances et aux incertitudes des données GPR.

2.5.1 Robustesses face à un faible volume de données

Un des principaux défis des données GPR est le faible volume de données disponibles pour l'entraînement. En effet, les acquisitions de GPR nécessitent des équipements spécialisés et sont coûteuses à réaliser, ce qui limite la quantité d'exemples annotés exploitables. Plutôt que d'utiliser des techniques d'augmentation de données ou de transfert d'apprentissage classiques, nous nous orientons vers une approche basée sur les matrices de covariance pour capturer des informations plus robustes et exploitables, même avec un faible nombre d'échantillons.

Dans notre future approche, les caractéristiques extraites par un CNN seront transformées en matrices de covariance, permettant ainsi d'obtenir une représentation plus stable face aux variations intra-classe et au bruit. Cette transformation aidera à compenser le manque de données en exploitant les relations structurelles entre les features du réseau profond.

2.5.2 Gestion des labels imprécis ou erronés

L'étiquetage des données GPR repose souvent sur l'expertise humaine, ce qui peut entraîner des erreurs ou des incohérences entre annotateurs. Ces imprécisions peuvent nuire à la performance des modèles supervisés traditionnels. Pour répondre à ce problème, notre approche future intégrera une représentation par covariance des caractéristiques extraites d'un CNN, ce qui permettra d'atténuer la dépendance aux labels bruyants en exploitant des statistiques globales plutôt que des informations locales sensibles aux erreurs d'annotation.

En utilisant ces matrices de covariance, nous cherchons à renforcer la robustesse du modèle face aux variations des annotations, tout en capturant des propriétés invariantes qui facilitent la distinction entre différentes classes, indépendamment d'éventuelles erreurs de labellisation.

2.5.3 Robustesse face aux variations expérimentales

Les conditions d'acquisition des données GPR varient selon plusieurs paramètres, tels que la nature du sol, l'élévation de l'antenne et la fréquence d'acquisition. Ces variations peuvent affecter la généralisation des modèles standards qui apprennent directement à partir des images brutes. Pour pallier ce problème, notre approche reposera sur l'exploitation des matrices de covariance des features extraites d'un CNN, qui permettent de capturer des relations structurelles indépendantes des conditions expérimentales spécifiques.

En structurant les données sous forme de matrices symétriques définies positives (SPD), nous facilitons leur exploitation dans des espaces riemannien où les relations géométriques restent invariantes face à certaines transformations du signal GPR. Cela offrira une meilleure robustesse face aux changements des conditions d'acquisition, garantissant ainsi une meilleure adaptabilité du modèle.

2.6 Conclusion

Ce chapitre a permis d'explorer différentes stratégies de classification supervisée pour l'analyse des hyperboles présentes dans les images GPR. En s'appuyant sur la base de données structurée et annotée construite au chapitre précédent, nous avons comparé plusieurs familles de méthodes, allant des approches classiques à des modèles plus avancés issus de l'apprentissage profond.

Nous avons tout d'abord évalué des méthodes classiques reposant sur des descripteurs fondés sur une réduction de dimension par ACP, combinés à des classifieurs tels que les machines à vecteurs de support (SVM) ou les forêts aléatoires (RF). Bien que ces modèles soient rapides à entraîner et relativement interprétables, leurs performances demeurent limitées, principalement en raison de leur dépendance à une représentation manuelle des données.

Nous avons ensuite introduit une approche plus riche consistant à extraire des représentations à l'aide d'un réseau de neurones convolutif (CNN), puis à appliquer un *covariance pooling* sur ces activations afin de capturer des relations de second ordre, avant de procéder à une classification à l'aide d'un SVM ou d'une RF. Cette combinaison a permis d'améliorer les performances tout en intégrant des informations plus structurées.

Nous avons également expérimenté une architecture convolutive peu profonde, S-CNN, conçue spécifiquement pour un contexte de faible volume de données. Bien qu'intéressante, cette architecture reste inférieure aux modèles profonds lorsqu'un entraînement efficace est possible.

L'étude s'est poursuivie avec des architectures plus complexes, telles qu'AlexNet [45], ResNet-34 [46] et MobileNetV2 [47], testées à la fois from scratch et en transfert learning (préentraînement sur ImageNet). L'analyse des résultats montre que :

- ResNet-34 entraîné *from scratch* offre les meilleures performances globales sur notre jeu de données.
- Le transfert d'apprentissage n'apporte pas de gain significatif et peut même dégrader les résultats dans notre cas. En effet, les caractéristiques apprises sur des images naturelles (ImageNet) ne sont pas directement exploitables pour des images radar, dont les structures diffèrent fortement.
- Les modèles entraînés *from scratch* apprennent des représentations plus pertinentes, spécifiquement adaptées à la nature des signaux GPR, tout en évitant le biais induit par une tâche d'apprentissage source non adaptée.

Enfin, cette première évaluation souligne les limites des approches classiques, notamment en termes de robustesse aux variations expérimentales (type de sol, élévation, fréquence), à la présence de bruit ou de labels incertains. Ces constats motivent l'étude approfondie d'architectures plus robustes face aux changements de distribution, intégrant des représentations statistiques de second ordre et opérant directement dans des espaces géométriques adaptés.

C'est l'objectif du chapitre 3, qui s'appuie sur les sorties convolutives issues de ResNet pour construire des matrices de covariance et les exploiter au sein d'un réseau de type SPDNet, opérant dans l'espace riemannien des matrices définies positives. Cette nouvelle approche permettra d'améliorer la généralisation des modèles, surtout lorsque les données sont peu nombreuses ou très hétérogènes.

Chapitre 3

CNN-SPDNet

Dans ce chapitre, nous présentons un cadre général de classification pour les données GPR, basé sur l'extraction de caractéristiques par un réseau convolutionnel classique (CNN) et leur traitement via un réseau adapté aux matrices symétriques définies positives (SPD), le ***SPDNet****, que nous appelons* ***CNN-SPDNet****. Les caractéristiques extraites par un* ***ResNet-34*** *entraîné from scratch sont transformées en matrices de covariance, qui sont ensuite traitées par SPDNet pour la classification. Deux architectures spécifiques* ***SRCNet*** *et* ***RCNet****. La propagation avant (forward) et la rétropropagation spécifique (backward) de SPDNet sont détaillées en s'appuyant sur les formulations avancées issues de [12] et [15].*

Sommaire

3.1 Modèles proposés

L'utilisation des représentations par matrice de covariance a démontré son efficacité dans le traitement des signaux bruités issus des systèmes radar [51], ainsi que dans le chapitre 2. En particulier, les tests d'hypothèses statistiques sur la modélisation d'ordre deux ont été appliqués avec succès à la détection de cibles en GPR [1]. Pour la classification, l'approche de [52] et [53] a exploré l'intégration du pooling de covariance avec un ResNet, ce qui améliore la robustesse aux décalages des données tout en capturant des invariants spatiaux.

Dans le cadre de CNN-SPDNet, nous proposons ici une nouvelle approche qui affine l'intégration des couches convolutionnelles en rétropropageant le gradient de la perte sur les couches de covariance [54]. Cette approche améliore l'adaptation des couches aux données GPR, qui sont fondamentalement différentes des images de vision par ordinateur.

Pour estimer les matrices de covariance à partir des caractéristiques extraites par le CNN, deux architectures spécifiques sont explorées : **SRCNet** (Stacked Residual Covariance Network) et **RCNet** (Residual Covariance Network) [17]. Ces pipelines sont illustrés dans la Figure 3.1.

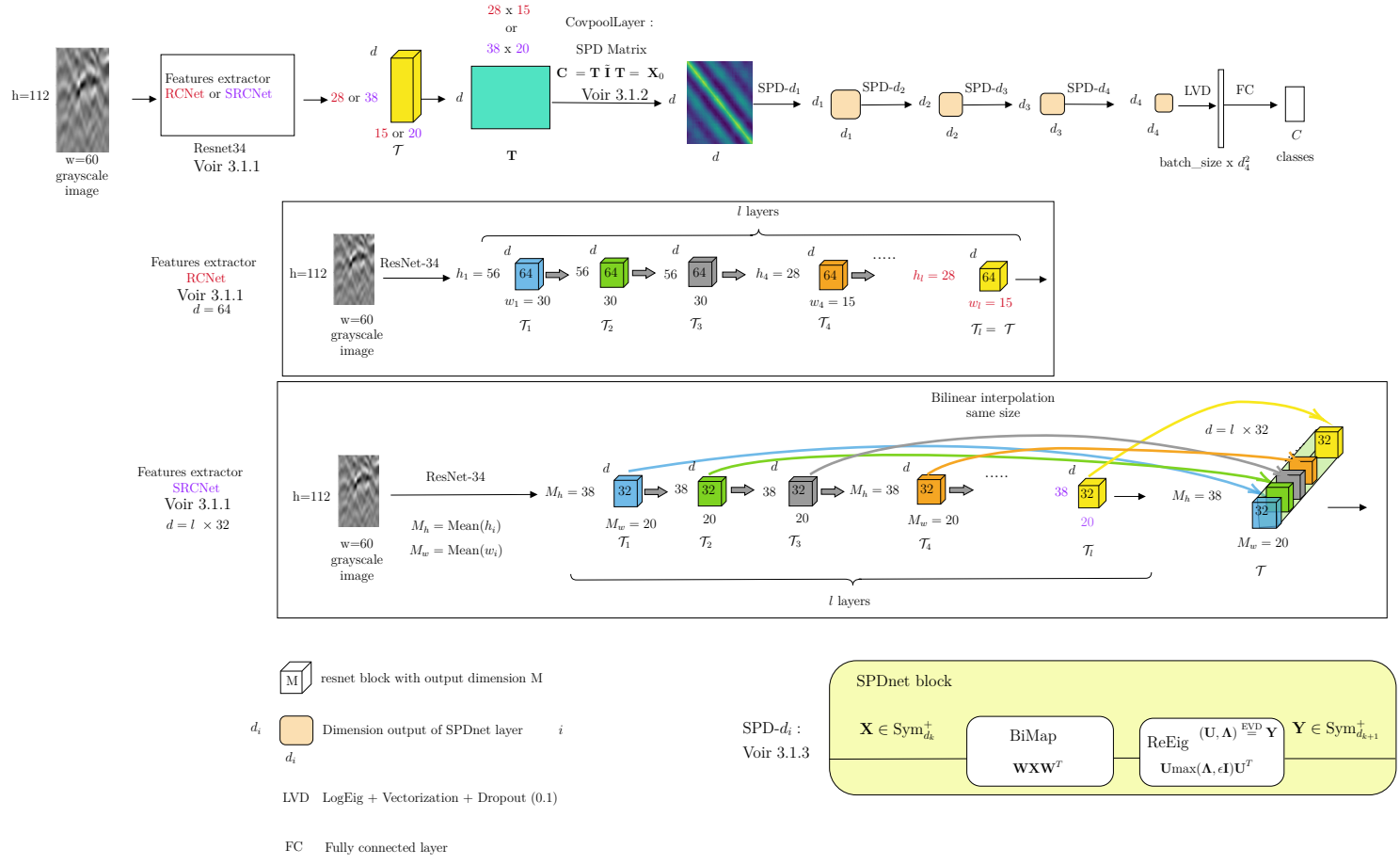


FIGURE 3.1 – Illustration des deux architectures utilisées pour estimer les matrices de covariance. Dans **RCNet**, seule la dernière sortie des blocs ResNet est utilisée, tandis que dans **SRCNet**, les 32 premières sorties sont empilées (pour économiser de l'espace mémoire) et interpolées à une taille commune de 38 × 20. Avec **RCNet**, les dimensions des trois premières couches sont $h_i = 54$ et $w_i = 30$, et $h_i = 28$, $w_i = 15$ pour les autres.

Ces architectures exploitent le pooling de covariance sur les caractéristiques extraites par le CNN et appliquent des couches SPDNet adaptées au traitement des matrices SPD [12]. L'objectif est de réduire la dimensionnalité tout en conservant l'information pertinente pour la classification, en tirant parti à la fois des représentations locales extraites par le CNN et des corrélations intra-couches capturées par les matrices de covariance.

3.1.1 Extraction des caractéristiques

Ici, nous reprenons exactement ce qui a été présenté au chapitre 2. Les étapes de construction sont les suivantes :

ResNet-34 avec l couches : Tout d'abord, nous utilisons les l premières couches d'un modèle ResNet-34 entraîné from scratch [1]. Dans notre cas, nous utilisons $l = 8$, chaque couche produisant 64 filtres de sortie. Ce choix résulte d'un compromis entre expressivité et généralisation : les couches initiales du réseau permettent de capturer des caractéristiques locales de bas niveau (bords, motifs simples), pertinentes pour les images GPR, tandis qu'un empilement plus profond risquerait d'encoder des informations trop spécifiques, voire inadaptées, surtout dans un contexte de données limitées et hétérogènes. Des expérimentations exploratoires ont d'ailleurs montré qu'au-delà de **huit couches**, la performance tend à stagner, voire à se dégrader légèrement, probablement en raison d'une perte d'informations spatiales ou d'un surapprentissage. Nous commençons avec une image en niveaux de gris $\mathbf{I} \in \mathbb{R}^{h \times w}$, dont les dimensions initiales $h = 112$ et $w = 60$ sont tirées du S-CNN de [8], afin de fournir une base commune pour la comparaison dans la suite de cet article. Nous avons considéré deux scénarios (**SRCNet** et **RCNet**, voir Figure 3.2), dans lesquels nous retenons une portion de la sortie de chaque couche ou uniquement la dernière.

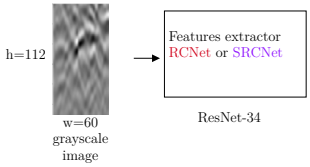


FIGURE 3.2 – Extraction de feature pour chacune des images en entrée d'un ResNet-34.

SRCNet : Partant de notre image en niveaux de gris, les étapes sont les suivantes :

- Les l premières couches du modèle ResNet-34 contiennent chacune $\bar{d} = 64$ filtres.
- Seuls $\bar{d} = 32$ filtres seront retenus pour chaque couche afin d'économiser de l'espace mémoire. Par conséquent, le nombre total de sorties de filtres est donné par $d = l \cdot \bar{d}$.
- Soit $\mathbf{I}_i \in \mathbb{R}^{h_i \times w_i}$ [2] avec $i \in [1, d]$ représentant l'ensemble des sorties de filtres.
- Définir M_h comme la moyenne de h_i et M_w comme la moyenne de w_i.
- Nous devons redimensionner les sorties des filtres $\mathbf{I}_i$ avant de les empiler. À cette fin, nous redimensionnons $\mathbf{I}_i$ en $\tilde{\mathbf{I}}_i \in \mathbb{R}^{M_h \times M_w}$.
- Nous empilons ensuite les sorties des filtres redimensionnés $\tilde{\mathbf{I}}_i$ dans un tenseur $\mathcal{T}$. Ainsi, $\mathcal{T} = \{\tilde{\mathbf{I}}_i\}_{i \in [1,d]} \in \mathbb{R}^{d \times M_h \times M_w}$.
- Enfin, nous redimensionnons $\mathcal{T}$ en $\mathbf{T}$ où $\mathbf{T} \in \mathbb{R}^{d \times M}$ avec $M = M_h \times M_w$.

1. Les résultats préliminaires du chapitre 2 ont montré que la performance est meilleure avec un réseau entraîné from scratch qu'avec un réseau appris sur des données de vision par ordinateur, probablement parce que les premières couches contiennent généralement des caractéristiques simples.

2. Notons que chaque image a une taille différente pour chaque i, ce qui explique l'étape suivante pour redimensionner toutes les images à une taille identique,

Les étapes de SRCNet sont illustrées dans la Figure 3.3 ci-dessous.

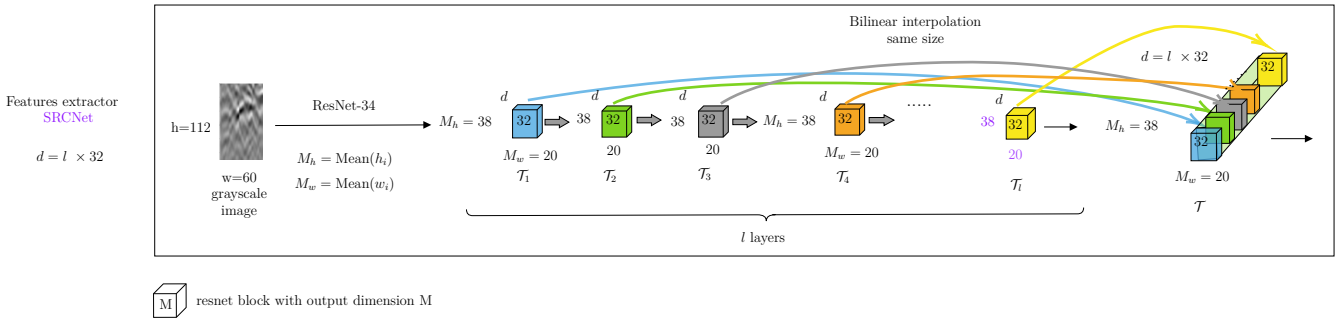


FIGURE 3.3 – **SRCNet**.

RCNet : Dans cette approche alternative et différente de celle du chapitre précédent, la covariance sera calculée sans empiler les sorties des filtres, mais en prenant les couches séquentiellement avec toutes les sorties de filtres (et donc pas seulement 32 dans cette configuration). De cette manière, la matrice de covariance sera estimée en utilisant uniquement les caractéristiques de la dernière couche l. Dans ce cas, le nombre total de filtres est $d = 64$, ce qui conduit à $\mathcal{T} = \{\tilde{\mathbf{I}}_i\}_{i\in[1,d]} \in \mathbb{R}^{d\times h_l \times w_l}$. Ensuite, nous transformons $\mathcal{T}$ en $\mathbf{T}$ où $\mathbf{T} \in \mathbb{R}^{d\times M}$ avec $M = h_l \times w_l$. La stratégie de cette seconde option vise à explorer les avantages potentiels des connexions résiduelles et à optimiser les performances en termes de temps de calcul et d'espace mémoire. Les étapes de RCNet sont illustrées dans la Figure 3.4 ci-dessous.

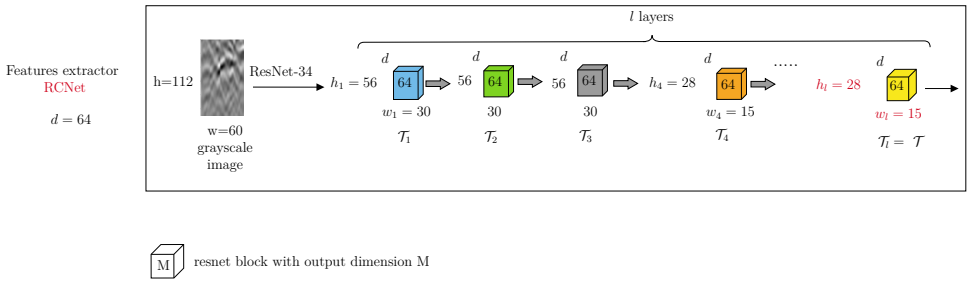


FIGURE 3.4 – **RCNet**.

Après ces étapes pour construire les données du tenseur $\mathcal{T}$ et sa matrice redimensionnée correspondante à $\mathbf{T}$ par **SRCNet** ou **RCNet**, nous avons les étapes suivantes :

3.1.2 Pooling de covariance

Ensuite, une couche de covariance pooling est ajoutée. À partir de $\mathbf{T}$ de **SRCNet** ou **RCNet**, nous calculons [3] :

$$\mathbf{C} = \mathbf{T}\bar{\mathbf{I}}\mathbf{T}^{\mathrm{T}},$$

où $\bar{\mathbf{I}} = \frac{1}{N}\left(\mathbf{I} - \mathbf{1}_N\mathbf{1}_N^{\mathrm{T}}\right)$ et $\mathbf{C} \in \mathbb{R}^{d\times d}$ est une matrice SPD (Symmetric Positive Definite).

3. En réalité, cette matrice de covariance est la matrice de covariance classique de l'échantillon (SCM), qui suppose que la distribution des données est gaussienne. On pourrait également utiliser une estimation de Ledoit-Wolf [55] si nous avons un problème de singularité, par exemple, en ce qui concerne la taille de la matrice de covariance ou le nombre d'échantillons spatiaux

Dans les cas de **SRCNet** et **RCNet**, la matrice de covariance obtenue **C** peut avoir une dimension très élevée, et il semble intéressant de réduire l'espace des données afin d'améliorer les performances. Par conséquent, nous suggérons d'ajouter des couches convolutionnelles adaptées aux matrices de covariance en utilisant le cadre proposé dans [12], désigné sous le nom de SPDNet.

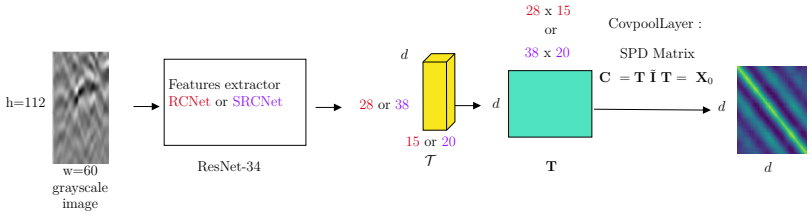


FIGURE 3.5 – Détails de l'extraction de feature pour chacune des images en entrée d'un ResNet-34, avec l'estimation de la matrice de covariance.

3.1.3 Couche SPDNet

SPDNet [12] est un modèle qui, pour $k \geq 1$, prend en entrée une matrice $\boldsymbol{X}_{k-1} \in \mathrm{Sym}^{+}_{d_{k-1}}$. Dans notre cas, nous commençons avec $\mathbf{X}_0 = \mathbf{C} \in \mathbb{R}^{d \times d}$, une matrice symétrique définie positive de taille d_{k-1}. La dimension de l'espace est réduite par plusieurs couches de convolution **BiMap** (Bilinear Mapping) et une couche de régularisation **ReEig** (Rectification of Eigenvalues) basée sur une décomposition en valeurs propres (EVD). Enfin, une couche **LogEig** (Logarithm of Eigenvalues) est utilisée pour effectuer des mesures entre les matrices de covariance dans un espace commun (plan tangent de la variété riemannienne des matrices SPD prises à l'identité). Les inconnues dans ce problème sont les matrices de convolution $\boldsymbol{W}_k \in \mathbb{R}_{*}^{d_k \times d_{k-1}}$, qui sont de faible rang et appartiennent à une variété de Stiefel.

Voici quelques détails sur les différentes couches à chaque étape de la propagation en avant :

- **Couche BiMap** (pour générer des matrices SPD plus compactes et discriminantes) : Cette couche applique une projection bilinéaire

$$\mathbf{X}_k = f_b^{(k)}(\mathbf{X}_{k-1}; \mathbf{W}_k) = \mathbf{W}_k \mathbf{X}_{k-1} \mathbf{W}_k^{\top},$$

  où $\mathbf{X}_{k-1}$ est la matrice SPD d'entrée de la k-ème couche, $\mathbf{W}_k \in \mathbb{R}_{*}^{d_k \times d_{k-1}}$ (avec $d_k < d_{k-1}$) est une matrice de transformation orthonormée, et $\mathbf{X}_k \in \mathbb{R}^{d_k \times d_k}$ est la matrice de sortie. Elle joue un rôle similaire à celui d'une couche de convolution linéaire dans les réseaux classiques : en projetant les représentations dans un espace de plus faible dimension, elle extrait les composantes les plus discriminantes tout en respectant la structure géométrique des matrices SPD (symétrie, positivité définie). Cette opération permet ainsi de construire des représentations hiérarchiques dans l'espace riemannien, analogues à celles apprises dans les CNN euclidiens.

- **Couche ReEig** (pour améliorer la performance discriminative, inspirée par ReLU) : Elle introduit une non-linéarité dans l'espace des matrices SPD en tronquant les petites valeurs propres, ce qui permet de conserver les directions spectrales informatives. Cela évite que le réseau reste trop peu profond (shallow), car sans non-linéarités, l'empilement de couches SPD serait équivalent à une simple transformation linéaire globale, donc peu expressif.

$$\mathbf{X}_k = f_r^{(k)}(\mathbf{X}_{k-1}) = \mathbf{U}_{k-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{k-1}) \mathbf{U}_{k-1}^{T},$$

  où la décomposition en valeurs propres (EIG) de $\boldsymbol{X}_{k-1} = \mathbf{U}_{k-1} \mathbf{\Sigma}_{k-1} \mathbf{U}_{k-1}^{T}$, ϵ est un seuil de rectification, $\mathbf{I}$ est une matrice identité et $\max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{k-1})$ est une matrice diagonale des valeurs propres corrigées afin de rester sur la variété SPD. Avant les étapes finales, les couches BiMap et ReEig pourraient être répétées plusieurs fois afin de trouver la meilleure représentation pour la classification.

- **Couche LogEig** (pour retourner dans l'espace euclidien)
  Dans la dernière étape du modèle, nous visons à classifier une matrice SPD discriminante de dimension inférieure. Pour permettre l'application de couches entièrement connectées (FC) traditionnelles, nous transformons d'abord la matrice SPD en une caractéristique dans l'espace euclidien. Cette transformation est effectuée à l'aide de l'opérateur LogEig, qui calcule le logarithme matriciel de l'entrée :
  $$\mathbf{X}_k = f_l^{(k)}(\mathbf{X}_{k-1}) = \mathbf{U}_{k-1} \log(\mathbf{\Sigma}_{k-1}) \mathbf{U}_{k-1}^T,$$
  où $\log(\mathbf{\Sigma}_{k-1})$ est la matrice diagonale des logarithmes des valeurs propres de la matrice SPD, issue de la décomposition en valeurs propres de $\mathbf{X}_{k-1}$ comme dans la couche ReEig.

Enfin, nous vectorisons la matrice résultante avant les couches FC et nous introduisons un mécanisme de **dropout** [56] pour atténuer le risque de surapprentissage.

La Figure 3.6 illustre le pipeline de traitement appliqué à chaque imagette extraite d'un ResNet-34, via une série de blocs SPDNet. Cette modélisation est essentielle car les couches SPDNet opèrent dans l'espace riemannien des matrices symétriques définies positives (SPD), avec des transformations spécifiques (BiMap, ReEig, LogEig) permettant de conserver leur structure et de bénéficier de leurs propriétés géométriques.

Contrairement aux couches classiques de réseaux de neurones, il ne suffit pas d'empiler des transformations linéaires : les opérations doivent être conçues pour rester différentiables et compatibles avec une optimisation de bout en bout. C'est pourquoi chaque transformation est écrite de manière explicite en formulation matricielle.

- La couche **BiMap** permet de réduire la dimension tout en maintenant la structure SPD, jouant un rôle analogue à une convolution projetée dans un espace de plus faible dimension.
- La couche **ReEig** introduit une non-linéarité spectrale (analogue à ReLU), évitant que le réseau ne soit trop peu profond. Empiler des couches linéaires serait inutile sans cela.
- La couche **LogEig** permet de ramener les données dans un espace euclidien vectoriel, facilitant leur exploitation par des couches denses classiques.

Ce design assure la différentiabilité du modèle de bout en bout et permet de l'entraîner efficacement par rétropropagation, tout en exploitant la richesse géométrique des données SPD.

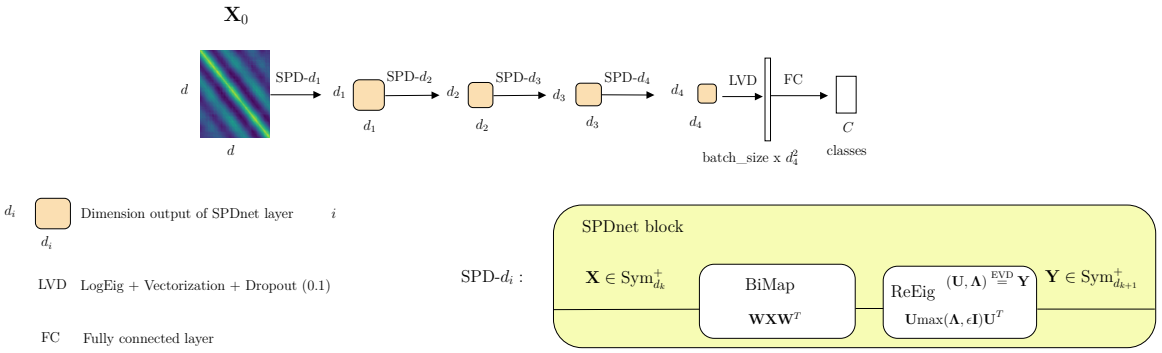


FIGURE 3.6 – Extraction des features pour chacune des imagettes en entrée d'un ResNet-34, via une séquence de blocs SPDNet (BiMap, ReEig, LogEig). Chaque opération est définie en formulation matricielle pour garantir la différentiabilité du modèle dans l'espace des matrices SPD.

3.2 Étapes de rétropropagation

La rétropropagation est effectuée sur l'ensemble des blocs définis en section 3.1, en particulier sur les couches BiMap, ReEig et LogEig du SPDNet. Contrairement aux réseaux classiques, où la rétropro-

pagation repose sur des opérations élémentaires différentiables dans un espace euclidien, ici les étapes principales sont des opérations matricielles non triviales. Elles exigent un traitement spécifique, notamment pour les couches fondées sur la décomposition en valeurs singulières (SVD), comme c'est le cas pour les couches ReEig et LogEig.

Le réseau peut être vu comme une composition de fonctions :

$$\mathcal{L} = \mathcal{L}^\ell \circ f_\ell \circ f_{\ell-1} \circ \cdots \circ f_1,$$

où chaque couche f_i transforme une entrée $\mathbf{X}_{i-1}$ en une sortie $\mathbf{X}_i$, éventuellement à l'aide de paramètres appris $\boldsymbol{\theta}_i$. En particulier, pour les couches BiMap, ces paramètres sont notés $\boldsymbol{\theta}_i = \mathbf{W}_i$, tandis que pour ReEig et LogEig, on a $\boldsymbol{\theta}_i = \emptyset$.

La rétropropagation consiste à calculer les gradients $\partial\mathcal{L}/\partial\boldsymbol{\theta}_i$ à partir de la dérivée finale $\partial\mathcal{L}/\partial\hat{y}$, en suivant les règles de la chaîne pour chaque fonction f_i. Les gradients des opérations SVD ont d'abord été dérivés dans [57], mais une version plus stable et mieux adaptée à l'apprentissage profond a été proposée dans [15], et sera adoptée ici.

La figure 3.7 illustre l'ensemble des étapes de propagation avant et arrière dans ce réseau, en précisant le cheminement des dérivées et des paramètres.

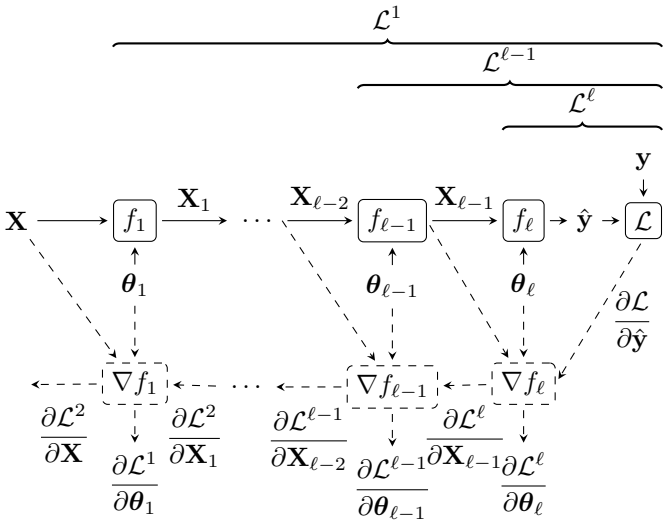


FIGURE 3.7 – Principe de la rétropropagation des matrices. Les lignes pleines représentent la propagation avant ; les lignes en pointillés, la propagation arrière des gradients. On note $\boldsymbol{\theta}_i = \mathbf{W}_i$ pour les couches BiMap, et $\boldsymbol{\theta}_i = \emptyset$ pour ReEig et LogEig. $\mathbf{X}$ est le tenseur en sortie du ResNet, et $\hat{y}$ est la sortie prédite. Chaque fonction f_i est différentiable, permettant l'optimisation du réseau de bout en bout.

Propagation avant (lignes pleines)

- On part du tenseur $\mathbf{X}$, qui est l'entrée du modèle après certaines couches (par exemple, ResNet).
- À chaque couche ℓ, une transformation est appliquée par une fonction f_ℓ paramétrée par θ_ℓ.
- Cette transformation produit une sortie intermédiaire $\mathbf{X}_\ell$, qui devient l'entrée de la couche suivante.
- Ce processus continue jusqu'à la couche finale, qui produit une sortie $\hat{\mathbf{y}}$ correspondant à une estimation de la classe.

Propagation arrière (lignes pointillées)

- La rétropropagation permet de calculer les gradients pour ajuster les paramètres θ_ℓ.

- L'erreur $\frac{\partial \mathcal{L}}{\partial \mathbf{y}}$ (où $\mathcal{L}$ est la fonction de coût) est propagée en arrière.
- Les dérivées successives sont calculées selon la règle de la chaîne, en remontant couche par couche.
- Chaque paramètre θ_ℓ est mis à jour en fonction de ces gradients.

Cas particulier de $\mathbf{W}_\ell$

- Pour certaines couches spécifiques (ex : BiMap), $\theta_\ell = \mathbf{W}_\ell$, où $\mathbf{W}_\ell$ est une matrice orthonormée.
- L'optimisation suit une descente de gradient riemannienne sur la variété de Stiefel (espace des matrices orthonormées).
- Cela nécessite une adaptation de la rétropropagation classique.
- Les calculs concernant donc $\mathbf{W}_\ell$ diffèrent, car ils sont basés sur la descente de gradient riemannienne sur la variété de Stiefel, où $\mathbf{W}_\ell$ représente une matrice orthonormée. Des explications détaillées de cette approche peuvent être trouvées dans [12]. Résumons ici les étapes de la rétropropagation pour le modèle proposé.

La Figure 3.7 montre donc comment les informations circulent dans le réseau et comment les gradients sont calculés et propagés en arrière pour ajuster les paramètres du modèle.

3.2.1 Définition de la fonction de perte

Étant donné la fonction de perte $\mathcal{L} : \mathbb{R}^K, \mathbb{R}^K \to \mathbb{R}$, où K est le nombre de classes. En notant f_l la l-ième couche du réseau, nous pouvons définir la perte à partir de cette couche comme :

$$\mathcal{L}^l = \underbrace{\mathcal{L} \circ f_\ell \circ \dots f_{l-1}}_{\mathcal{L}^{l-1}} \circ f_l, \tag{3.1}$$

où ℓ est le nombre total de couches dans le réseau.

3.2.2 Rétropropagation dans les différentes couches

Couche CovPool

Considérons cette couche comme une fonction f_l dans le réseau, ce qui signifie $\mathbf{C} = f_l(\mathbf{T}) = \mathbf{T}\bar{\mathbf{I}}\mathbf{T}^T$ et supposons que nous avons déjà calculé $\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{C}}$. Puisqu'il n'y a pas de paramètre à apprendre, la règle de la chaîne matricielle [57] sur $\mathcal{L}^l(\mathbf{T}) = \mathcal{L}^{l+1} \circ f_l(\mathbf{T})$ donne :

$$\langle \frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{C}}, \mathrm{d}\mathbf{C} \rangle = \langle \frac{\partial \mathcal{L}^{l}}{\partial \mathbf{T}}, \mathrm{d}\mathbf{T} \rangle, \tag{3.2}$$

où $\langle \cdot, \cdot \rangle$ est le produit scalaire de Frobenius tel que $\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{Tr}(\mathbf{A}^T\mathbf{B})$. Nous avons que $\mathrm{d}\mathbf{C} = (\mathrm{d}\mathbf{T})\bar{\mathbf{I}}\mathbf{T}^{\mathrm{T}} + \mathbf{T}\bar{\mathbf{I}}(\mathrm{d}\mathbf{T})^{\mathrm{T}}$ en remplaçant dans 3.2 et en utilisant les symétries sur $\bar{\mathbf{I}}$ et la relation $\mathrm{Tr}(\mathbf{B}^\top\mathbf{A}) = \mathrm{Tr}(\mathbf{A}\mathbf{B}^\top)$, nous obtenons :

$$\frac{\partial \mathcal{L}^l}{\partial \mathbf{T}} = 2\bar{\mathbf{I}}\mathbf{T}^{\mathrm{T}} \frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{C}}. \tag{3.3}$$

Rétropropagation pour les couches SPDNet

Les transformations SPDNet impliquent des opérations spécifiques sur les matrices symétriques définies positives (SPD). Les principales transformations sont :

- BiMap : une transformation bilinéaire contrainte.
- ReEig : une régularisation par rectification des valeurs propres.
- LogEig : une projection dans l'espace des matrices symétriques par le logarithme des valeurs propres.

Nous détaillons ci-dessous la rétropropagation pour chacune de ces transformations.

Couche BiMap

Ici, nous utilisons alors les mêmes principes de rétropropagation matricielle que ceux présentés pour la couche *CovPool*, mais appliqués à $f_l : \mathcal{S}^{d_{l-1}} \times \mathcal{O}_{d_l \times d_{l-1}} \to \mathcal{S}^{d_l}$, où $\mathcal{O}_{m\times n}$ est l'ensemble des matrices orthonormées de taille $m \times n$ et $\mathbf{X}_l = f_l(\mathbf{X}_{l-1}, \mathbf{W}) = \mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T$. Pour cette couche, nous avons deux gradients à propager, en supposant qu'on a calculé $\dfrac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}$:

- $\dfrac{\partial \mathcal{L}^l}{\partial \mathbf{X}_{l-1}}$, le gradient vers l'entrée de la couche.
- $\dfrac{\partial \mathcal{L}^l}{\partial \mathbf{W}}$, le gradient pour mettre à jour les poids $\mathbf{W}$ du mapping bilinéaire à apprendre.

Comme $\mathbf{W}$ a une structure particulière, il est nécessaire d'utiliser un opérateur de rétraction riemannienne [58] pour garder les poids sur la variété de Stiefel (certaines notions de Géométrie Riemannienne sont données dans l'annexe B). Les principales étapes sont décrites ci-dessous.

Principe de la rétropropagation : Nous exploitons la relation suivante entre les variations infinitésimales :

$$\left\langle \frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}, \mathrm{d}\mathbf{X}_l \right\rangle = \left\langle \frac{\partial \mathcal{L}^l}{\partial \mathbf{W}}, \mathrm{d}\mathbf{W} \right\rangle + \left\langle \frac{\partial \mathcal{L}^l}{\partial \mathbf{X}_{l-1}}, \mathrm{d}\mathbf{X}_{l-1} \right\rangle. \tag{3.4}$$

En considérant la transformation suivante :

$$\mathbf{X}_l = \mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T, \tag{3.5}$$

En différenciant l'expression $\mathbf{X}_l = \mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T$, nous obtenons :

$$\mathrm{d}\mathbf{X}_l = \mathrm{d}\mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T + \mathbf{W}\mathrm{d}\mathbf{X}_{l-1}\mathbf{W}^T + \mathbf{W}\mathbf{X}_{l-1}(\mathrm{d}\mathbf{W})^T. \tag{3.6}$$

Nous regroupons les termes en mettant $\mathrm{d}\mathbf{X}_{l-1}$ et $\mathrm{d}\mathbf{W}$ en facteur :

$$\mathrm{d}\mathbf{X}_l = \mathbf{W}\mathrm{d}\mathbf{X}_{l-1}\mathbf{W}^T + \left(\mathrm{d}\mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T + \mathbf{W}\mathbf{X}_{l-1}(\mathrm{d}\mathbf{W})^T\right). \tag{3.7}$$

en remplaçant dans 3.4

$$\mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \left(\mathbf{W}\mathrm{d}\mathbf{X}_{l-1}\mathbf{W}^T + \mathrm{d}\mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T + \mathbf{W}\mathbf{X}_{l-1}(\mathrm{d}\mathbf{W})^T\right)\right)$$

Par linéarité de la trace :

$$= \mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathbf{W}\mathrm{d}\mathbf{X}_{l-1}\mathbf{W}^T\right) + \mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathrm{d}\mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T + \left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathbf{W}\mathbf{X}_{l-1}(\mathrm{d}\mathbf{W})^T\right).$$

En utilisant la linéarité et la propriété de la trace $\mathrm{Tr}(\mathbf{A}\mathbf{B}^T) = \mathrm{Tr}(\mathbf{B}\mathbf{A}^T)$ pour $\mathbf{A} = \mathbf{W}\mathbf{X}_{l-1}$ et $\mathbf{B} = \mathrm{d}\mathbf{W}$, on réécrit le dernier terme :

$$= \mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathbf{W}\mathrm{d}\mathbf{X}_{l-1}\mathbf{W}^T\right) + \mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathrm{d}\mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T + \left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathrm{d}\mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T\right).$$

$$= \mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathbf{W}\mathrm{d}\mathbf{X}_{l-1}\mathbf{W}^T\right) + \mathrm{Tr}\left(2\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathrm{d}\mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T\right).$$

Regardons le premier terme pour le Calcul du gradient par rapport à $\mathbf{X}_{l-1}$: En utilisant la cyclicité de la trace $\mathrm{Tr}\,(\mathbf{ABC}) = \mathrm{Tr}\,(\mathbf{CAB})$ nous avons :

$$\mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathbf{W}\mathrm{d}\mathbf{X}_{l-1}\mathbf{W}^T\right) = \mathrm{Tr}\left(\mathbf{W}^T\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathbf{W}\mathrm{d}\mathbf{X}^{l-1}\right) = \mathrm{Tr}\left(\left(\mathbf{W}^T\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\mathbf{W}\right)^T \mathrm{d}\mathbf{X}^{l-1}\right)$$

En isolant la contribution de $\mathrm{d}\mathbf{X}_{l-1}$, nous obtenons :

$$\frac{\partial \mathcal{L}^l}{\partial \mathbf{X}_{l-1}} = \mathbf{W}^{\mathrm{T}}\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\mathbf{W}. \tag{3.8}$$

Regardons maintenant le calcul du gradient par rapport à $\mathbf{W}$: Nous avons :

$$\mathrm{Tr}\left(2\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathrm{d}\mathbf{W}\mathbf{X}_{l-1}\mathbf{W}^T\right) = \mathrm{Tr}\left(2\mathbf{X}_{l-1}\mathbf{W}^T\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \mathrm{d}\mathbf{W}\right) = \mathrm{Tr}\left(\left(2\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\mathbf{W}\mathbf{X}_{l-1}\right)^T \mathrm{d}\mathbf{W}\right)$$

En isolant la contribution de $\mathrm{d}\mathbf{W}$, nous trouvons :

$$\frac{\partial \mathcal{L}^l}{\partial \mathbf{W}} = 2\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\mathbf{W}\mathbf{X}_{l-1}. \tag{3.9}$$

Ces deux gradients résultent de la différentiation conjointe de $\mathbf{X}_l$ par rapport à $\mathbf{W}$ et $\mathbf{X}_{l-1}$, conformément à la règle du produit matriciel.

Rétraction sur la variété de Stiefel : Étant donné que $\mathbf{W}$ appartient à la variété de Stiefel, une mise à jour naïve via la descente de gradient classique ne garantirait pas que $\mathbf{W}$ reste sur cette variété. Nous appliquons donc une mise à jour adaptée :

$$\mathbf{W}_{\mathrm{new}} = R_{\mathbf{W}}\big(\mathbf{W} - \eta\tilde{\Delta}L_{\mathbf{W}}\big), \tag{3.10}$$

où $R_{\mathbf{W}}$ est l'opération de rétraction sur Stiefel (on rappelle que c'est une approximation de l'opérateur $\exp_{\mathbf{W}}$), η est le taux d'apprentissage, et $\tilde{\Delta}L_{\mathbf{W}}$ est la projection du gradient euclidien $\dfrac{\partial \mathcal{L}^l}{\partial \mathbf{W}}$ sur l'espace tangent de la variété de Stiefel :

$$\tilde{\Delta}L_{\mathbf{W}} = \frac{\partial \mathcal{L}^l}{\partial \mathbf{W}} - \frac{\partial \mathcal{L}^l}{\partial \mathbf{W}}\mathbf{W}^T\mathbf{W}. \tag{3.11}$$

Cette approche garantit que $\mathbf{W}$ reste une matrice orthogonale après mise à jour. Ainsi, nous avons explicité les calculs menant aux gradients et mis en avant l'importance de respecter la contrainte géométrique sur $\mathbf{W}$.

Couche Eig

La couche **Eig** applique la décomposition spectrale de la matrice SPD :

$$\mathrm{Eig}(\mathbf{X}) = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^T, \tag{3.12}$$

où $\mathbf{U}$ contient les vecteurs propres et $\mathbf{\Sigma}$ les valeurs propres de $\mathbf{X}$.
On peut réécrire 3.12 en considérant les couches :

$$\mathbf{X}_l = \mathbf{U}_{l-1}\mathbf{\Sigma}_{l-1}\mathbf{U}_{l-1}^T. \tag{3.13}$$

Pour appliquer les principes de rétropropagation dans la couche Eig , nous avons deux gradients à propager après avoir calculé $\dfrac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}$:

- $\dfrac{\partial \mathcal{L}^l}{\partial \mathbf{X}_{l-1}}$, le gradient vers l'entrée de la couche.
- $\dfrac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{U}_{l-1}}$, le gradient pour mettre à jour la matrice $\mathbf{U}_{l-1}$ (vecteurs propres).
- $\dfrac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{\Sigma}_{l-1}}$, le gradient pour mettre à jour la matrice $\mathbf{\Sigma}_{l-1}$ (valeurs propres).

Principe de la rétropropagation : Nous exploitons la règle de la chaîne matricielle entre les variations infinitésimales :

$$\left\langle \frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}, \mathrm{d}\mathbf{X}_l \right\rangle = \left\langle \frac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{U}_{l-1}}, \mathrm{d}\mathbf{U}_{l-1} \right\rangle + \left\langle \frac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{\Sigma}_{l-1}}, \mathrm{d}\mathbf{\Sigma}_{l-1} \right\rangle. \tag{3.14}$$

En considérant la transformation 3.13 pour la couche Eig, la variation $\mathrm{d}\mathbf{X}_l$ est donnée par :

$$\mathrm{d}\mathbf{X}_l = \mathrm{d}\mathbf{U}_{l-1}\mathbf{\Sigma}_{l-1}\mathbf{U}_{l-1}^T + \mathbf{U}_{l-1}\mathrm{d}\mathbf{\Sigma}_{l-1}\mathbf{U}_{l-1}^T + \mathbf{U}_{l-1}\mathbf{\Sigma}_{l-1}\mathrm{d}\mathbf{U}_{l-1}^T.$$

De plus on a :

$$\mathrm{d}\mathbf{U}_{l-1} = 2\mathbf{U}_{l-1}\left(\mathbf{P}^T \odot \left(\mathbf{U}_{l-1}^T \mathrm{d}\mathbf{X}_l \mathbf{U}_{l-1}\right)_{\mathrm{sym}}\right),$$

$$\mathrm{d}\mathbf{\Sigma}_{l-1} = \left(\mathbf{U}_{l-1}^T \mathrm{d}\mathbf{X}_l \mathbf{U}_{l-1}\right)_{\mathrm{diag}},$$

où $\mathbf{P}$ est une matrice carrée définie par :

$$\mathbf{P}(i,j) = \begin{cases} \dfrac{1}{\sigma_i - \sigma_j} & \text{si } i \neq j, \\ 0 & \text{sinon.} \end{cases} \tag{3.15}$$

Les σ_i correspondent aux valeurs propres de $\mathbf{X}_{l-1}$ et où $\odot$ est le produit de Hadamard, et les opérateurs $\mathbf{A}_{\mathrm{sym}}$ et $\mathbf{A}_{\mathrm{diag}}$ sont définis comme suit :

$$\mathbf{A}_{\mathrm{sym}} = \frac{1}{2}\left(\mathbf{A} + \mathbf{A}^T\right),$$

$\mathbf{A}_{\mathrm{diag}}$ est $\mathbf{A}$ avec tous les éléments hors diagonale mis à zéro.

Enfin, le gradient par rapport à $\mathbf{X}_{l-1}$ est donné par [15] :

$$\begin{aligned} \frac{\partial \mathcal{L}^l}{\partial \mathbf{X}_{l-1}} =& 2\mathbf{U}_{l-1}\left(\mathbf{P}^{\mathrm{T}} \odot \left(\mathbf{U}_{l-1}^{\mathrm{T}} \frac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{U}_{l-1}}\right)_{sym}\right)\mathbf{U}_{l-1}^{\mathrm{T}} \\ &+ \mathbf{U}_{l-1}\left(\frac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{\Sigma}_{l-1}}\right)_{diag}\mathbf{U}_{l-1}^{\mathrm{T}}, \end{aligned} \tag{3.16}$$

Couches ReEig et LogEig :

Étant donné que ces deux opérations reposent sur la décomposition en valeurs propres, nous pouvons décomposer $\mathbf{f}_l$ comme

$$\mathbf{f}_l = \mathbf{f}_l' \circ \mathrm{Eig}$$

où eig désigne la décomposition en valeurs propres (EVD) de la matrice d'entrée, et $\mathbf{f}_l'$ correspond à l'opération appliquée aux valeurs propres, suivie de la reconstruction.

La couche **ReEig** applique une régularisation sur les valeurs propres de la matrice SPD d'entrée :

$$\mathbf{f}_l = \mathbf{f}_l' \circ \mathrm{Eig}(\mathbf{X}) = \mathbf{U} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}) \mathbf{U}^T, \tag{3.17}$$

où $\mathbf{U}$ contient les vecteurs propres et $\mathbf{\Sigma}$ les valeurs propres de $\mathbf{X}$. Cette transformation permet de conserver uniquement les directions principales en imposant un seuil minimal ϵ aux valeurs propres, ce qui stabilise le calcul et améliore la robustesse de la représentation.

Principe de la rétropropagation : En considérant la transformation suivante pour la couche ReEig :

$$\mathbf{X}_l = \mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T.$$

La variation $\mathrm{d}\mathbf{X}_l$ est donnée par :

$$\mathrm{d}\mathbf{X}_l = \mathrm{d}\mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T + \mathbf{U}_{l-1} \mathrm{d}\left(\max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1})\right) \mathbf{U}_{l-1}^T + \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathrm{d}\mathbf{U}_{l-1}^T.$$

$$\mathrm{d}\mathbf{X}_l = 2 \left(\mathrm{d}\mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T\right)_{\mathrm{sym}} + \left(\mathbf{U}_{l-1} \mathbf{Q} \mathrm{d}\mathbf{\Sigma}_{l-1} \mathbf{U}_{l-1}^T\right)_{\mathrm{sym}}.$$

$$\mathrm{d}\mathbf{X}_l = \mathrm{d}\mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T + \mathbf{U}_{l-1} \mathrm{d}\left(\max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1})\right) \mathbf{U}_{l-1}^T + \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathrm{d}\mathbf{U}_{l-1}^T.$$

La fonction $\max(\epsilon, \mathbf{\Sigma}_{l-1})$ agit sur les valeurs propres de $\mathbf{X}_l$. Sa dérivée est donnée par :

$$\mathrm{d}\left(\max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1})\right) = \mathbf{Q}\, \mathrm{d}\mathbf{\Sigma}_{l-1},$$

où la matrice $\mathbf{Q}$ est diagonale avec des éléments :

$$Q(i,i) = \begin{cases} 1, & \text{si } \Sigma_{l-1}(i,i) > \epsilon, \\ 0, & \text{si } \Sigma_{l-1}(i,i) \leq \epsilon. \end{cases}$$

En remplaçant cette expression dans l'équation précédente, on obtient :

$$\mathrm{d}\mathbf{X}_l = \mathrm{d}\mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T + \mathbf{U}_{l-1} \mathbf{Q} \mathrm{d}\mathbf{\Sigma}_{l-1} \mathbf{U}_{l-1}^T + \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathrm{d}\mathbf{U}_{l-1}^T.$$

Puisque $\mathbf{X}_l$ est une matrice symétrique par construction, sa dérivée doit aussi être symétrique. On applique donc l'opération de symétrisation A_{sym} :

$$\mathrm{d}\mathbf{X}_l = \left(\mathrm{d}\mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T + \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathrm{d}\mathbf{U}_{l-1}^T\right)_{\mathrm{sym}} + \left(\mathbf{U}_{l-1} \mathbf{Q} \mathrm{d}\mathbf{\Sigma}_{l-1} \mathbf{U}_{l-1}^T\right)_{\mathrm{sym}}.$$

Enfin, par linéarité de l'opérateur A_{sym}, on peut écrire :

$$\mathrm{d}\mathbf{X}_l = 2 \left(\mathrm{d}\mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T\right)_{\mathrm{sym}} + \left(\mathbf{U}_{l-1} \mathbf{Q} \mathrm{d}\mathbf{\Sigma}_{l-1} \mathbf{U}_{l-1}^T\right)_{\mathrm{sym}}.$$

Cela justifie rigoureusement l'égalité demandée. Nous remplaçons cette expression dans le produit scalaire $\langle \frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}, \mathrm{d}\mathbf{X}_l \rangle$ donné dans 3.14, que l'on écrit sous forme de trace :

$$\mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \left(2 \left(\mathrm{d}\mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T\right)_{\mathrm{sym}} + \left(\mathbf{U}_{l-1} \mathbf{Q} \mathrm{d}\mathbf{\Sigma}_{l-1} \mathbf{U}_{l-1}^T\right)_{\mathrm{sym}}\right)\right)$$

Par linéarité de la trace, cela donne :

$$\mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T 2 \left(\mathrm{d}\mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T\right)_{\mathrm{sym}}\right) + \mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l}\right)^T \left(\mathbf{U}_{l-1} \mathbf{Q} \mathrm{d}\mathbf{\Sigma}_{l-1} \mathbf{U}_{l-1}^T\right)_{\mathrm{sym}}\right). \tag{3.18}$$

Calcul des gradients : En utilisant 3.18 et en identifiant dans 3.14, les sous-gradients pour la couche **ReEig** sont donnés par :

$$\frac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{U}_{l-1}} = 2 \left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l} \right)_{sym} \mathbf{U}_{l-1} \max(\epsilon \mathbf{I}, \mathbf{\Sigma}_{l-1}), \tag{3.19}$$

$$\frac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{\Sigma}_{l-1}} = \mathbf{Q} \mathbf{U}_{l-1}^{\mathrm{T}} \left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l} \right)_{sym} \mathbf{U}_{l-1}, \tag{3.20}$$

Enfin, le gradient par rapport à $\mathbf{X}_{l-1}$ est donné par 3.16

LogEig

La transformation **LogEig** applique la fonction logarithme aux valeurs propres :

$$\mathbf{f}_l = \mathbf{f}_l' \circ \mathrm{Eig}(\mathbf{X}) = \mathbf{U} \log(\mathbf{\Sigma}) \mathbf{U}^T. \tag{3.21}$$

où $\mathbf{f}_l'$ est log Cette transformation permet de projeter les matrices SPD dans un espace vectoriel euclidien en exploitant la géométrie de l'espace des matrices définies positives. Cela facilite la classification avec des modèles linéaires ou des couches fully connected.

Principe de la rétropropagation : Comme pour la couche **ReEig**, nous utilisons la relation suivante entre les variations infinitésimales donnée par 3.14 et étant donné

$$\mathbf{X}_l = \mathbf{U}_{l-1} \log(\mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T.$$

la variation $\mathrm{d}\mathbf{X}_l$ est donnée par :

$$\mathrm{d}\mathbf{X}_l = 2 \left(\mathrm{d}\mathbf{U}_{l-1} \log(\mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T \right)_{\mathrm{sym}} + \left(\mathbf{U}_{l-1} \mathbf{\Sigma}_{l-1}^{-1} \mathrm{d}\mathbf{\Sigma}_{l-1} \mathbf{U}_{l-1}^T \right)_{\mathrm{sym}}.$$

En remplaçant $\mathrm{d}\mathbf{X}_l$ dans 3.14 et en utilisant la linéarité de la trace, on a :

$$\mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l} \right)^T 2 \left(\mathrm{d}\mathbf{U}_{l-1} \log(\mathbf{\Sigma}_{l-1}) \mathbf{U}_{l-1}^T \right)_{\mathrm{sym}} \right) + \mathrm{Tr}\left(\left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l} \right)^T \left(\mathbf{U}_{l-1} \mathbf{\Sigma}_{l-1}^{-1} \mathrm{d}\mathbf{\Sigma}_{l-1} \mathbf{U}_{l-1}^T \right)_{\mathrm{sym}} \right). \tag{3.22}$$

Calcul des gradients : En utilisant 3.22 et en identifiant dans 3.14, les sous-gradients pour la couche **LogEig** sont donnés par

$$\frac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{U}_{l-1}} = 2 \left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l} \right)_{sym} \mathbf{U}_{l-1} \log(\mathbf{\Sigma}_{l-1}), \tag{3.23}$$

$$\frac{\partial \mathcal{L}^{l+1} \circ f_l^{'}}{\partial \mathbf{\Sigma}_{l-1}} = \mathbf{\Sigma}_{l-1}^{-1} \mathbf{U}_{l-1}^{\mathrm{T}} \left(\frac{\partial \mathcal{L}^{l+1}}{\partial \mathbf{X}_l} \right)_{sym} \mathbf{U}_{l-1}. \tag{3.24}$$

Enfin, le gradient par rapport à $\mathbf{X}_{l-1}$ est donné par 3.16

Conclusion

Chaque transformation dans SPDNet implique une rétropropagation adaptée à la structure des matrices SPD :

- BiMap : Applique une transformation linéaire contrainte, nécessitant une mise à jour des poids sur Stiefel.
- ReEig : Stabilise les matrices SPD en imposant un seuil sur les valeurs propres.
- LogEig : Transforme les matrices SPD dans un espace plus adapté aux distances euclidiennes.

Ces transformations permettent de capturer la structure géométrique des données SPD tout en garantissant une optimisation efficace du réseau.

3.3 Expérimentations

Dans cette section, tous les modèles sont utilisés avec les mêmes paramètres. L'optimiseur choisi est **SGD**, avec un momentum de 0.9, une taille de batch de 8 et un taux d'apprentissage de 0.007, comme dans la référence 2.4.6. Nous reproduisons les mêmes expérimentations avec **SRCNet** et **RCNet**.

Notre matrice de covariance est de taille 256×256, ce qui représente une dimension élevée. Pour réduire cette complexité, nous utilisons les couches **BiMap** et **ReEig**, que nous répétons ensuite quatre fois. Le tableau 3.1 présente les dimensions de la matrice de covariance avec **SRCNet** et **RCNet** après l'application de ces transformations.

	d_0	d_2	d_4	d_6	d_8
SRCNet	256	235	217	179	128
RCNet	64	58	54	44	32

TABLE 3.1 – Dimensions de sortie des couches SPDNet.

Dans le Chapitre 2, nous avons observé que les performances étaient meilleures avec un ResNet-34 *fine-tuned*. Nous allons donc comparer les performances des CNN *shallow* et profonds avec nos modèles **SRCNet** et **RCNet**. Avec ces dimensions pour les couches de SPDNet, le modèle **SRCNet** possède environ 323 000 paramètres, tandis que le modèle **RCNet** en possède environ 93 000, soit moins que le **S-CNN** présenté dans le Chapitre 2, qui comporte environ 120 000 paramètres.

3.3.1 Influence du nombre de données d'apprentissage

Comme nous pouvons le voir sur la Figure 3.8, globalement, les résultats sont à chaque fois meilleurs avec **RCNet** et **SRCNet** ; seul RRT a des résultats à peu près similaires pour des ratios supérieurs à 0.5. Pour les petits ratios, le modèle **RCNet** performe globalement mieux que les autres et que **SRCNet**.

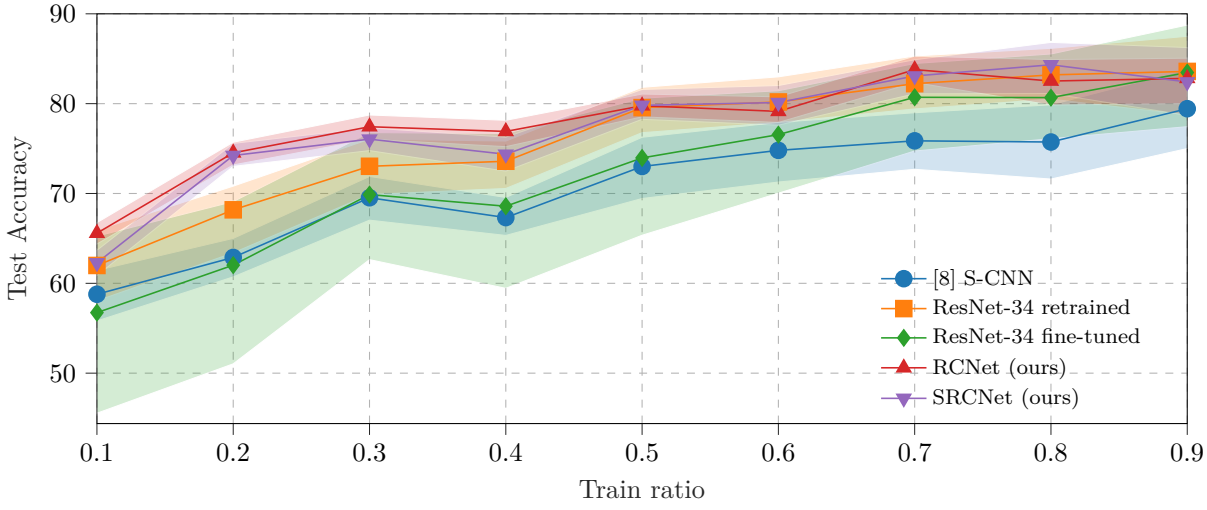


FIGURE 3.8 – Comparaison des résultats de **RCNet** et **SRCNet** avec S-CNN, RRT et RFT. Précision sur les données test par rapport au pourcentage de l'ensemble de données d'apprentissage sur 100 graines différentes. Pour chaque méthode, la ligne correspond à la moyenne de la précision sur toutes les graines, et la zone remplie correspond aux 5ème et 95ème quantiles.

3.3.2 Robustesse aux données mal étiquetées

La deuxième expérimentation étudie l'effet de la présence de données mal étiquetées dans l'ensemble d'entraînement. En effet, il peut être difficile d'étiqueter correctement les hyperboles, notamment en raison du faible rapport signal/bruit (RSB) dans les radargrammes. Pour cette expérimentation, nous introduisons un niveau variable de données mal étiquetées, allant de 0 à 20 % d'erreur, dans l'ensemble d'entraînement. Ici, le ratio des données d'entraînement est de 70%, soit 1108 imagettes provenant de notre dataset (cf 1.13).

Les résultats de précision des tests en fonction du pourcentage d'erreurs d'étiquetage sont présentés dans la Figure 3.9. Nous observons une dynamique comparable à celle de l'expérience précédente, avec une meilleure performance et robustesse des modèles **RCNet** et **SRCNet** (les résultats sont meilleurs lorsque seule la dernière couche est prise en compte pour la construction du tenseur). Le cas de [8] est intéressant, car il montre que les performances des modèles peu profonds diminuent rapidement dès que quelques données sont mal étiquetées. Comme dans l'expérimentation précédente, nos approches présentent la variabilité la plus faible en fonction des jeux de données. En conclusion, ce résultat est en accord avec l'analyse réalisée dans [59], qui montre également que l'utilisation des matrices de covariance apporte une grande robustesse face aux erreurs d'étiquetage dans les méthodes d'apprentissage métrique.

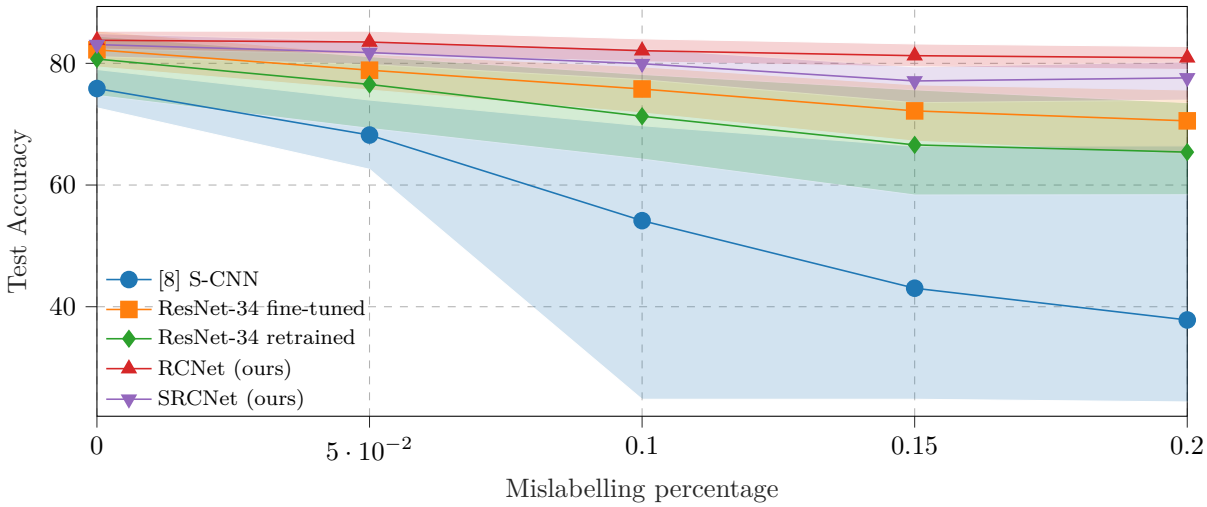


FIGURE 3.9 – Résultats de la précision des tests en fonction du pourcentage d'erreurs d'étiquetage dans l'ensemble d'entraînement. Pour chaque méthode **RCNet**, **SRCNet**, S-CNN, RRT et RFT, la ligne correspond à la moyenne de la précision sur l'ensemble des jeux de données, et la zone remplie correspond aux quantiles 5% et 95%.

3.3.3 Robustesse aux changements de distribution des données

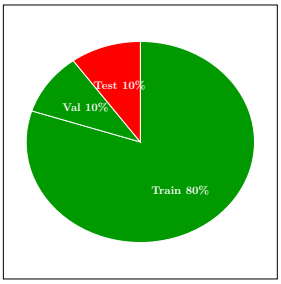
Dans le cas des données GPR, celles-ci sont collectées dans différentes configurations : différentes fréquences et altitudes pour le radar, ainsi que différents types de sol et de conditions météorologiques. Comme il est difficile (voire impossible) d'obtenir des données étiquetées pour toutes ces configurations, il est essentiel de construire des modèles capables d'être robustes aux décalages entre les données d'entraînement, de validation et de test. Ces transformations, courantes dans de nombreuses applications, sont connues sous le nom de *changements de distribution des données* (data shifts). Afin d'étudier le comportement des modèles proposés dans cette partie, nous considérons quatre scénarios :

* **Scénario A** : les ensembles d'entraînement et de validation incluent des images obtenues avec un radar situé à une altitude de 75 cm ou 100 cm, tandis que l'ensemble de test est constitué d'images

acquises à une altitude de 50 cm.

* **Scénario B** : la fréquence est de 200 MHz pour les ensembles d'entraînement et de validation, tandis que l'ensemble de test ne contient que des données obtenues avec une fréquence de 350 MHz.
* **Scénario C** : les ensembles d'entraînement et de validation sont constitués de données acquises sur du grave sèche, alors que l'ensemble de test est réalisé sur du grave humide.
* **Scénario D** : similaire au scénario C, mais avec du sable humide pour les ensembles d'entraînement et de validation, et du sable sec pour l'ensemble de test.

La distribution et le nombre d'imagettes dans les différents ensembles pour chaque scénario sont présentés dans la Figure 3.10.



(A) ELV	**75 100**	vs	**50**	791 imagettes
(B) FRQ	**350**	vs	**200**	491 imagettes
(C) SOL-G	**Grave sèche**	vs	**Grave humide**	492 imagettes
(D) SOL-S	**Sable humide**	vs	**Sable sec**	840 imagettes

FIGURE 3.10 – Les quatre jeux de données considérés pour l'étude du décalage ou du changement de distribution. Le nombre total d'images est indiqué à droite pour chaque scénario, avec une répartition fixe : 80 % pour l'entraînement, 10 % pour la validation et 10 % pour le test.

.

Scénario A : Changement d'élévation du radar GPR

Le scénario A (ou du Dataset A) comprend 791 échantillons, répartis selon l'élévation du radar (50, 75, 100 cm), la fréquence d'acquisition (200, 350 MHz), le type de sol (sable sec, sable humide, grave sèche, grave humide) et la nature de la cible (abris en bois, métallique, non métallique ou empty). Le sol sableux constitue la majorité du jeu de données avec 368 échantillons, suivi par le sable humide (224), la grave sèche (111) et la grave humide (88), comme indiqué dans le tableau 3.2. La distribution selon l'élévation est relativement équilibrée, avec une légère surreprésentation à 75 cm (340 échantillons).

	Abris en bois			Métallique			Non Métallique			Empty			**Total**
Sol/Élévation (cm)	**50**	**75**	**100**	**50**	**75**	**100**	**50**	**75**	**100**	**50**	**75**	**100**	
grave humide	3	12	5	6	13	7	3	9	3	5	9	13	**88**
grave sèche	2	15	16	3	12	5	2	12	10	4	18	12	**111**
sable sec	10	43	43	5	41	41	6	39	45	9	42	44	**368**
sable humide	6	20	24	7	30	29	5	27	33	3	19	21	**224**
Total	21	90	88	21	96	82	16	87	91	21	88	90	**791**

TABLE 3.2 – Distribution des classes du Dataset A pour chaque élévation (50 cm, 75 cm, 100 cm) en fonction du type de sol.

Concernant la fréquence, le jeu de données est dominé par les acquisitions à 350 MHz (509 échantillons), contre 282 à 200 MHz dans le tableau 3.3. Ce déséquilibre entre les types de sols, les fréquences

et les élévations devra être pris en compte dans la phase d'entraînement, notamment pour éviter un apprentissage biaisé.

	Abris en bois			**Métallique**			**Non Métallique**			**Empty**			**Total**
Fréquence/Élévation (cm)	**50**	**75**	**100**	**50**	**75**	**100**	**50**	**75**	**100**	**50**	**75**	**100**	
200 MHz	11	50	46	4	25	15	6	23	19	10	34	39	**282**
350 MHz	10	40	42	17	71	67	10	64	72	11	54	51	**509**
Total	21	90	88	21	96	82	16	87	91	21	88	90	**791**

TABLE 3.3 – Distribution des classes du Dataset A pour chaque élévation (50 cm, 75 cm, 100 cm) en fonction de la fréquence (MHz).

La Figure 3.11 présente des boîtes à moustaches des performances en précision des différents modèles. Nous pouvons facilement conclure que nos modèles, en particulier **RCNet**, maintiennent des performances quasi inchangées par rapport au cas classique, ce qui prouve leur robustesse face à cette transformation. Dans ce cas, les principales transformations sont des changements d'échelle, bien que la forme de l'hyperbole varie légèrement. Comme prévu, le modèle peu profond est le moins robuste. Nous remarquons également que la variabilité des performances entre les différents jeux de données est très faible avec les modèles construits à partir de matrices de covariance. Ce résultat est cohérent avec ceux obtenus dans [60], où les matrices de covariance ont été utilisées pour classifier des cultures dans des séries temporelles d'images satellites.

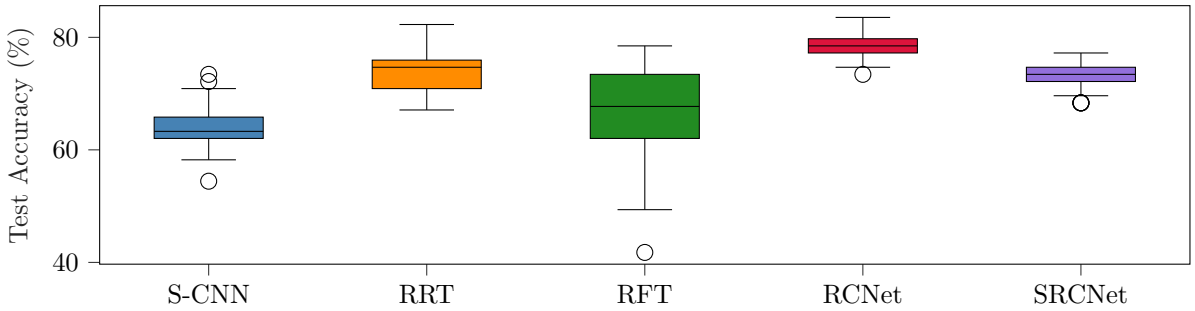


FIGURE 3.11 – Résultats de la précision des tests pour le scénario A (ELV 75,100 vs 50).

Scénario B : Changement de fréquence

Le scénario B comprend 491 échantillons, répartis selon la fréquence (200 MHz, 350 MHz), l'élévation du radar (de 0 à 150 cm), le type de sol (sable sec, sable humide, grave sèche, grave humide) et la catégorie de la cible. Le sable sec constitue la majorité du jeu de données avec 231 échantillons, suivi par le sable humide (128), la grave sèche (68) et la grave humide (64), comme présenté dans le tableau 3.4. La fréquence de 350 MHz est fortement surreprésentée, avec 442 échantillons contre seulement 49 à 200 MHz.

	Abris en bois		**Métallique**		**Non Métallique**		**Empty**		**Total**
Sol/ Fréquence	**200 MHz**	**350 MHz**	**200 MHz**	**350 MHz**	**200 MHz**	**350 MHz**	**200 MHz**	**350 MHz**	
grave humide	2	7	2	19	2	13	1	18	64
grave sèche	4	15	1	13	2	12	5	16	68
sable sec	5	60	5	47	2	58	7	47	231
sable humide	3	28	6	32	1	27	1	30	128
Total	14	110	14	111	7	110	14	111	491

TABLE 3.4 – Distribution des classes du Dataset B pour chaque fréquence (200 MHz, 350 MHz) en fonction du type de sol.

En ce qui concerne l'élévation, les niveaux les plus bas (0 à 75 cm) regroupent environ deux tiers des échantillons, avec une densité particulièrement forte à 25, 50 et 75 cm (respectivement 93, 84 et 85 échantillons), comme indiqué dans le tableau 3.5. Ces déséquilibres doivent être pris en compte dans la phase d'entraînement pour limiter les biais d'apprentissage liés à la fréquence ou à l'élévation.

	Abris en bois		**Métallique**		**Non Métallique**		**Empty**		**Total**
Élévation (cm)/ Fréquence	**200 MHz**	**350 MHz**	**200 MHz**	**350 MHz**	**200 MHz**	**350 MHz**	**200 MHz**	**350 MHz**	
0	2	21	2	12	1	27	3	24	92
25	1	20	2	19	3	22	3	23	93
50	4	17	1	23	1	24	1	13	84
75	0	20	6	14	1	20	4	20	85
100	2	17	0	26	1	10	0	18	74
150	5	15	3	17	0	7	3	13	63
Total	14	110	14	111	7	110	14	111	491

TABLE 3.5 – Distribution des classes du Dataset B pour chaque fréquence (200 MHz, 350 MHz) en fonction de l'élévation (cm).

Les diagrammes en boîte sont présentés dans la Figure 3.12. Comme dans le scénario précédent, les modèles proposés surpassent les modèles classiques. Cependant, dans ce cas, le meilleur algorithme est **SRCNet**, et nous observons une forte dégradation des performances pour toutes les approches. Le nombre de transformations entre les deux ensembles est trop important, car la fréquence influe sur la pénétration de l'onde dans le sol et sur la résolution obtenue, entraînant ainsi des radargrammes très différents.

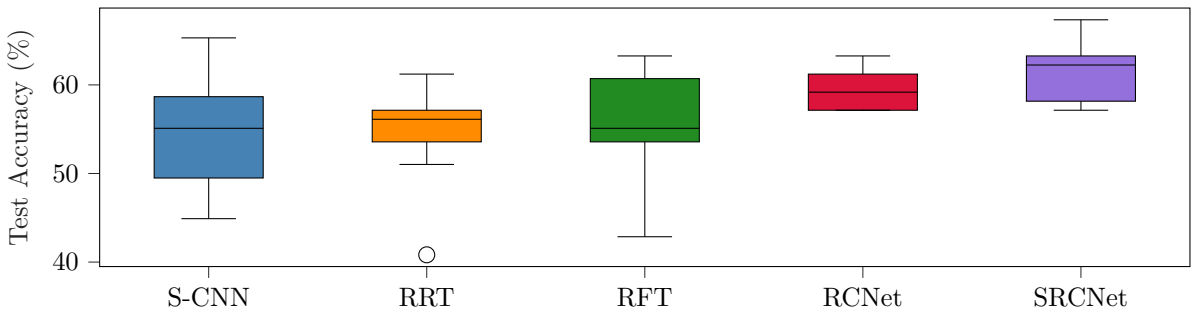


FIGURE 3.12 – Résultats de la précision des tests pour le scénario B (FRQ 350 vs 200).

Scénarios C et D : Changement du type de sol

Le scénario C contient 491 échantillons répartis équitablement entre deux types de sol : la grave humide (242 échantillons) et la grave sèche (249 échantillons), comme illustré dans les tableaux 3.6 et 3.7. La fréquence de 350 MHz est majoritaire avec 442 échantillons, contre seulement 49 à 200 MHz. La distribution selon l'élévation est également inégale, avec une concentration plus importante aux niveaux bas (0 à 50 cm) représentant plus de la moitié du jeu de données. On note une répartition relativement homogène entre les différentes catégories cibles, bien que la classe « métallique » soit légèrement surreprésentée, avec 125 occurrences.

	Abris en bois		Métallique		Non Métallique		Empty		Total
Élévation (cm)/ Sol	grave humide	grave sèche	grave humide	grave sèche	grave humide	grave sèche	grave humide	grave sèche	
0	2	25	1	35	7	46	1	31	148
25	3	20	4	17	0	20	5	16	85
50	5	21	3	17	0	19	1	19	85
75	3	19	3	16	0	16	2	14	73
100	0	14	2	12	0	9	2	18	57
150	1	11	1	14	0	0	3	13	43
Total	14	110	14	111	7	110	14	111	491

TABLE 3.6 – Distribution des classes du Dataset C en fonction du type de sol (grave humide, grave sèche) et de l'élévation (cm).

	Abris en bois		Métallique		Non Métallique		Empty		Total
Fréquence / Sol	grave humide	grave sèche	grave humide	grave sèche	grave humide	grave sèche	grave humide	grave sèche	
200 MHz	10	4	0	14	1	6	9	5	49
350 MHz	72	38	33	78	42	68	64	47	442
Total	82	42	33	92	43	74	73	52	491

TABLE 3.7 – Distribution des classes du Dataset C en fonction du type de sol (grave humide, grave sèche) et de la fréquence (MHz).

Le scénario D est plus volumineux avec 840 échantillons, répartis entre deux types de sol : sable sec (462 échantillons) et sable humide (378 échantillons), comme présenté dans les tableaux 3.8 et 3.9. Comme dans les scénarios précédents, la fréquence de 350 MHz domine très largement avec 756 échantillons contre 84 à 200 MHz. La distribution selon l'élévation est équilibrée, bien que les hauteurs de 75 cm et 100 cm soient légèrement plus représentées. Les différentes classes cibles sont globalement bien réparties, et chaque combinaison sol/catégorie est suffisamment représentée pour envisager un entraînement stable dans un cadre supervisé.

	Abris en bois		Métallique		Non Métallique		Empty		Total
Élévation (cm)/ Sol	sable sec	sable humide	sable sec	sable humide	sable sec	sable humide	sable sec	sable humide	
0	4	32	3	36	5	80	4	47	211
25	2	33	3	31	4	27	6	24	130
50	2	33	3	29	4	27	1	27	126
75	5	32	6	30	1	28	5	34	141
100	5	30	4	34	3	22	1	32	131
150	4	29	3	29	1	5	5	25	101
Total	22	189	22	189	18	189	22	189	840

TABLE 3.8 – Distribution des classes du Dataset D en fonction du type de sol (sable sec, sable humide) et de l'élévation (cm).

	Abris en bois		Métallique		Non Métallique		Empty		Total
Fréquence / Sol	sable sec	sable humide	sable sec	sable humide	sable sec	sable humide	sable sec	sable humide	
200 MHz	11	11	6	16	5	13	14	8	84
350 MHz	88	101	67	122	61	128	60	129	756
Total	99	112	73	138	66	141	74	137	840

TABLE 3.9 – Distribution des classes du Dataset D en fonction du type de sol (sable sec, sable humide) et de la fréquence (MHz).

Les diagrammes en boîte pour le scénario C sont présentés dans la Figure 3.13, tandis que ceux du scénario D sont illustrés dans la Figure 3.14. Dans les deux cas, les modèles de deep learning de second ordre offrent de meilleurs résultats et une variabilité plus faible. Nous pouvons conclure que, pour garantir des performances robustes, nos approches sont clairement mieux adaptées que les modèles peu profonds ou les modèles de deep learning classiques, tels que **ResNet-34**.

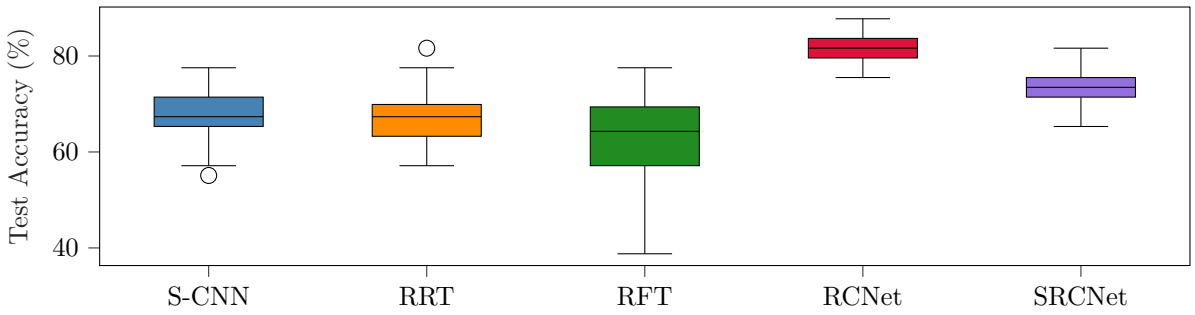


FIGURE 3.13 – Résultats de la précision des tests pour le scénario C (SOL-G grave sèche vs grave humide).

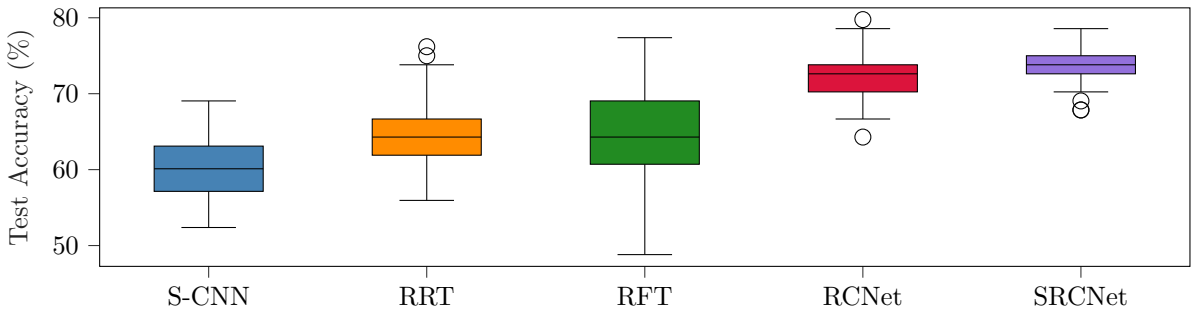


FIGURE 3.14 – Résultats de la précision des tests pour le scénario D (SOL-S sable humide vs sable sec).

3.4 Conclusion

Ce chapitre a été consacré au développement d'un modèle robuste pour la classification des cibles dans les données GPR, en s'appuyant sur une représentation par matrices de covariance et sur l'exploitation de la géométrie riemannienne des matrices symétriques définies positives (SPD). Cette approche répond aux limitations identifiées dans le chapitre 2, notamment la sensibilité des modèles profonds aux décalages expérimentaux, la faible disponibilité de données annotées et la présence de labels bruités.

Pour cela, nous avons introduit deux architectures hybrides : **RCNet** (Residual Covariance Network) et **SRCNet** (Stacked Residual Covariance Network). Ces modèles combinent une extraction de caractéristiques convolutionnelles (issues des premières couches de ResNet-34 entraîné from-scratch) avec un pooling de covariance, permettant d'encoder des statistiques de second ordre à partir des features locales. Ces représentations sont ensuite traitées dans un espace géométrique adapté, à l'aide d'un réseau **SPDNet** composé de couches spécifiques :

- *BiMap*, pour effectuer des projections sur des sous-espaces SPD,
- *ReEig*, pour régulariser les spectres propres des matrices,
- *LogEig*, pour transformer les matrices SPD dans un espace euclidien via le logarithme matriciel.

Les expérimentations ont mis en évidence plusieurs résultats majeurs :

- **RCNet** obtient les meilleures performances globales, surpassant notamment ResNet-34 fine-tuned (RFT) ou réentraîné (RRT), avec une robustesse marquée en cas de faible ratio d'entraînement.
- **SRCNet**, tout en étant plus compact, maintient d'excellents résultats, en particulier sur les tests de décalage de distribution (sol, fréquence, élévation).

- Les deux modèles à base de covariance montrent une variabilité inter-répétition très faible, preuve de leur stabilité et de leur capacité à généraliser à des données non vues lors de l'entraînement.

Par ailleurs, nous avons validé leur robustesse dans plusieurs configurations simulant des décalages expérimentaux réalistes :

- Changement d'élévation (75 cm/100 cm vs 50 cm),
- Changement de fréquence (350 MHz vs 200 MHz),
- Changement de type de sol (grave sèche vs grave humide, sable humide vs sable sec).

Dans tous ces cas, les modèles **RCNet** et **SRCNet** conservent un haut niveau de performance, bien supérieur à celui obtenu avec des approches CNN classiques ou avec des descripteurs plus simples comme ceux de S-CNN.

En résumé, ce chapitre a démontré l'intérêt de structurer les données extraites d'un CNN sous forme de matrices de covariance, puis de les traiter dans un espace géométrique adapté via SPDNet. Cette approche permet de construire des modèles plus robustes, particulièrement efficaces dans des contextes de faible volume de données et de forte hétérogénéité expérimentale.

Cependant, toutes les méthodes présentées ici supposent que la localisation des objets enfouis est connue à l'avance, ce qui constitue une hypothèse forte et rarement vérifiable en pratique. Pour dépasser cette limite, nous proposons dans le chapitre 4 qui suit une approche conjointe de détection et de classification, en adaptant le détecteur *Faster R-CNN* au contexte GPR. Cette extension intégrera les représentations par covariance étudiées ici dans une architecture end-to-end, capable de localiser automatiquement les objets d'intérêt tout en les classifiant de manière fiable.

Chapitre 4

Faster R-CNN et CNN-SPDNet

Dans le chapitre précédent, nous avons étudié plusieurs modèles de classification, en supposant que la localisation des objets dans les images GPR était connue et déjà effectuée. Dans ce chapitre, nous abordons une approche unifiée de détection d'objets, qui intègre simultanément la localisation et la classification des cibles au sein d'un même réseau. En vision par ordinateur, la détection d'objets regroupe deux tâches complémentaires : la localisation spatiale des cibles à l'aide de boîtes englobantes et leur classification selon des catégories prédéfinies. Dans le contexte du GPR, il est essentiel de pouvoir localiser précisément les objets enfouis tout en les identifiant de manière fiable. Une approche conjointe permet de tirer parti à la fois des relations spatiales et des caractéristiques locales du signal radar. Nous proposons ici une adaptation du modèle ***Faster R-CNN*** *à la détection sur des données GPR, en y intégrant un module de normalisation de second ordre fondé sur des matrices de covariance (MPN-COV) projetées dans un espace SPD. Cette combinaison vise à enrichir les descripteurs extraits par le backbone, afin d'améliorer la capacité du modèle à discriminer les cibles (hyperboles), tout en maintenant une localisation cohérente. Nous présentons les métriques d'évaluation utilisées dans ce contexte, ainsi que les premiers résultats expérimentaux obtenus sur notre jeu de test. Ces résultats préliminaires suggèrent un gain potentiel, mais nécessitent d'être confirmés par des études complémentaires.*

Sommaire

4.1 Faster R-CNN

Dans cette section, nous présentons l'architecture Faster R-CNN, l'un des modèles de détection d'objets les plus performants et largement utilisés [61, 62]. Avant de détailler les étapes, nous introduisons brièvement les modèles R-CNN et Fast R-CNN qui les ont précédés et sur lesquels ils s'appuient. Nous décrirons ensuite les différentes étapes de Faster R-CNN en prenant pour exemple une image GPR. Enfin, nous détaillerons les fonctions de perte utilisées, les paramètres appris ainsi que le procédé d'optimisation associé.

4.1.1 Architectures précédentes

R-CNN (Region-based Convolutional Neural Network, 2014) Le modèle R-CNN a été introduit par Girshick et al. en 2014 [63]. Il s'agit du premier détecteur d'objets à utiliser la notion de régions d'intérêt (RoI) pour guider la détection.

Le processus commence par l'extraction de régions candidates à l'aide d'un algorithme de propositions, tel que *Selective Search* [64], qui génère environ 2000 propositions indépendantes de la catégorie.

Chaque région d'intérêt est ensuite redimensionnée à une taille fixe, puis passée à travers un réseau de neurones convolutionnel (CNN) afin d'extraire des descripteurs de caractéristiques. Ces descripteurs sont ensuite utilisés pour la classification à l'aide d'un classifieur SVM. En parallèle, un régresseur linéaire affine les coordonnées des boîtes englobantes des objets détectés.

Enfin, pour éliminer les redondances parmi les propositions, une méthode de suppression des doublons appelée **NMS (Non-Maximum Suppression)** est appliquée. Il supprime les boîtes qui se chevauchent trop fortement, en conservant uniquement les boîtes avec le score de confiance le plus élevé et en utilisant un certain seuil d'IoU (Intersection over Union), qui est le rapport entre l'intersection et l'union des boîtes englobantes de la détection prédite et de la vérité terrain (voir Annexe D). Il répète ensuite le processus avec la boîte suivante la plus élevée parmi celles restantes.

La Figure 4.1 illustre les différentes étapes du pipeline R-CNN sur une image GPR (radargramme). L'image en entrée est d'abord segmentée en régions d'intérêt. Ces régions sont redimensionnées, et chacune est traitée individuellement par un CNN. Le schéma montre également, en sortie, les boîtes englobantes résultantes superposées à l'image GPR, représentant les objets détectés et classifiés.

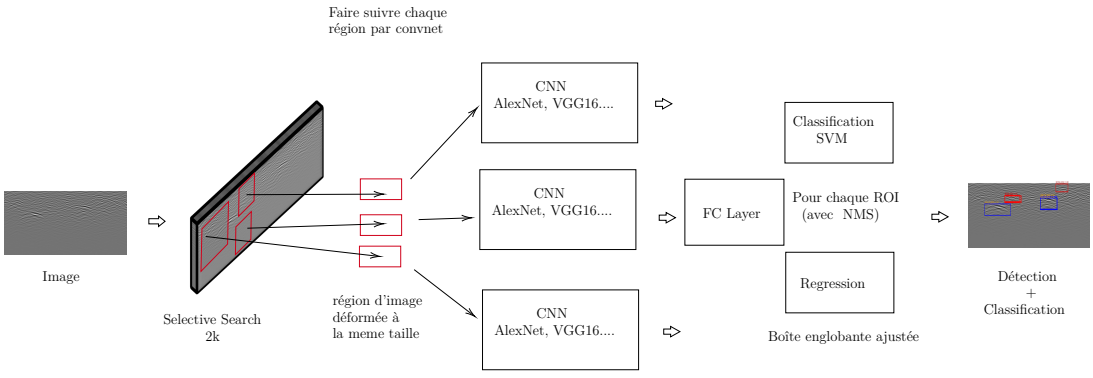


FIGURE 4.1 – Schéma de R-CNN : application sur une image GPR.

Fast R-CNN Le modèle Fast R-CNN, proposé par les mêmes auteurs en 2015 [65], introduit plusieurs améliorations en termes d'efficacité computationnelle.

Contrairement à R-CNN où chaque RoI est traité indépendamment par un CNN, Fast R-CNN traite l'image entière en une seule passe à travers le CNN.

Après cette passe unique, une couche de *RoI pooling* est utilisée pour extraire les régions d'intérêt à partir de la carte de caractéristiques globale. Cette couche construit, pour chaque région d'intérêt, une représentation de taille fixe compatible avec les couches entièrement connectées suivantes.

Les caractéristiques extraites par cette opération sont ensuite utilisées à la fois pour la classification (prédiction des classes des objets) et pour la régression des coordonnées des boîtes englobantes. Comme dans R-CNN, une étape de *NMS* est appliquée à la fin du processus.

La Figure 4.2 illustre ce flux de traitement sur une image GPR. L'image radargramme est d'abord entièrement analysée par un CNN, produisant une carte de caractéristiques. Les régions d'intérêt sont ensuite projetées sur cette carte et traitées via la couche RoI pooling. Le schéma montre également le résultat final avec plusieurs boîtes de détection superposées à l'image GPR.

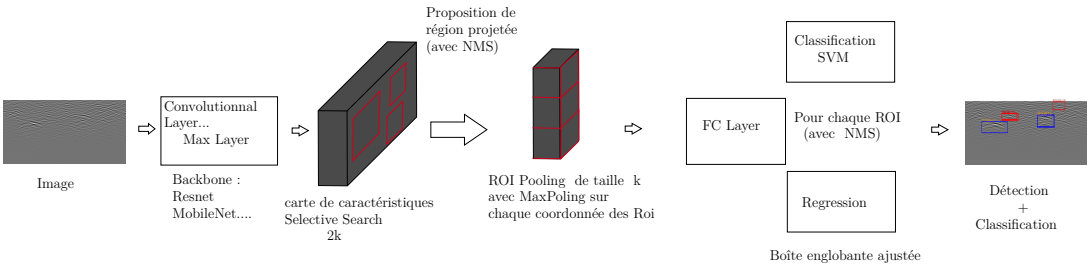


FIGURE 4.2 – Schéma de Fast R-CNN : application sur une image GPR.

Faster R-CNN Le modèle Faster R-CNN, proposé par Ren et al. en 2015 [66], constitue une évolution directe de Fast R-CNN [65], lui-même basé sur le modèle R-CNN initial [63].
Alors que Fast R-CNN améliore la vitesse et la précision de la détection d'objets en partageant les calculs convolutionnels pour l'ensemble de l'image, il reste dépendant d'un algorithme externe (comme Selective Search) pour générer les régions d'intérêt (RoIs). Cette étape reste coûteuse en temps de calcul et n'est pas optimisée conjointement avec le reste du réseau.

Faster R-CNN surmonte cette limitation en introduisant un module appelé *Region Proposal Network (RPN)*, entraîné de manière conjointe avec le reste du modèle. Le RPN partage les mêmes couches convolutionnelles que le réseau principal, ce qui permet de générer efficacement des propositions de régions directement à partir de la carte de caractéristiques de l'image.
Ces propositions sont ensuite traitées par une couche de RoI pooling, comme dans Fast R-CNN, suivie d'un réseau fully connected qui effectue à la fois la classification des objets et la régression des boîtes englobantes.

La suppression des doublons détectés est ensuite effectuée à l'aide de la méthode *Non-Maximum Suppression (NMS)*, comme dans les modèles précédents.

La Figure 4.3 illustre le fonctionnement de Faster R-CNN sur une image GPR. Le schéma montre une image radargramme en entrée, suivie de l'extraction de caractéristiques par un CNN, la génération des propositions par le RPN, puis la classification et la régression des boîtes. Les boîtes de détection finales sont représentées sur l'image en sortie.

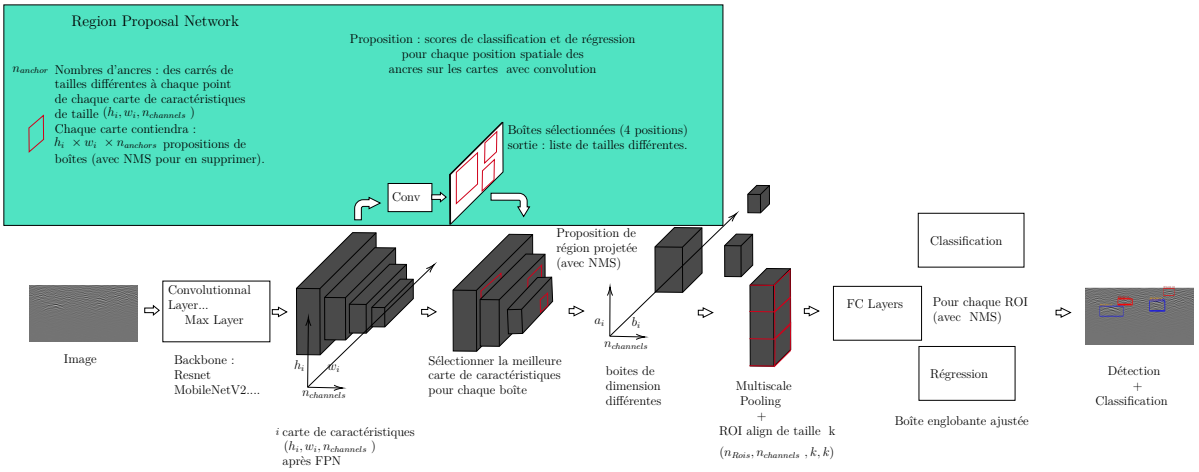


FIGURE 4.3 – Schéma de Faster R-CNN.

4.1.2 Exemple d'application de Faster R-CNN avec ResNet50 et FPN

1. Image d'entrée

L'image d'entrée est la représentation numérique d'une scène capturée par un capteur (ex. caméra) Dans notre cas, il s'agit de l'image d'un radargramme. Dans Faster R-CNN, elle est de taille (H, W) avec 3 canaux de couleur (RGB) et est traitée en batch de taille N.

- **Dimensions** : $[N, 3, H, W]$
  - N : Nombre d'images dans le lot (batch) (taille du batch).
  - 3 : Nombre de canaux (correspondant aux couleurs RGB).
  - H, W : Hauteur et largeur de l'image.

2. Backbone (ResNet-50)

Le *backbone* constitue la première étape de l'architecture Faster R-CNN. Il est chargé d'extraire des représentations riches en caractéristiques à partir des images GPR en entrée. Parmi les modèles testés dans les chapitres 2 et 3, les architectures de type ResNet ont montré les meilleurs compromis entre robustesse et expressivité des descripteurs. À titre d'exemple, nous illustrons ici le fonctionnement du détecteur avec un ResNet-50, mais les simulations présentées par la suite sont réalisées avec un ResNet-34.

ResNet-50 génère des cartes de caractéristiques à différentes profondeurs du réseau, nommées C2, C3, C4 et C5. Ces cartes ont respectivement pour dimensions : [N, 256, H/4, W/4] pour C2, [N, 512, H/8, W/8] pour C3, [N, 1024, H/16, W/16] pour C4, et [N, 2048, H/32, W/32] pour C5 (Voir un peu plus loin la Figure 4.4). Ces différentes résolutions permettent de capturer à la fois les structures globales du sol et les détails fins des objets souterrains visibles sur les images GPR.

3. Le cou : FPN (Feature Pyramid Network)

Le **Feature Pyramid Network (FPN)** [67] occupe le rôle de *cou* dans l'architecture et permet de renforcer les capacités multi-échelles (multiscales) du modèle. En utilisant les cartes C2 à C5 extraites par le backbone, le FPN construit une pyramide de caractéristiques hiérarchisées notées P2 à P5. Ce mécanisme est essentiel pour détecter efficacement des objets souterrains de tailles variées.

Le FPN repose sur une voie descendante (top-down) qui suréchantillonne les caractéristiques des couches profondes, combinée à des connexions latérales avec les couches correspondantes du backbone.

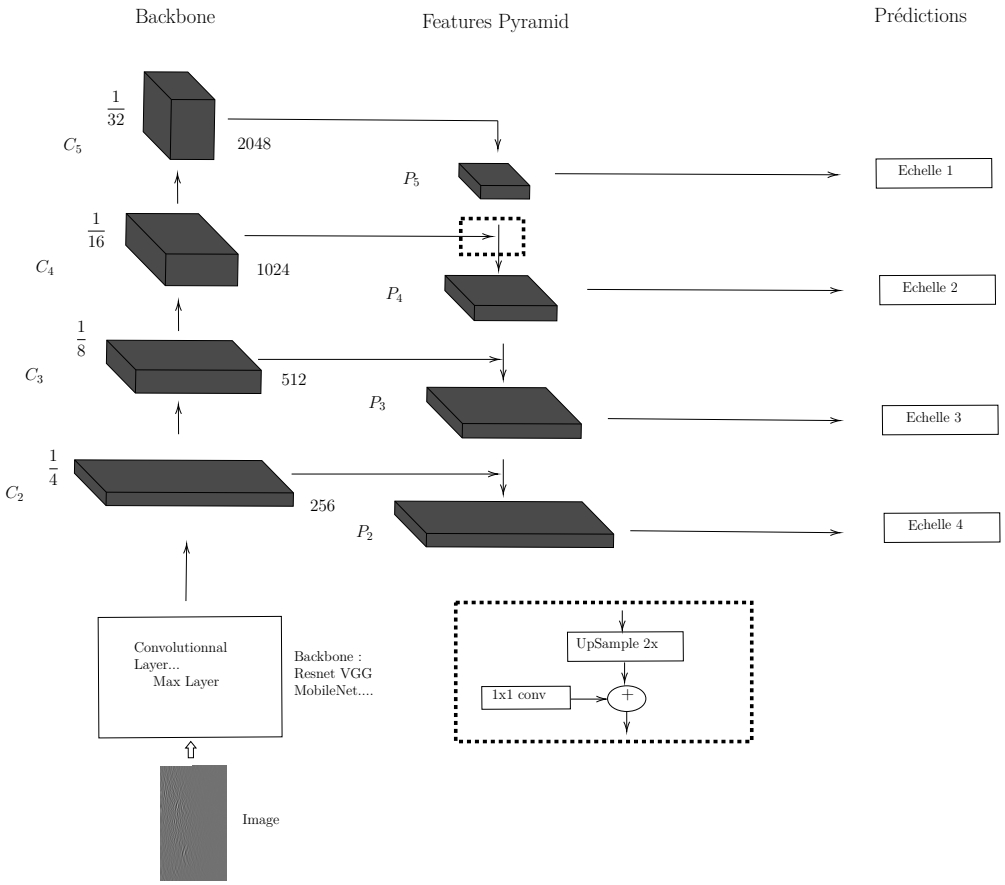


FIGURE 4.4 – Schéma du FPN avec le Backbone.

La Figure 4.4 illustre l'architecture globale du FPN utilisée dans notre modèle. Cette fusion produit des cartes enrichies à chaque niveau de la pyramide. Chaque sortie (P2 à P5) présente une profondeur de 256 canaux, avec des résolutions respectives : [N, 256, H/4, W/4] pour P2, [N, 256, H/8, W/8] pour P3, [N, 256, H/16, W/16] pour P4, et [N, 256, H/32, W/32] pour P5. Une carte supplémentaire P6, de taille [N, 256, H/64, W/64], peut être ajoutée pour capturer les objets de plus grande taille.

4. La tête (Head) : RPN et RoI Head

Dans une architecture de détection en deux étapes comme **Faster R-CNN**, la **tête** joue un rôle central dans la tâche de détection d'objets. Il est composé de deux sous-modules complémentaires :

- le **RPN (Region Proposal Network)**, qui génère des propositions de régions susceptibles de contenir des objets ;
- le **RoI Head (Region of Interest Head)**, qui affine ces propositions pour effectuer la classification et la régression finale des boîtes englobantes.

Cette décomposition en deux étapes permet d'abord d'identifier les régions potentiellement pertinentes dans l'image, puis de les analyser de manière plus fine, ce qui améliore la précision globale de la détection.

Nous allons maintenant détailler le fonctionnement de chacun de ces modules, en commençant par le **RPN**.

Le **RPN** est un composant clé de Faster R-CNN. Il a pour objectif de générer automatiquement un ensemble restreint de régions d'intérêt susceptibles de contenir un objet. Cette approche remplace les

méthodes classiques de recherche sélective (e.g., *Selective Search*) par une solution entièrement apprise et intégrée au réseau.

Le RPN prend en entrée les cartes de caractéristiques multi-échelles issues du **Feature Pyramid Network (FPN)** (par exemple P_2, P_3, P_4, P_5). Sur chaque niveau, un mécanisme de glissement d'une fenêtre (sliding window) permet d'analyser localement les régions de l'image en utilisant des *boîtes d'ancrage* (anchors) [1] , qui sont des boîtes englobantes de tailles et rapports d'aspect fixes, centrées sur chaque position de la carte.

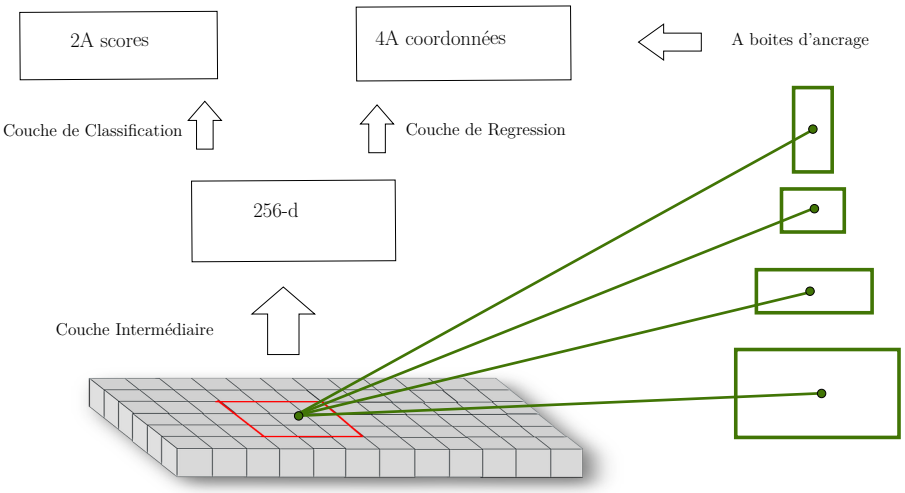


FIGURE 4.5 – Schéma Boite d'ancrage.

Architecture du RPN Le RPN est un petit réseau convolutionnel appliqué de manière glissante sur les cartes de caractéristiques. Il est composé de :

- Une couche de convolution 3×3 avec 512 canaux.
- Deux branches parallèles :
  - **Tête de classification (objectness)** : une couche de convolution 1×1 qui produit $2A$ scores, soit deux scores (objet / non-objet) par boîte d'ancrage.
  - **Tête de régression de boîtes** : une autre convolution 1×1 qui produit $4A$ valeurs correspondant aux décalages $(\Delta x, \Delta y, \Delta w, \Delta h)$ pour chaque boîte d'ancrage.
    Chaque boîte d'ancrage est représentée par son centre (x_a, y_a), sa largeur w_a et sa hauteur h_a. La boîte de vérité associée (ground truth) est notée (x^*, y^*, w^*, h^*), avec les mêmes conventions. Le RPN apprend à prédire les décalages cibles suivants :

$$\Delta x = \frac{x^* - x_a}{w_a}, \quad \Delta y = \frac{y^* - y_a}{h_a}, \quad \Delta w = \log\left(\frac{w^*}{w_a}\right), \quad \Delta h = \log\left(\frac{h^*}{h_a}\right)$$

    Ces formules expriment comment la boîte cible diffère de l'ancre initiale : les translations sont normalisées par les dimensions de l'ancre, tandis que les variations de taille sont exprimées de manière logarithmique. Le RPN apprend à prédire ces valeurs à partir des caractéristiques locales extraites de l'image, afin de générer des propositions d'objets (proposals) proches des vérités terrain.

1. **Boîtes d'ancrage :** Une ancre (ou boîte d'ancrage) est représentée par un rectangle défini par les coordonnées de son centre (x_a, y_a), sa largeur w_a, et sa hauteur h_a. Pour chaque position spatiale de la carte de caractéristiques, on génère A ancres (généralement $A = 3$ ou plus) de différentes tailles et rapports d'aspect, afin de couvrir une large variété de formes d'objets. Ces ancres servent de boîtes candidates initiales que le RPN ajuste pour proposer des régions d'intérêt.

Forme des tenseurs de sortie Soit N la taille du batch, A le nombre de boîtes d'ancrage par position, et $H \times W$ les dimensions spatiales de la carte de caractéristiques. Les sorties du RPN ont la forme suivante :

- **Classification (objectness logits)** : un tenseur de taille $[N, A, H, W]$, où chaque valeur indique si une boîte d'ancrage contient un objet ou non.
- **Régression (bbox regression)** : un tenseur de taille $[N, A \times 4, H, W]$ contenant les décalages $(\Delta x, \Delta y, \Delta w, \Delta h)$ pour chaque boîte d'ancrage.

Multi-niveau avec FPN Lorsque le RPN est utilisé avec une pyramide de caractéristiques (FPN), il est appliqué à chaque niveau (P_2 à P_5),. Les prédictions sont ensuite **concaténées** à tous les niveaux pour former un ensemble global de propositions.

Filtrage et sélection Le RPN génère en général plus de 100 000 propositions initiales. Pour réduire ce nombre :

- Un **Non-Maximum Suppression (NMS)** est appliqué pour éliminer les propositions de boîte redondantes engendrées par le RPN ou les boîtes prédictives finales après classification et régression.
- Seules les top-k régions les plus probables (souvent $k = 1000$) sont conservées et transmises au module suivant.

Entraînement Le RPN est entraîné conjointement avec le reste du réseau via une fonction de perte combinée :

- **Perte de classification** (entropie croisée binaire).
- **Perte de régression** (Smooth L1 Loss) pour les boîtes positives.

Les ancres sont étiquetées comme positives si elles ont une IoU supérieure à 0.7 avec une boîte de vérité, négatives si l'IoU est inférieure à 0.3, et ignorées sinon.

Résumé Le RPN permet ainsi une génération rapide, dense et apprise de propositions d'objets, adaptée aux caractéristiques extraites par le backbone. Il constitue un pont crucial entre la détection globale et la classification fine des objets via le **RoI Head**.

Le **RoI Head** constitue la seconde composante du head dans Faster R-CNN. Il intervient après que le RPN a proposé des régions d'intérêt (ou *proposals*) dans l'image. Son rôle est double :

- **Classification** : déterminer si chaque région contient un objet, et si oui, prédire sa classe.
- **Régression de boîtes** : ajuster précisément les coordonnées des boîtes proposées pour mieux englober les objets détectés.

Les régions proposées sont projetées sur les cartes de caractéristiques extraites par le backbone (et éventuellement enrichies via le FPN) à l'aide d'une opération qui convertit chaque RoI en une représentation de taille fixe. Cette opération est essentielle pour pouvoir appliquer les couches entièrement connectées de classification et de régression ; ainsi, pour chaque boîte d'ancrage proposée par le RPN, on choisit le niveau de carte le plus approprié en fonction de sa taille.

Deux variantes existent, en partant de la carte de caractéristiques provenant du backbone, ou des niveaux P_2 à P_5 du FPN, ainsi que des régions d'intérêt K proposées par le RPN. Chaque coordonnée du RoI est représentée par un tenseur de dimension $[N_i, x_1, y_1, x_2, y_2]$, où N_i est l'indice du batch correspondant à cette RoI :

- **RoI Pooling** : utilisée dans **Fast R-CNN**, cette opération convertit chaque RoI en une grille fixe (par exemple 7×7) en divisant la région en cellules, puis en appliquant un max-pooling dans chaque cellule. Cependant, elle introduit des erreurs dues à la quantification des coordonnées entières (arrondis), ce qui peut nuire à la précision.

- **RoI Align** : introduite dans **Faster R-CNN**, elle remplace RoI Pooling pour résoudre ce problème. RoI Align supprime l'arrondi en utilisant une *interpolation bilinéaire* fine, qui est redimensionnée à une taille fixe ici 7×7 pour extraire les valeurs exactes à des positions fractionnaires dans les cartes de caractéristiques. En sortie, nous obtenons un tenseur de forme $[K, 256, 7, 7]$, où K est le nombre de RoIs et 256 le nombre de canaux.

Ces régions normalisées sont ensuite passées à travers des couches entièrement connectées qui réalisent :

- une **classification multiclasse** (incluant la classe "fond") ;
- une **régression de boîtes englobantes** spécifique à chaque classe.

Cette étape permet de raffiner les propositions du RPN afin d'obtenir les prédictions finales.

Après le RoI Align, la classification finale et la régression des boîtes sont effectuées à l'aide de couches linéaires (ou couches pleinement connectées).

Classification finale L'entrée de cette étape est un tenseur de forme $[K, 256, 7, 7]$, en provenance du RoI Align. Ce tenseur est aplati, puis passé dans une couche linéaire pour produire une sortie de dimension $[K, num\_classes]$. Chaque ligne correspond aux scores de classification d'une région d'intérêt (RoI) parmi les différentes classes.

Régression des boîtes finales Comme pour la classification, l'entrée est également le tenseur $[K, 256, 7, 7]$ en sortie du RoI Align. Ce tenseur est transformé via une autre couche linéaire pour produire un tenseur de sortie de forme $[K, num\_classes \times 4]$, représentant les décalages $(\Delta x, \Delta y, \Delta w, \Delta h)$ pour chaque classe. Ces décalages permettent d'ajuster les boîtes englobantes initiales pour chaque RoI.

NMS après classification finale

Un second NMS est appliqué aux boîtes de détection finales pour garantir que seules les meilleures prédictions par classe sont conservées et que les autres sont supprimées.

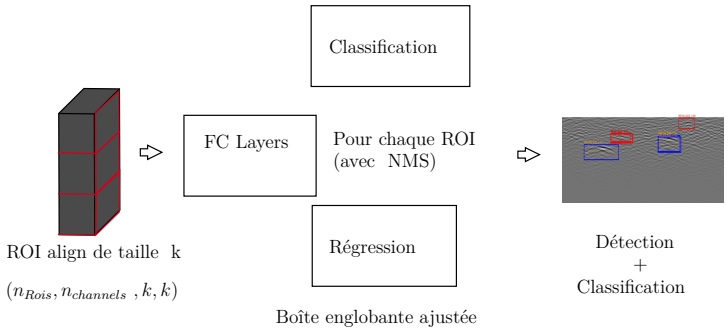


FIGURE 4.6 – Schéma du RoI head.

Conclusion

Le réseau Faster R-CNN ne s'adapte pas dynamiquement à chaque image pour ne prédire que les classes présentes ; au contraire, il génère des scores pour toutes les classes potentielles (y compris le fond) pour chaque RoI, indépendamment du nombre d'objets ou de classes dans l'image. Ensuite, ces scores sont interprétés pour déterminer les classes réellement présentes dans chaque image.

- Faster R-CNN a introduit une architecture entièrement intégrée pour la détection d'objets, éliminant ainsi le besoin d'un algorithme externe de propositions de régions.

- Il propose un module appelé Réseau de propositions de régions (Region Proposal Network, RPN) qui génère les propositions de régions de manière entièrement différentiable et intégrée dans le réseau. Ces régions sont par la suite filtrées par NMS (Non-Maximum Suppression) avec un seuil (threshold) de 0.7 par exemple. NMS est une méthode qui permet de passer au crible les régions proposées et de choisir uniquement celles qui sont intéressantes.
- Le RPN partage la plupart des couches convolutionnelles avec le réseau de détection d'objets, ce qui le rend très efficace et permet un entraînement de bout en bout (end-to-end).
- Les régions regroupées passent ensuite par des couches entièrement connectées pour la prédiction des coordonnées des zones des objets et des classes de sortie
- Avec Faster R-CNN, le processus de détection d'objets devient plus rapide et plus précis, car il élimine le besoin de générer des propositions de régions en amont et fusionne le processus de génération de propositions avec le réseau de détection lui-même.

4.1.3 Fonctions de perte dans Faster R-CNN

Le réseau Faster R-CNN est entraîné de manière conjointe, en ajustant l'ensemble de ses composants : le backbone, le RPN et les têtes de détection, grâce à la rétropropagation. Cette approche permet une optimisation globale des performances en affinant simultanément la génération de propositions et la détection d'objets. Dans Faster R-CNN, la fonction de perte globale L_{total} est composée de deux termes principaux : la perte associée au réseau de propositions (RPN) et celle liée à la tête de détection (RoI head).

- *Perte du RPN* :

(4.1)

où :

  - p_i est la probabilité prédite que l'ancre i corresponde à un objet.
  - $p_i^* \in \{0, 1\}$ est l'étiquette de vérité terrain : 1 si l'ancre est positive, 0 sinon.
  - t_i et t_i^* désignent respectivement les décalages de régression prédits et cibles (vérité terrain) pour l'ancre i.
  - N_{cls} est le nombre total d'ancres dans le mini-lot, et N_{reg} est le nombre d'ancres positives.
  - λ est un coefficient d'équilibrage entre les pertes.
  - L_{cls} est une perte d'entropie croisée binaire.
  - L_{reg} est une perte de type *smooth L1*.

- *Perte de la tête RoI* :

$$L_{\text{RoI}}(p_i, t_i) = \frac{1}{N} \sum_i \left[L_{\text{cls}}(p_i, p_i^*) + \lambda \cdot \mathbf{1}_{[p_i^* > 0]} L_{\text{reg}}(t_i, t_i^*) \right] \tag{4.2}$$

où :

  - p_i est la distribution prédite sur $K + 1$ classes (fond + K objets).
  - p_i^* est la classe cible (encodée en one-hot).
  - t_i et t_i^* sont les décalages de régression associés à la classe p_i^*.
  - N est le nombre de régions d'intérêt (RoIs) dans le mini-lot.
  - La régression est uniquement appliquée aux régions positives ($p_i^* > 0$).

La somme de ces deux termes permet donc de guider l'optimisation globale du modèle :

$$L_{\text{total}} = L_{\text{RPN}} + L_{\text{RoI}}$$

4.2 Adaptation de Faster R-CNN à CNN-SPDNet

Dans cette section, nous présentons une adaptation de Faster R-CNN au domaine du radar à pénétration de sol (GPR), en intégrant des représentations basées sur la covariance issues des RoIs extraites. L'objectif est d'exploiter des **statistiques d'ordre 2** (covariances), plus riches que les moyennes classiques (statistiques d'ordre 1) ou les maximums, afin de mieux modéliser les relations entre les canaux dans les données radar, souvent bruitées et complexes.

L'architecture Faster R-CNN est particulièrement bien adaptée à cette approche en raison de l'étape **RoI Align**, qui permet d'extraire des régions d'intérêt sous forme de tenseurs bien définis. À partir de ces RoIs, nous proposons d'appliquer une transformation de type **MPN-COV**, produisant pour chaque RoI une matrice de covariance, puis d'exploiter ces matrices dans une architecture compatible avec la géométrie des matrices SPD comme SPDNet.

Nous comparons cette approche au Faster R-CNN classique afin d'évaluer les bénéfices liés à l'intégration explicite des statistiques d'ordre 2 dans le pipeline de détection.

4.2.1 RC R-CNN (Residual Covariance R-CNN)

L'objectif de cette approche est d'exploiter les **statistiques d'ordre 2** issues des régions d'intérêt extraites par Faster R-CNN afin d'améliorer la représentation des objets détectés dans les images GPR. Contrairement aux descripteurs classiques basés sur des moyennes (statistiques d'ordre 1), les matrices de covariance permettent de modéliser les relations entre canaux (corrélations inter-features), ce qui s'avère particulièrement pertinent pour des données complexes ou bruitées, comme les images radar.

MPN-COV [13] MPN-COV (Matrix Power Normalized COVariance pooling) est une méthode de second ordre qui extrait des statistiques de covariance à partir d'un tenseur de caractéristiques issu d'un CNN, associé à une régularisation sur les valeurs propres pour un meilleur conditionnement de l'estimé en grande dimension. Dans notre architecture, cette opération est appliquée **après l'empilement des RoIs** extraits via la couche `RoI Align`. Le tenseur obtenu est de forme :

$$\mathcal{T} \in \mathbb{R}^{n\times d\times h\times w},$$

où $n = 1024$ est le nombre de RoIs, $d = 256$ le nombre de canaux, et $h = w = 7$ les dimensions spatiales. Chaque RoI est un tenseur $\mathcal{T}_{R_a} \in \mathbb{R}^{d\times h\times w}$, qui est ensuite aplati spatialement pour former une matrice $\mathbf{T}_{R_a} \in \mathbb{R}^{d\times M}$, avec $M = h \cdot w = 49$. Le batch complet pour les $n = 1024$ RoIs est ainsi représenté par :

$$\mathbf{T} \in \mathbb{R}^{n\times d\times M}.$$

Comme dans le chapitre 3 avec la couche de pooling de covariance 3.1.2, MPN-COV commence par calculer, pour chaque RoI, une matrice de covariance centrée :

$$C_{R_a} = \mathbf{T}_{R_a}\,\bar{\mathbf{I}}\,\mathbf{T}_{R_a}^{\mathrm{T}},$$

où la matrice de centrage est définie par :

$$\bar{\mathbf{I}} = \frac{1}{M}\left(\mathbf{I}_M - \frac{1}{M}\mathbf{1}_M\mathbf{1}_M^{\mathrm{T}}\right),$$

et $\mathbf{1}_M \in \mathbb{R}^{M\times 1}$ est un vecteur colonne de 1. La matrice $C_{R_a} \in \mathbb{R}^{256\times 256}$ capture les corrélations entre les canaux, offrant une représentation plus riche que les agrégations de premier ordre classiques.

Dans la version originale de MPN-COV, chaque matrice C_{R_a} est soumise à une décomposition en valeurs propres :

$$C_{R_a} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{T}},$$

suivie d'une régularisation par une puissance sur les valeurs propres :

$$S_{R_a} = \mathbf{U}\mathbf{\Lambda}^{\alpha}\mathbf{U}^{\mathrm{T}},$$

où $\alpha > 0$ est un hyperparamètre de régularisation. Typiquement, $\alpha = 0.5$, ce qui revient à appliquer la racine carrée matricielle.

Les détails de la rétropropagation dans MPN-COV sont décrits dans les travaux [13, 14], eux-mêmes fondés sur les dérivations mathématiques de [68].

Pour réduire le coût de calcul de la SVD, la variante **Fast MPN-COV** [14] utilise une approximation de la racine carrée matricielle par l'algorithme de Newton-Schulz. Cette opération sera représentée par une couche de notre modèle que nous nommerons *SqrtmLayer*. Cette couche est appliquée sur un tenseur $\mathbf{C} \in \mathbb{R}^{n\times d\times d}$, contenant les matrices C_{R_a} pour chaque RoI, et s'effectue par les étapes suivantes :

- **Normalisation** : chaque matrice est divisée par sa trace :

$$\mathbf{A}_{R_a} = \frac{C_{R_a}}{\mathrm{Tr}(C_{R_a})}.$$

- **Initialisation** :

$$\mathbf{Y}_0 = \mathbf{A}_{R_a}, \quad \mathbf{Z}_0 = \mathbf{I}_d.$$

- **Itérations de Newton-Schulz** :

$$\mathbf{Y}_{k+1} = \frac{1}{2}\mathbf{Y}_k(3\mathbf{I} - \mathbf{Z}_k\mathbf{Y}_k),$$
$$\mathbf{Z}_{k+1} = \frac{1}{2}(3\mathbf{I} - \mathbf{Z}_k\mathbf{Y}_k)\mathbf{Z}_k.$$

- **Sortie finale** :

$$S_{R_a} = \sqrt{\mathrm{Tr}(C_{R_a})} \cdot \mathbf{Y}_N.$$

Le résultat est un tenseur $\mathbf{S} \in \mathbb{R}^{n\times d\times d}$, où chaque $\S_{R_a}$ est une matrice symétrique définie positive (SPD).

À ce stade, nous souhaitons procéder à la classification de ce tenseur. Pour ce faire, nous adoptons la même approche que celle présentée dans le chapitre 3. Autrement dit, le modèle **SPDNet** est appliqué aux matrices $\S_{R_a}$ afin d'en réduire la dimension et de réaliser la classification finale dans un espace euclidien. Afin de limiter la complexité de traitement dans SPDNet, une unique étape de réduction dimensionnelle est effectuée, faisant passer les matrices de 256×256 à 32×32.

Branche de régression La régression des boîtes englobantes est effectuée classiquement à partir des sorties `RoI Align` ($n \times 256 \times 7 \times 7$), sans passer par les couches de covariance. Cette séparation permet de bénéficier des statistiques d'ordre 2 uniquement pour la branche classification.

La Figure 4.7 illustre l'architecture du modèle.

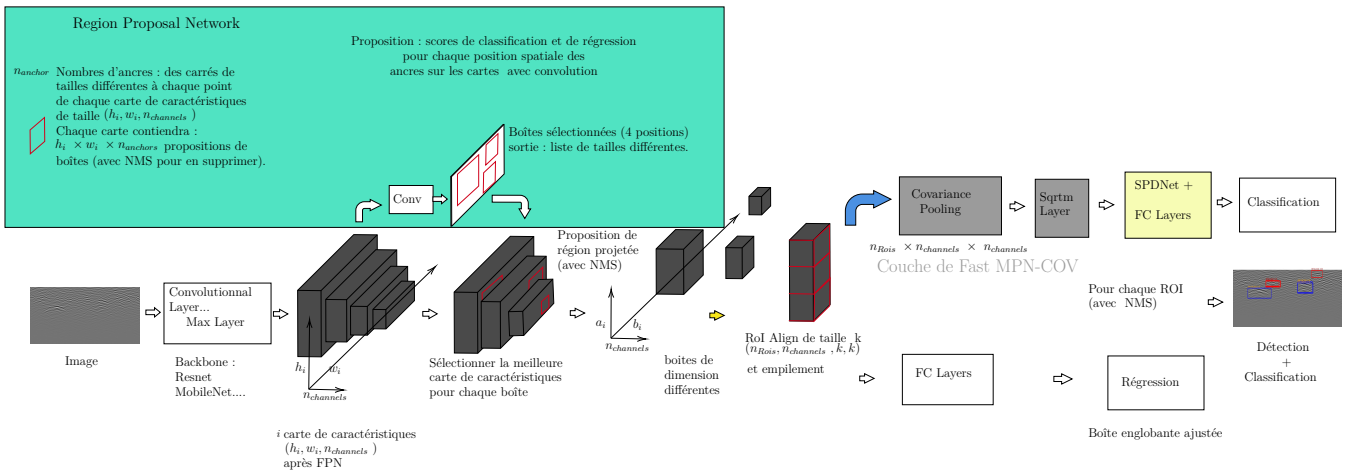


FIGURE 4.7 – Schéma de **RC R-CNN** avec application de la covariance après `RoI Align`.

Avantages et inconvénients Dans cette configuration, l'ensemble des RoIs est redimensionné à une taille fixe (7×7) à l'aide de `RoI Align`, puis empilé dans un tenseur de forme $(N \times 256 \times 7 \times 7)$. On applique ensuite successivement `CovPool` puis `Sqrtm`, ce qui produit, pour chaque RoI, une matrice de taille 256×256.

Avantages :

- **Stabilité numérique :** Le traitement par lot ($N = 1024$) améliore l'estimation des statistiques d'ordre 2 malgré le faible nombre d'observations par RoI (49), atténuant l'impact du bruit GPR.
- **Traitement GPU efficace :** La couche `SqrtmLayer` (Fast MPN-COV) permet une approximation différentiable et vectorisée de la racine de la matrice de covariance, adaptée à une exécution accélérée sur GPU.
- **Intégration directe dans SPDNet :** Chaque RoI est représentée par une matrice SPD, compatible avec les couches géométriques (ReEig, LogEig), sans adaptation particulière.

Inconvénients :

- **Perte de structure locale :** Le redimensionnement fixe peut écraser des détails fins propres à la géométrie initiale des RoIs.
- **Limitation adaptative :** Toutes les RoIs sont traitées de manière uniforme, ce qui peut nuire à la modélisation de formes très variées.
- **Coût élevé :** Malgré le traitement par lot (batch), les opérations `ReEig` et `LogEig` restent coûteuses ($O(d^3)$) sur des matrices 256×256, ce qui ralentit l'entraînement.

En résumé, cette approche constitue un compromis pertinent entre l'expressivité statistique, la compatibilité avec les couches SPDNet et l'efficacité d'exécution via Fast MPN-COV, au prix d'un certain coût computationnel sur les matrices de covariance hautes dimensionnelles. Nous évaluons désormais dans quelle mesure cette représentation enrichie améliore les performances de détection, en la comparant au Faster R-CNN classique sur des données GPR réalistes.

4.3 Simulations et Résultats

Dans cette section, nous évaluons les performances de deux modèles : le **Faster R-CNN** classique et notre modèle **RC R-CNN**. Le **backbone** choisi pour les deux modèles est le **ResNet-34**, comme présenté dans le chapitre 2.

Ces deux modèles sont comparés afin de déterminer dans quelle mesure l'introduction d'une structure matricielle améliore la **capacité de détection des hyperboles**.

Les expérimentations sont menées sur la majorité du jeu de données, avec une attention particulière portée au **changement d'élévation du radar** et au **type de sol** dans les données de test. Cela permet de mesurer la robustesse de chaque approche dans des conditions réalistes et hétérogènes.

Le jeu de données Common Objects in Context (COCO) est une base d'images annotées riche, contenant des milliers d'images d'objets variés dans leur contexte naturel. Il est largement utilisé pour l'évaluation des performances des modèles de détection d'objets, de segmentation et de détection de points clés [69].

Les métriques initialement proposées avec ce jeu de données sont devenues la méthode d'évaluation de référence dans ces domaines. Afin de pouvoir les utiliser, nos résultats ont été adaptés au format COCO, ce qui nous a permis d'employer la bibliothèque *pycocotools*[2], un standard reconnu pour le calcul du mAP (mean Average Precision) et du mAR (mean Average Recall).

4.3.1 Prétraitement des données

L'adaptation du modèle aux données radar nécessite une étape préalable de **prétraitement**.

Chaque radargramme est ainsi annoté au format **COCO (Common Objects in Context)**, qui encode les boîtes englobantes des hyperboles à l'aide de coordonnées normalisées. Ce format facilite l'intégration avec les architectures de détection telles que Faster R-CNN (voir Annexe C).

Le jeu de données final comprend un total de **731 radargrammes**, répartis selon trois facteurs : la **fréquence** du radar (200 MHz ou 350 MHz), l'**élévation** (de 0 à 150 cm) et le **type de sol** (grave humide, grave sèche, sable humide, sable sec). Tous les radargrammes ont été redimensionnés à une taille de $\mathbf{h = 900}$ et $\mathbf{w = 4000}$, puis normalisés selon la même procédure que celle décrite en Section 1.7.2.

La répartition détaillée est présentée dans le tableau suivant :

	grave humide						grave sèche						sable sec						sable humide						**Total**
Fréquence / Élévation (cm)	0	25	50	75	100	150	0	25	50	75	100	150	0	25	50	75	100	150	0	25	50	75	100	150	
200 MHz	8	10	10	10	6	10	14	10	12	12	12	9	22	22	22	20	23	22	16	10	10	11	10	11	**322**
350 MHz	12	12	12	9	9	3	16	11	9	10	9	11	33	34	34	30	32	30	18	16	16	16	14	13	**409**
Total	20	22	22	19	15	13	30	21	21	22	21	20	55	56	56	50	55	52	34	26	26	27	24	24	**731**

La lecture du tableau révèle plusieurs observations importantes.

Tout d'abord, la distribution selon l'élévation est bien équilibrée. Chaque type de sol est représenté à toutes les hauteurs (de 0 à 150 cm), ce qui garantit une diversité géométrique dans les acquisitions radar. Cela permet d'entraîner un modèle plus **robuste aux changements d'altitude**, ce qui constitue un aspect essentiel en conditions réelles.

Sur le plan des fréquences, on constate que les données à **350 MHz** sont légèrement plus nombreuses (**409 images**) que celles à **200 MHz** (**322 images**).Ce déséquilibre modéré est bénéfique, chaque fréquence possédant des avantages spécifiques : le **200 MHz** permet une meilleure **pénétration dans le sol**, tandis que le **350 MHz** offre une **résolution plus fine**.

En ce qui concerne les types de sol, les **sables** (secs et humides) sont clairement majoritaires dans la base, représentant plus de **65 %** du total des images. Le **sable sec**, à lui seul, regroupe **324 radargrammes**, soit environ **44 %** du jeu. Cette sur-représentation reflète l'intérêt particulier porté à ces

2. *pycocotools,* qui est open source sur GitHub : `https://github.com/cocodataset/cocoapi/tree/master/PythonAPI/pycocotools`

milieux complexes, souvent rencontrés sur le terrain.

Il est toutefois important de noter que ce déséquilibre pourrait introduire un **biais** lors de l'apprentissage ou de l'évaluation si les métriques ne sont pas normalisées par classe de sol.

Malgré cela, la base de données reste globalement variée. Cette diversité constitue un avantage important pour les modèles testés, car elle favorise leur capacité à **généraliser à des situations variées et réalistes**.

4.3.2 Hyperparamètres de l'entraînement

L'entraînement des deux modèles **Faster R-CNN** et **RC R-CNN** sur les données GPR repose sur l'utilisation de l'optimiseur **SGD** (Stochastic Gradient Descent), combiné à un scheduler d'apprentissage de type `StepLR`. Plusieurs hyperparamètres influencent alors la performance, la stabilité et la convergence du réseau. Certains d'entre eux sont standards dans l'entraînement de réseaux de neurones convolutionnels, tandis que d'autres sont propres à la détection d'objets et au fonctionnement interne de Faster R-CNN, notamment ceux liés au RPN (Region Proposal Network) et au module RoI (Region of Interest).

Les principaux hyperparamètres utilisés sont détaillés ci-dessous :

- **lr (learning rate)** : taux d'apprentissage contrôlant la vitesse de mise à jour des poids. Une valeur trop élevée peut déstabiliser l'entraînement, tandis qu'une valeur trop faible ralentit la convergence.
- **momentum** : facteur d'inertie qui permet de lisser les oscillations du gradient et d'accélérer la convergence.
- **lr scheduler** : stratégie d'ajustement du taux d'apprentissage au cours de l'entraînement. Ici, un scheduler de type `StepLR` est utilisé.
- **batch size** : nombre d'exemples traités simultanément à chaque itération. Il affecte la stabilité de l'estimation du gradient.
- **num_workers** : nombre de processus parallèles pour le chargement des données. Sa valeur impacte le temps d'accès aux données.
- **rpn_nms** : seuil NMS (Non-Maximum Suppression) appliqué aux propositions générées par le RPN, contrôlant le degré de suppression des propositions redondantes.
- **rpn_score** : seuil minimal de score de confiance pour qu'une proposition du RPN soit retenue.
- **roi_nms** : seuil NMS appliqué aux boîtes sélectionnées après le RoI pooling.
- **roi_score** : score minimal requis pour qu'une prédiction soit conservée lors de la phase finale de détection.

Le tableau 4.1 ci-dessous résume les valeurs adoptées pour ces hyperparamètres :

Paramètre	Valeur	Description
Learning rate (lr)	1×10^{-2}	Vitesse de mise à jour des poids (SGD).
Momentum	0.9	Stabilise l'optimisation en cumulant les gradients précédents.
Batch size	2	Taille du lot d'entraînement.
lr scheduler	StepLR	Diminue le taux d'apprentissage toutes les 10 époques avec $\gamma = 0.1$.
num_workers	1	Un seul processus utilisé pour le chargement des données.
rpn_nms	0.7	Seuil NMS pour la suppression des propositions du RPN.
rpn_score	0	Seuil de confiance minimal pour les propositions du RPN.
roi_nms	0.5	Seuil NMS pour les boîtes candidates du RoI head.
roi_score	0.6	Score minimal pour qu'une boîte soit retenue après classification.

TABLE 4.1 – Hyperparamètres utilisés pour l'entraînement de Faster R-CNN sur les données GPR.

4.3.3 Métriques utilisées

Dans cette étude, nous utilisons une sélection de métriques issues de *pycocotools* pour évaluer la performance des modèles de détection d'objets. Ces métriques standardisées permettent une évaluation fine des performances, notamment en termes de précision et de rappel, tout en tenant compte de différents seuils d'Intersection over Union (IoU). Nous avons retenu les quatre métriques les plus représentatives :

- **mAP** (mean Average Precision) ou **mAP@0.50 :0.95** : précision moyenne calculée sur des seuils d'IoU allant de 0.50 à 0.95 avec un pas de 0.05. C'est la métrique principale utilisée pour évaluer la qualité globale du détecteur.
- **mAP@0.50** : précision moyenne à un seuil d'IoU fixé à 0.50, représentant une évaluation plus tolérante des détections.
- **mAP@0.75** : précision moyenne à un seuil d'IoU plus strict (0.75), permettant d'apprécier de manière plus précise la qualité de localisation des objets détectés.
- **mAR@100** (Average Recall) : rappel moyen maximal obtenu en autorisant jusqu'à 100 détections par image, mesurant la capacité du modèle à récupérer les objets présents.

Ces métriques sont reprises dans les tableaux comparatifs suivants, où elles permettent d'analyser le comportement des modèles selon différents niveaux d'exigence en précision et en rappel. Plus de détails sont encore donnés en annexe D.

4.3.4 Critère d'arrêt fondé sur la métrique mAP@0.50 :0.95

Dans le cadre de l'entraînement de notre modèle de détection, nous avons adopté une stratégie d'arrêt anticipé (*early stopping*) reposant sur la métrique **mAP@0.50 :0.95** que l'on notera par la suite : $\text{mAP}_{50:95}$, (*Average Precision* à un seuil d'IoU de 0.5 à 0.95). Cette métrique, utilisée dans le protocole d'évaluation COCO, évalue la précision moyenne des prédictions pour une tolérance modérée sur le recouvrement des boîtes. Elle constitue un indicateur robuste de la qualité des détections et permet d'évaluer les performances du modèle de manière cohérente avec les objectifs finaux.

Afin de suivre efficacement cette métrique tout au long de l'apprentissage, nous avons utilisé la quantité complémentaire $1 - \text{mAP}_{50:95}$ comme fonction de suivi sur l'ensemble de validation. Cette démarche permet d'interpréter la performance du modèle de manière analogue à une fonction de coût : plus le score est élevé, plus le modèle est jugé sous-performant. Ce choix s'inspire notamment du travail de [70], qui

propose une formulation différentiable de l'*Average Precision*, appelée *AP-Loss*, dans le but d'aligner les objectifs d'optimisation avec les métriques d'évaluation.

Concrètement, l'entraînement est interrompu si aucune amélioration du score $\text{mAP}_{50:95}$ n'est observée pendant **cinq époques consécutives** (**patience = 5**). Cette stratégie permet de limiter le surapprentissage tout en garantissant une convergence vers un optimum pertinent selon la métrique de référence. Le meilleur modèle obtenu selon ce critère est ensuite conservé pour l'évaluation finale.

4.3.5 Étude sur une partie du dataset

Répartition des données

L'ensemble de données utilisé dans cette étude comprend un total de 570 radargrammes, acquis à l'aide d'antennes de fréquences 200 MHz et 350 MHz, à différentes hauteurs d'élévation allant de 0 à 150 cm. Ces données ont été collectées sur quatre types de sols représentatifs des conditions rencontrées lors des campagnes expérimentales : *grave humide*, *grave sèche*, *sable sec*, et *sable humide*.

Le tableau 4.2 présente la répartition des radargrammes en fonction de la fréquence d'acquisition (200 MHz et 350 MHz), de l'élévation du radar (de 0 à 150 cm) et du type de sol (grave humide, grave sèche, sable sec et sable humide). Cette structuration fine permet d'assurer une couverture représentative et variée des cas possibles.

	grave humide						grave sèche						sable sec						sable humide						**Total**
Fréquence / Élévation (cm)	0	25	50	75	100	150	0	25	50	75	100	150	0	25	50	75	100	150	0	25	50	75	100	150	
200 MHz	5	8	9	10	5	8	7	7	9	11	11	8	16	17	16	16	18	20	11	7	6	8	8	9	**250**
350 MHz	9	11	11	6	6	2	13	8	6	7	8	11	20	27	25	25	26	24	14	10	13	12	13	13	**320**
Total	14	19	20	16	11	10	20	15	15	18	19	19	36	44	41	41	44	44	25	17	19	20	21	22	**570**

TABLE 4.2 – Distribution des radargrammes de notre base de données pour chaque élévation du radar en fonction de la fréquence de l'antenne et du type de sol.

Chaque hauteur (0, 25, 50, 75, 100 et 150 cm) est renseignée pour chacun des quatre types de sol, avec environ 95 échantillons par élévation, assurant ainsi une diversité verticale. Cette répartition permet au modèle de s'adapter à différentes configurations de géométrie d'acquisition.

Les deux fréquences utilisées offrent un compromis entre profondeur et résolution : la fréquence de 350 MHz, plus représentée (320 échantillons), apporte une meilleure résolution, tandis que la fréquence de 200 MHz (250 échantillons) permet une meilleure pénétration.

Le sable, qu'il soit sec ou humide, est le type de sol le plus représenté. Ce choix peut s'expliquer par la volonté de mieux caractériser des milieux dont la réponse aux ondes radar est plus complexe. Toutefois, cette surreprésentation pourrait introduire un biais si elle n'est pas prise en compte dans l'évaluation des performances du modèle.

Enfin, malgré quelques déséquilibres ponctuels dans certaines combinaisons fréquence/sol/élévation, l'ensemble de la base reste suffisamment équilibré pour permettre un apprentissage robuste. Cette diversité constitue un atout majeur pour la généralisation du modèle.

La base de données complète contient un total de **570 images annotées**, réparties comme suit : **456 images pour l'entraînement**, **57 images pour la validation** et **57 images pour le test**. Comme l'illustre le Tableau 4.3, la classe **Metallique** prédomine en nombre d'annotations dans chaque sous-ensemble, tandis que les classes **Abris en bois** et **Non métallique** sont représentées de manière comparable. Cette répartition relativement équilibrée assure une exposition adéquate du modèle à chaque type d'objet durant l'apprentissage et permet une évaluation juste sur les jeux de validation et de test.

Jeu de données	Metallique	Abris en bois	Non métallique
Entraînement (456 images)	781	570	560
Validation (57 images)	106	74	66
Test (57 images)	117	67	69

TABLE 4.3 – Répartition des annotations par classe pour chaque jeu de données.

Nous allons désormais entraîner nos deux modèles sur ce jeu de données.

Les modèles sont évalués en fonction de leur **critère d'arrêt**, défini ici comme $1 - \text{mAP}_{50:95}$, sur les données de validation. Ce critère nous permet de déterminer à quel moment l'amélioration des performances devient négligeable et d'arrêter l'entraînement pour éviter tout sur-apprentissage. Nous observerons également le comportement de la **loss totale** au cours des itérations et analyserons les résultats des métriques de *pycocotools* sur les données de test.

En outre, des exemples de prédictions en image seront présentés, permettant d'illustrer concrètement les performances des deux modèles. Ainsi, dans la suite, nous examinerons de manière détaillée les résultats obtenus pour les deux configurations, en nous concentrant sur leur efficacité à détecter les objets sur le jeu de test, et sur l'impact des améliorations introduites par le module SPDNet et la covariance.

Les sections suivantes détailleront les performances des deux modèles, en mettant en lumière les différences notables en termes de précision et de robustesse.

Suivi du critère d'arrêt $1 - \text{mAP}_{50:95}$ sur les données de validation

À la fin de l'entraînement de nos deux modèles, **Faster R-CNN** et **RC R-CNN**, le critère de suivi utilisé pendant l'apprentissage, $1 - \text{mAP}_{50:95}$, mesuré sur l'ensemble de validation, nous permet de suivre la progression de la précision moyenne pour des seuils d'IoU allant de 0.5 à 0.95, tout en étant interprété comme une fonction de coût sur les données de validation.

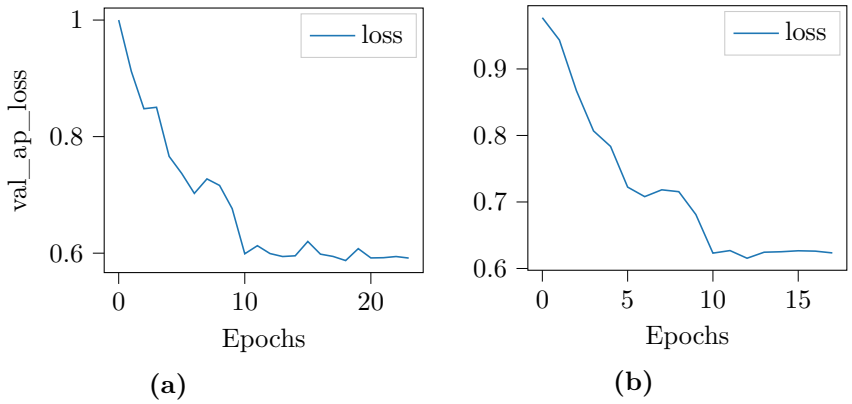


FIGURE 4.8 – Évolution du critère $1 - \text{mAP}_{50:95}$ sur les données de validation pendant l'entraînement : (a) Évolution du critère pour **Faster R-CNN** et (b) Évolution du critère pour **RC R-CNN**.

La Figure 4.8 montre l'évolution de ce critère pour les deux modèles. Le graphique de gauche correspond à **Faster R-CNN**, dont l'entraînement s'est arrêté après **23 époques**, avec une valeur finale de $1 - \text{mAP}_{50:95} = 0.408$, soit un $\text{mAP}_{50:95}$ de **0.592**. Le graphique de droite illustre la courbe pour **RC**

R-CNN, stoppé après **17 époques**, avec une valeur finale de $1 - \text{mAP}_{50:95} = 0.623$, soit un $\text{mAP}_{50:95}$ de **0.377**.

Ces résultats confirment la bonne convergence des deux modèles et montrent que le critère $\text{mAP}_{50:95}$ est pertinent pour suivre la qualité des détections pendant l'apprentissage. On note également que **RC R-CNN** atteint sa convergence en un nombre d'époques légèrement inférieur, ce qui suggère un apprentissage plus rapide sans préjuger pour l'instant des performances finales sur les données de test.

Loss totale et convergence globale

Nous présentons maintenant l'évolution de la *loss* totale pour chacun des deux modèles. Ces courbes agrègent l'ensemble des composantes internes du réseau (RPN et RoI) et permettent de suivre la dynamique globale d'apprentissage.

La Figure 4.9 illustre l'évolution de la loss totale pendant l'entraînement pour les deux modèles. Ces courbes agrègent les différentes composantes internes du réseau (issues du RPN et du RoI head) et permettent de visualiser la dynamique d'optimisation.

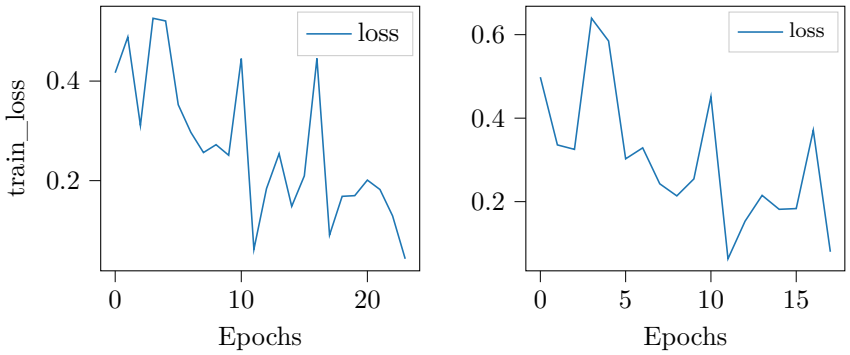


FIGURE 4.9 – Évolution de la loss totale pendant l'entraînement : (a) Pour **Faster R-CNN**, la perte diminue jusqu'à l'époque 17 avant de présenter de légères variations ; (b) Pour **RC R-CNN**, le même comportement est observé avec un ralentissement plus précoce vers l'époque 15.

Le graphique (a) de gauche correspond à **Faster R-CNN**, dont la perte fluctue fortement tout au long de l'entraînement, avec une chute nette vers la fin (époque 23). Le graphique (b) de droite montre un comportement similaire pour **RC R-CNN**, avec également des variations importantes et une baisse marquée juste avant l'arrêt à l'époque 17.

Il convient de souligner que ces courbes n'ont pas été utilisées pour déterminer le moment d'arrêt de l'entraînement. Le critère retenu repose exclusivement sur la performance obtenue sur le jeu de validation, à savoir la métrique $1 - \text{mAP}_{50:95}$. Les courbes de *loss* sont donc présentées ici à titre informatif : elles indiquent que, malgré des oscillations marquées, l'optimisation demeure globalement descendante.

Résultats des métriques sur les données de test

Pour évaluer l'impact global de l'ajout du bloc SPDNet sur les performances du détecteur, nous comparons ici encore **Faster R-CNN** et **RC R-CNN** en utilisant les métriques standards du protocole COCO. L'objectif est de mesurer non seulement l'amélioration de la précision moyenne (AP), mais également l'évolution de la capacité de localisation fine à travers différents seuils d'IoU.

Le tableau 4.4 présente les résultats obtenus sur les données de test. Pour **Faster R-CNN** et **RC R-CNN**.

Métrique	IoU	Area	MaxDets	Faster R-CNN	RC R-CNN
mAP	0.50 :0.95	all	100	0.480	0.488
mAP	**0.50**	**all**	**100**	**0.809**	**0.812**
mAP	0.75	all	100	0.489	0.522
mAR	0.50 :0.95	all	100	0.556	0.555

TABLE 4.4 – Comparaison des performances COCO des modèles Faster R-CNN et **RC R-CNN** sur le jeu de données *Full*.

Les résultats mettent en évidence une amélioration systématique, bien que modeste, des performances lorsque le modèle est enrichi avec SPDNet. Le gain le plus notable est observé sur la métrique mAP@75, qui passe de **0.489** à **0.522**, suggérant une amélioration de la précision de localisation, c'est-à-dire une meilleure capacité à produire des détections très bien alignées avec les objets réels.

La métrique mAP@50 reste globalement stable, avec une légère augmentation de **0.809** à **0.812**, tandis que la précision moyenne plus stricte (mAP@[0.50 :0.95]) progresse également légèrement. Enfin, la mesure du rappel moyen (mAR) reste quasiment inchangée.

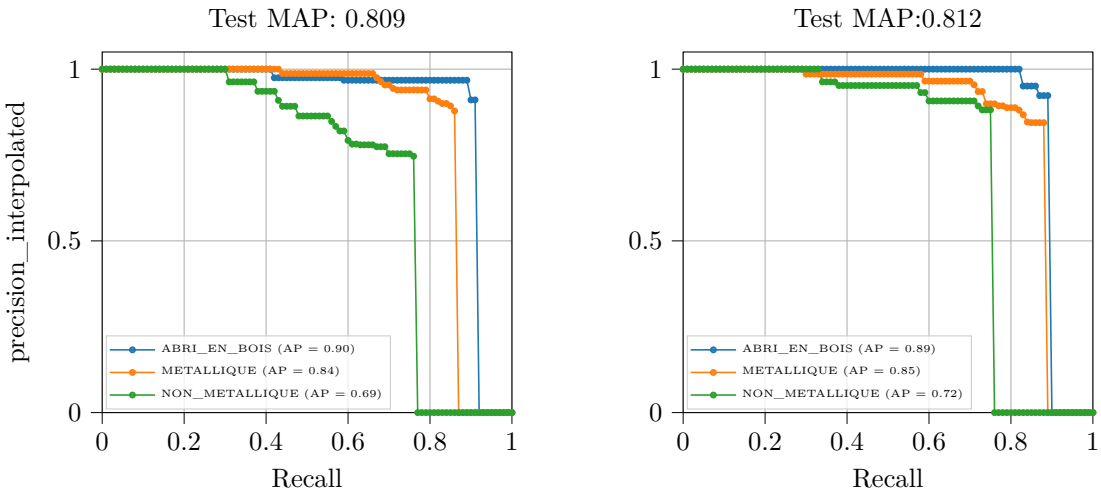
En conclusion, ces résultats valident l'hypothèse selon laquelle l'ajout d'une opération de normalisation de second ordre (SPDNet) améliore la qualité des détections produites par le modèle, en particulier lorsqu'un haut degré de précision est requis. Cette amélioration est d'autant plus significative que les objets sont souvent de grande taille dans le jeu de données et que le modèle doit différencier des structures visuellement proches. Cela montre que l'introduction d'opérations statistiques sur les représentations internes peut apporter un gain sans complexifier excessivement le modèle.

Courbe de précision-rappel interpolée

Dans le but de détailler les performances des modèles à un seuil fixe d'IoU, nous présentons ci-dessous les courbes de précision-rappel interpolées pour les trois classes principales, correspondant à la métrique **mAP@0.50** (IoU = 0.5). Cette visualisation complète les résultats chiffrés donnés dans le tableau 4.4, et permet d'analyser plus finement le comportement du modèle au seuil de recouvrement généralement utilisé dans la métrique pour Pascal VOC [3]

Sur la Figure 4.10 ci-dessous, la courbe (a) présente les résultats du modèle **Faster R-CNN**, avec des valeurs d'Average Precision (AP) de **0.90** pour la classe **Abris en bois**, **0.84** pour **Metallique** et **0.69** pour **Non métallique**, soit une **mAP@0.50 de 0.809**.

3. Le challenge Pascal VOC est une référence en vision par ordinateur pour l'évaluation des méthodes de détection d'objets [71].



(a) Faster R-CNN

(b) RC R-CNN

FIGURE 4.10 – Courbes de précision-rappel interpolées pour les trois classes du jeu de test, correspondant à la métrique mAP@0.50 du tableau 4.4.

La courbe (b), correspondant au modèle **RC R-CNN**, indique des scores AP de **0.89**, **0.85** et **0.72** respectivement, avec une **mAP@0.50 de 0.812**. On observe donc une amélioration légère mais tangible, en particulier sur la classe **Non métallique**, qui reste la plus difficile à détecter.

Dans l'ensemble, ces courbes montrent que l'ajout de MPN-COV et de SPDNet permet de stabiliser légèrement les performances du détecteur, tout en préservant une bonne précision sur les classes majoritaires. L'amélioration observée sur la classe **Non métallique** suggère que ces modules statistiques renforcent la capacité de représentation du modèle dans des situations plus complexes. Ces résultats encouragent à poursuivre l'intégration de mécanismes de normalisation statistique dans les architectures de détection pour améliorer leur robustesse.

Comparaison des performances entre Faster R-CNN et RC R-CNN

Afin d'évaluer l'impact de l'intégration des modules **MPN-COV** et **SPD** dans l'architecture de détection, nous comparons ici les performances du modèle **Faster R-CNN** standard à celles du modèle **RC R-CNN**, toujours dans le cadre de la métrique **mAP@0.50** (IoU = 0.5) dans le tableau 4.4. Cette comparaison repose sur les matrices de confusion étendues et les métriques globales extraites de celles-ci, permettant une analyse fine des erreurs de classification et de localisation.

La Figure 4.11 illustre les matrices de confusion des deux modèles. Contrairement aux matrices classiques, une **colonne *No Detect*** est ajoutée pour indiquer les objets présents mais non détectés (faux négatifs), ainsi qu'une **ligne *IoU < 0.5*** qui regroupe les prédictions mal localisées ou affectées à une mauvaise classe (faux positifs).

Le modèle **Faster R-CNN** atteint une **accuracy globale de 71.4%**, tandis que l'ajout des couches MPN-COV et SPDNet permet de l'améliorer légèrement jusqu'à **73.6%**. Ces résultats traduisent une meilleure robustesse du modèle enrichi, notamment en termes de localisation correcte des objets détectés.

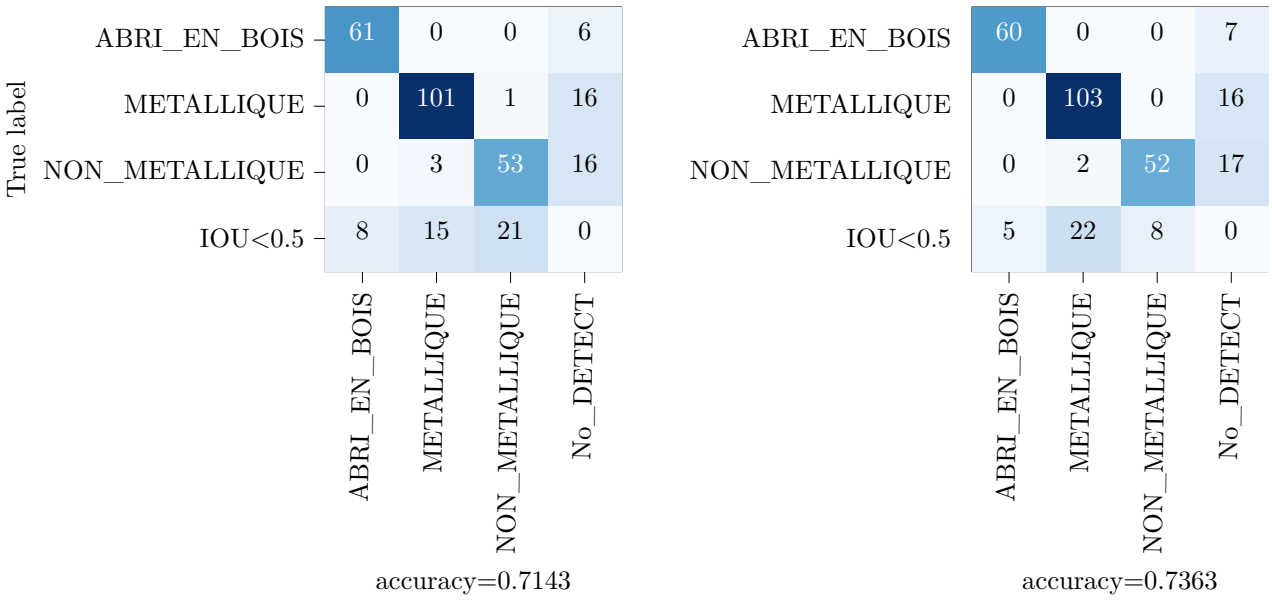


FIGURE 4.11 – Matrice de confusion sur les données de test. La colonne *No Detect* indique les objets manqués. La ligne $IoU < 0.5$ regroupe les prédictions incorrectes en termes de localisation.

Le **tableau 4.5** présente les métriques classiques (*précision*, *rappel*, *F1-score* et *support*) extraites à l'aide de la fonction `classification_report` de la bibliothèque `scikit-learn`, permettant de quantifier ces observations.

On constate que la classe **Abris en bois** bénéficie clairement de l'enrichissement du modèle, avec un F1-score qui passe de 0.89 à 0.91. La classe **Metallique** conserve des performances stables, et la classe **Non métallique** maintient un F1-score constant malgré un rappel plus faible. Ces résultats confirment une amélioration globale modérée mais robuste des performances avec l'intégration de MPN-COV et SPDNet.

Classe	Précision		Rappel		F1-score		Support	
	F R-CNN	RC R-CNN	F R-CNN	RC R-CNN	F R-CNN	RC R-CNN	F R-CNN	RC R-CNN
Abris en bois	0.89	0.92	0.88	0.90	0.89	0.91	67	67
Metallique	0.82	0.81	0.85	0.87	0.84	0.84	119	119
Non métallique	0.83	0.87	0.75	0.73	0.79	0.79	71	71

TABLE 4.5 – Comparaison des métriques de performance entre Faster R-CNN (noté F R-CNN) et **RC R-CNN** sur les données de test

En résumé, l'analyse des matrices de confusion permet d'identifier plus précisément les faiblesses et les atouts des deux modèles, au-delà des simples métriques globales. **RC R-CNN** montre non seulement une légère amélioration de l'accuracy globale, mais également une meilleure gestion des objets mal localisés, notamment pour les classes les plus difficiles comme **Non métallique**.

L'ajout de ces modules réduit la confusion entre les classes et stabilise les prédictions, sans hausse notable des faux positifs. Cela souligne l'intérêt des représentations de second ordre pour améliorer la sélectivité et la généralisation, notamment en présence d'objets aux caractéristiques visuelles proches.

Résultats visuels sur les données de test

Enfin, la Figure 4.12 montre des exemples visuels de détection. Les boîtes de gauche correspondent aux annotations manuelles, tandis que celles de droite affichent les prédictions du modèle **RC R-CNN**.

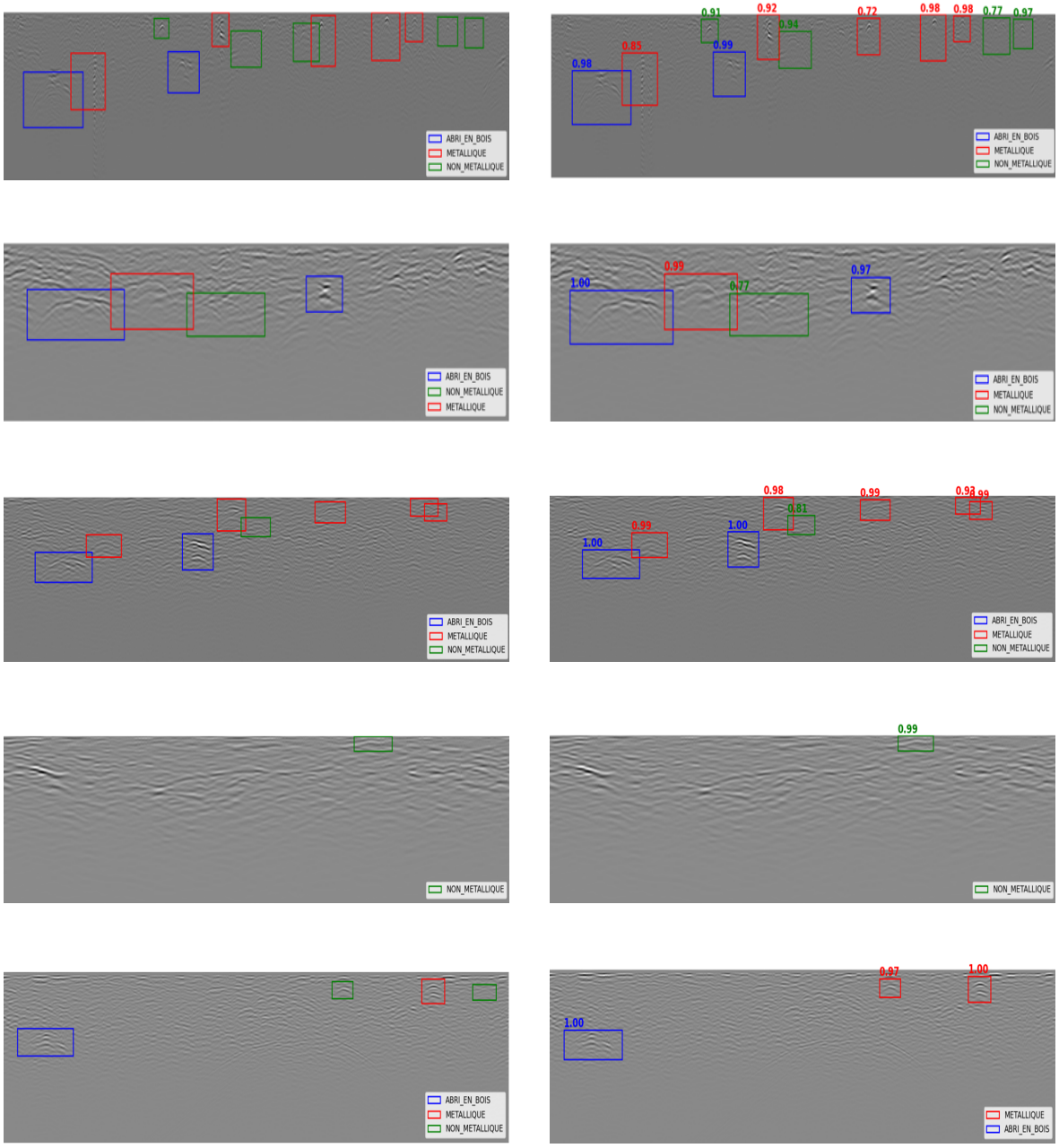


FIGURE 4.12 – Exemples de détections sur les données de test. À gauche : annotations réelles ; à droite : prédictions du modèle **RC R-CNN**

On constate une bonne correspondance spatiale entre les deux, preuve de la capacité du modèle à généraliser sur des données inédites.

4.3.6 Étude de la sensibilité au changement de distribution

Comme cela a été fait dans le chapitre 3 précédent de cette expérience, nous évaluons la robustesse du modèle face à un changement de distribution induit par l'élévation de mesure. Pour cela, nous avons construit trois ensembles à partir des données GPR acquises lors de deux scénarios. Contrairement à la partie précédente, dans cette partie, nous allons seulement montrer les résultats des métriques sur les données de test.

Scénario E : Changement du type d'élévation

Ce scénario E (ou du Dataset E) vise à analyser la robustesse des modèles de détection face à une modification des conditions d'acquisition, en particulier l'élévation du radar GPR par rapport au sol. Plus précisément, les élévations de **75 cm** et **100 cm** sont utilisées pour constituer les jeux d'entraînement et de validation, tandis que les données acquises à **50 cm** sont exclusivement réservées au test, comme cela a été fait dans le chapitre 3 précédent. Ce choix permet de simuler une distribution différente entre l'apprentissage et l'évaluation, afin d'évaluer la capacité de généralisation des modèles à des hauteurs d'acquisition non observées durant l'entraînement.

Le tableau 4.6 présente la distribution des **images radar** selon la fréquence d'antenne (200 MHz ou 350 MHz), le type de sol (grave humide, grave sèche, sable sec, sable humide), et l'élévation du radar (50 cm, 75 cm, 100 cm). Cette diversité de conditions permet d'assurer une variabilité suffisante dans les données, tout en respectant la contrainte du scénario expérimental.

	grave humide			grave sèche			sable sec			sable humide			**Total**
Fréquence / Élévation (cm)	50	75	100	50	75	100	50	75	100	50	75	100	
200 MHz	2	7	4	5	7	10	5	16	16	3	7	8	**90**
350 MHz	1	5	7	2	7	7	7	20	22	5	15	10	**108**
Total	3	12	11	7	14	17	12	36	38	8	22	18	**198**

TABLE 4.6 – Distribution des radargrammes selon la fréquence, le type de sol et l'élévation (75,100 vs 50) du radar.

Chaque image radar contient un ou plusieurs objets annotés sous la forme de boîtes englobantes (bounding boxes), appartenant à l'une des trois classes suivantes : **Metallique**, **Non métallique**, et **Abris en bois**.

Le tableau 4.7 présente la répartition de ces annotations selon les différentes partitions du jeu de données. On y observe que le jeu d'entraînement comprend 138 images, soit près de 70 % du total, pour un total de 629 objets annotés. Le jeu de validation, composé de 30 images, contient 96 annotations. Enfin, le jeu de test, également constitué de 30 images, inclut 137 objets annotés, ce qui en fait une base d'évaluation significative malgré sa taille restreinte.

Jeu de données	**Metallique**	**Non métallique**	**Abris en bois**	**Total**
Entraînement (138 images)	277	163	189	629
Validation (30 images)	40	24	32	96
Test (30 images)	52	44	41	137
Total général	369	231	262	**862**

TABLE 4.7 – Répartition des annotations par classe dans chaque partition du Dataset E (Scénario E : changement d'élévation).

Cette distribution, plutôt équilibrée malgré une présence importante des hyperboles de la catégorie métallique entre les classes, permet d'assurer une évaluation cohérente de la performance du modèle, tout en mettant l'accent sur sa capacité à généraliser à une élévation inconnue lors de l'apprentissage.

Résultats des métriques sur les données de test Le tableau 4.8 présente les résultats obtenus sur les données de test. Deux variantes du modèle Faster R-CNN sont évaluées : la version classique et une version enrichie par l'ajout d'un bloc SPDNet (RC R-CNN).

Métrique	IoU	Area	MaxDets	Faster R-CNN	RC R-CNN
mAP	0.50 :0.95	all	100	0.240	0.322
mAP	**0.50**	**all**	**100**	**0.518**	**0.612**
mAP	0.75	all	100	0.165	0.296
mAR	0.50 :0.95	all	100	0.320	0.368

TABLE 4.8 – Comparaison des performances pour les données de test du Dataset E sur l'élévation 50 cm entre **Faster R-CNN** et **RC R-CNN** selon les métriques (scénario E).

Dans les résultats, le gain le plus notable est observé sur la métrique **mAP@75**, qui passe de **0.165** à **0.296**, soit une progression de plus de 13 points. Cela suggère une nette amélioration de la précision de localisation : le modèle est plus apte à produire des détections très bien alignées avec les objets réels.

La métrique **mAP@50** progresse également, passant de **0.518** à **0.612**, ce qui traduit une meilleure capacité globale à détecter les objets, même avec un seuil d'IoU plus permissif. La précision moyenne stricte **(mAP@[0.50 :0.95])** augmente de manière significative, de **0.240** à **0.322**, confirmant ainsi une amélioration globale de la qualité des prédictions sur l'ensemble des seuils de recouvrement.

Enfin, le **rappel moyen (mAR)** progresse de **0.320** à **0.368**, ce qui indique que **RC R-CNN** parvient non seulement à détecter avec plus de précision, mais aussi à récupérer un plus grand nombre d'objets pertinents.

En conclusion, ces résultats confirment notre hypothèse selon laquelle l'ajout d'un module de normalisation de second ordre (SPDNet) améliore la robustesse du modèle face aux décalages de distribution entre les données d'entraînement et celles de test, reproduisant ainsi le même comportement que dans le cas de la classification.

Scénario F : Changement du type de sol

Ce scénario **F (ou Dataset F)** vise à évaluer la robustesse des modèles lorsqu'ils sont confrontés à un **changement de type de sol** entre l'entraînement et le test. Les données issues des terrains en *grave sèche* sont utilisées pour l'entraînement et la validation, tandis que celles en *grave humide* servent exclusivement pour le test.

Le tableau 4.9 montre la répartition des **images radar** selon la fréquence d'antenne, l'élévation du radar et le type de sol. La base est constituée de 64 images couvrant plusieurs élévations et deux types de terrain.

	grave humide			grave sèche			**Total**
Fréquence / Élévation (cm)	50	75	100	50	75	100	
200 MHz	2	7	4	5	7	10	**35**
350 MHz	1	5	7	2	7	7	**29**
Total	3	12	11	7	14	17	**64**

TABLE 4.9 – Distribution des radargrammes selon la fréquence, l'élévation et le type de sol (grave humide vs grave sèche).

La répartition des objets annotés est indiquée dans le tableau 4.10. On y constate une distribution relativement équilibrée entre les trois classes, avec un total de 80 annotations pour le test. Cette configuration permet de tester la généralisation des modèles sur un sol différent de celui observé pendant l'apprentissage.

Jeu de données	**Abris en bois**	**Metallique**	**Non métallique**	**Total**
Entraînement (94 images)	113	142	86	341
Validation (21 images)	25	29	20	74
Test (21 images)	29	31	20	80
Total général	167	202	126	**495**

TABLE 4.10 – Répartition des annotations par classe dans chaque partition du dataset F (Scénario F : changement du type de sol).

Résultats des métriques sur les données de test Le tableau 4.11 présente les résultats obtenus sur les données de test. Deux variantes du modèle Faster R-CNN sont évaluées : la version classique et une version enrichie par l'ajout d'un bloc SPDNet (RC R-CNN).

Métrique	**IoU**	**Area**	**MaxDets**	**Faster R-CNN**	**RC R-CNN**
mAP	0.50 :0.95	all	100	0.083	0.207
mAP	**0.50**	**all**	**100**	**0.237**	**0.447**
mAP	0.75	all	100	0.033	0.155
mAR	0.50 :0.95	all	100	0.146	0.249

TABLE 4.11 – Résultats de performance pour les données de test du Dataset F sur le sol grave humide (comparaison entre **Faster R-CNN** et **RC R-CNN**).

Les résultats mettent encore en évidence une amélioration systématique, bien que modeste, des performances lorsque le modèle est enrichi avec SPDNet. Le gain le plus notable est observé sur la métrique mAP@75, qui passe de **0.033** à **0.155**, suggérant une amélioration dans la précision de localisation, c'est-à-dire une meilleure capacité à produire des détections très bien alignées avec les objets réels.

La métrique mAP@50 présente une nette amélioration, passant de **0.237** à **0.447**. La précision plus stricte (mAP@[0.50 :0.95]) progresse également, mais de manière plus modérée. Enfin, le rappel moyen (mAR@0.50 :0.95) augmente de manière significative, passant de **0.146** à **0.249**. Cela indique que le modèle **RC R-CNN** ne se contente pas d'améliorer la précision des boîtes détectées, mais parvient également à détecter un plus grand nombre de cibles pertinentes. Cette progression conjointe des scores de précision et de rappel suggère que **RC R-CNN** fonctionne même dans des cas de forte ambiguïté inter-classes.

Nous obtenons donc les mêmes conclusions que pour l'expérimentation précédente, avec une robustesse accrue des modèles construits à partir de matrices de covariance.

4.4 Conclusion

L'objectif de ce chapitre était d'aller au-delà de la classification menée au chapitre 3, où la position des cibles était supposée connue, et d'examiner la faisabilité d'une détection *et* classification conjointe des hyperboles GPR dans un cadre entièrement automatique. Pour cela, nous avons commencé par rappeler les principes fondamentaux des détecteurs à régions, en retraçant brièvement l'évolution de R-CNN à Fast R-CNN, puis à Faster R-CNN, et en illustrant chaque étape à l'aide d'un exemple d'image GPR issu de notre jeu de données. Pour rester dans la logique amorcée précédemment, nous avons conservé la même base convolutive, à savoir un ResNet-34 non pré-entraîné, utilisé ici comme backbone complet, contrairement au chapitre 3 où seules les premières couches étaient utilisées pour construire une matrice de covariance globale. Dans le présent chapitre, nous avons intégré, après la phase RoI Align, une représentation de second ordre via MPN-COV, suivie d'un traitement géométrique par SPDNet. Ce processus transforme chaque région d'intérêt en une matrice de covariance normalisée et exploite la structure riemannienne de l'espace des matrices SPD pour renforcer la robustesse de la branche de classification, tandis que la branche de régression des boîtes demeure classique afin de ne pas pénaliser la précision spatiale.

Sur une grande partie du jeu de données, les premiers résultats obtenus montrent que cette architecture, que nous avons nommée **RC R-CNN** (Residual Covariance R-CNN), dépasse systématiquement la version standard de **Faster R-CNN** sur les données GPR considérées. L'amélioration est particulièrement sensible sur les métriques les plus strictes, notamment **mAP@0.75**, et se traduit par une localisation plus fine ainsi qu'une meilleure différenciation des classes les plus ambiguës, en particulier les cibles non métalliques. De plus, les meilleurs scores obtenus sur **mAP@0.5**, mis en gras dans les tableaux correspondants, témoignent d'un gain global sur l'ensemble des seuils d'IoU, confirmant ainsi l'intérêt de l'encodage par covariance et du traitement géométrique via SPDNet dans ce contexte spécifique.

Nous avons également évalué la sensibilité de cette approche face à des changements de distribution entre l'entraînement, la validation et le test. En simulant des écarts réalistes d'élévation de l'antenne (75 cm, 100 cm vs 50 cm) ou encore de nature du sol (grave seche vs grave humide), le RC R-CNN a conservé une performance stable et améliorée. Cela contraste avec Faster R-CNN, qui est plus sensible à ces changements. Ces résultats prolongent ceux du chapitre 3, où les modèles RCNet et SRCNet avaient déjà montré une meilleure généralisation en situation de décalage de distribution.

Il convient toutefois de souligner qu'il s'agit d'une étude préliminaire : les ensembles de données restent de taille limitée, les hyperparamètres n'ont pas été optimisés de façon exhaustive, et une seule configuration de backbone et de fréquence radar a été testée. Même si les résultats sont encourageants, le travail devra donc être approfondi par des phases d'entraînement plus étendues, comme cela a été réalisé dans le chapitre 3, afin de confirmer la robustesse de la méthode dans des contextes plus variés et d'évaluer son comportement sur des ensembles de données plus volumineux.

RC R-CNN constitue, à ce stade, un exemple prometteur illustrant l'intérêt d'associer **MPN-COV** et **SPDNet** à un détecteur à régions, sans modifier radicalement l'architecture d'origine et en apportant un bénéfice concret tant en termes de classification que de localisation. Ce dernier chapitre démontre ainsi qu'il est possible d'exploiter efficacement les représentations de second ordre dans un pipeline complet de détection automatique sur des données GPR, en conciliant robustesse géométrique, performance statistique et intégration pratique dans des architectures de détection.

Chapitre 5

Conclusion générale

Résumé des résultats

Cette thèse concerne le développement de modèles robustes pour la détection et la classification d'objets enfouis dans des données GPR (*Ground Penetrating Radar*), en s'appuyant sur l'exploitation des statistiques de second ordre sous forme de matrices de covariance, ainsi que sur des outils issus de la géométrie différentielle adaptés aux matrices symétriques définies positives (SPD). L'objectif était de proposer une chaîne de traitement automatique fiable dans un contexte marqué par des défis importants : faible rapport signal/bruit, conditions expérimentales variables et données peu nombreuses. À travers ce manuscrit, nous avons pu proposer plusieurs solutions et analyses pour répondre aux différentes problématiques énumérées en introduction.

Le **chapitre 1** a posé les bases physiques et expérimentales du travail. Après avoir rappelé le principe de fonctionnement du GPR, ce chapitre a mis en évidence la difficulté d'interprétation des images radar, notamment en raison du bruit, des artefacts et des variations dues au type de sol ou à la configuration d'acquisition. Une présentation d'une campagne de mesure rigoureuse sur la carrière de Barraux est ensuite proposée. Cette campagne a permis de constituer une base d'images riche et diversifiée, avec plusieurs configurations de fréquence, d'élévation et de sol. Un important travail d'annotation et de structuration des données réalisé par Géolithe a permis de constituer un corpus exploitable pour l'apprentissage supervisé. Ce chapitre a ainsi posé les conditions nécessaires à une exploration rigoureuse des approches de classification et de détection.

Dans le **chapitre 2**, nous avons évalué différentes approches de classification supervisée. Après avoir testé des classifieurs classiques tels que SVM et Random Forest sur des descripteurs linéaires, nous avons observé que leurs performances restaient très limitées dans des scénarios de test réalistes. En revanche, l'introduction d'un descripteur basé sur la matrice de covariance a permis de capturer les dépendances internes au signal, offrant des résultats nettement plus robustes, même en présence de bruit ou de déséquilibre dans les données. Ce chapitre a ainsi servi de point de départ à une généralisation plus ambitieuse : intégrer le second ordre directement dans un pipeline profond adapté.

Le **chapitre 3** a répondu à cette ambition en proposant une modélisation hybride des données GPR, fondée sur des représentations de second ordre traitées dans un espace géométrique non euclidien. Deux variantes ont été proposées : RCNet et SRCNet, qui exploitent uniquement les premières couches d'un ResNet-34 non préentraîné afin d'extraire des activations locales. Ces activations sont ensuite agrégées sous la forme de matrices de covariance, qui servent de représentation compacte et discriminante. Chaque matrice SPD ainsi obtenue est ensuite traitée par un réseau SPDNet, constitué de trois couches principales (BiMap, ReEig, LogEig), qui assurent successivement des projections riemanniennes, une régularisation

spectrale, puis un retour dans un espace euclidien.

Les expérimentations ont mis en évidence une forte supériorité de RCNet sur les approches CNN standard, en particulier lorsque le ratio d'entraînement diminue ou que les conditions d'acquisition changent (fréquence, sol, élévation). Ce modèle a également montré une grande stabilité inter-répétition, preuve d'une capacité de généralisation plus fiable. Ce chapitre a donc posé les fondations géométriques et statistiques d'un modèle robuste, en réponse directe aux limites constatées dans le chapitre 2.

Le **chapitre 4** a généralisé cette approche à une tâche de détection conjointe. L'hypothèse de position connue a été levée, et nous avons proposé une extension du modèle Faster R-CNN pour l'adapter au contexte GPR. Le backbone ResNet-34 est conservé pour garantir une cohérence avec les architectures précédentes. Après la phase `RoI Align`, chaque région d'intérêt est encodée à l'aide de `MPN-COV`, une opération de covariance régularisée, puis traitée par `SPDNet` dans la branche de classification. Cette architecture complète, baptisée **RC R-CNN**, permet de renforcer la robustesse de la classification tout en maintenant la précision spatiale. Les résultats obtenus montrent une amélioration sensible sur les métriques fines, telles que la mAP@0.75, ainsi qu'une meilleure différenciation des cibles difficiles (par exemple, les objets non métalliques). Ce chapitre valide ainsi que la logique d'encodage de second ordre développée pour la classification peut être transférée dans un cadre de détection *end-to-end* sans compromettre l'intégrité du pipeline.

Dans l'ensemble, cette thèse montre que les représentations de second ordre, via des matrices de covariance correctement normalisées et insérées dans un espace géométrique adapté, constituent une solution naturelle et efficace pour renforcer la robustesse des modèles d'analyse GPR. Ces représentations permettent de capter des invariants internes à la structure des signaux, lesquels sont peu sensibles aux changements expérimentaux, aux variations inter-sites ou aux erreurs d'annotation. L'exploitation du cadre riemannien, via SPDNet, permet d'opérer une transformation fidèle et discriminante de ces matrices, tout en conservant une compatibilité avec les outils profonds modernes. De la classification supervisée à la détection conjointe, cette stratégie s'est révélée à la fois stable, performante et extensible.

Perspectives

Plusieurs axes de recherche sont envisageables dans la continuité de ce travail. Un premier axe concerne l'estimation robuste des matrices de covariance. Les estimateurs tels que ceux de Tyler ou du type MLE-Student [51], bien que non retenus dans la version finale, constituent des alternatives pertinentes à l'estimateur de covariance empirique (SCM) utilisé dans les chapitres 2 et 3 pour sa simplicité et sa compatibilité avec la rétropropagation. Ces estimateurs offrent une meilleure gestion des valeurs aberrantes et des distributions non gaussiennes. Leur stabilisation numérique et leur différentiabilité permettront leur réintégration dans un pipeline géométrique profond, comme suggéré dans [42].

Un deuxième axe de travail porte sur l'adaptation du modèle **RC R-CNN** introduit au chapitre 4 à des variantes plus récentes et plus performantes du framework Faster R-CNN. Des architectures comme Cascade R-CNN [72], avec ses têtes successives à seuils d'IoU croissants, ou Libra R-CNN [73], avec son équilibrage pondéré et sa fusion multi-niveaux, pourraient améliorer significativement les performances, notamment pour la détection à faible IoU ou pour les cibles partiellement visibles.

Concernant l'utilisation de ResNet-34, il est important de distinguer les usages selon les chapitres : dans les chapitres 2 et 3, seules certaines couches convolutionnelles extraites de ResNet-34 ont servi à construire les matrices de covariance exploitables par SPDNet, tandis que, dans le chapitre 4, ResNet-34 est utilisé comme backbone complet du modèle RC R-CNN. Cette distinction invite à tester différentes architectures selon les contextes, allant de réseaux plus profonds comme ResNet-101, ResNeXt [74] ou Swin Transformer [75], à des modèles plus compacts comme MobileNet [49], EfficientNet [76] ou ShuffleNet [77], tout en conservant l'intégration modulaire des couches géométriques SPDNet dans la branche

de classification.

Enfin, un enjeu crucial pour un déploiement opérationnel reste l'**adaptation de domaine**. Comme souligné tout au long de cette thèse (chapitres 2, 3 et 4), les performances des modèles peuvent chuter lorsque les conditions d'acquisition changent (type de sol, fréquence, élévation). Il sera essentiel de développer des approches auto-supervisées ou faiblement supervisées, capables de s'adapter automatiquement à de nouveaux environnements sans nécessiter d'annotations coûteuses. Parmi les pistes prometteuses figurent les méthodes d'alignement statistique via le transport optimal [78] ou l'adaptation des représentations internes à l'image des Deep Adaptation Networks [79], qui rapprochent les distributions source et cible dans un espace latent commun.

À plus long terme, la méthode pourrait être testée sur d'autres jeux de données, qu'ils soient issus de mesures GPR variées ou de domaines similaires (par exemple, la tomographie, les données sismiques ou les signaux hyperfréquences), afin d'évaluer sa capacité de généralisation à des signaux physiques de nature comparable.

En définitive, cette thèse pose les bases d'une approche unifiée et robuste pour l'analyse automatique des données GPR, exploitant la structure de second ordre des activations et leur traitement géométrique. Elle ouvre la voie à des systèmes plus fiables, capables de s'adapter aux contraintes du terrain tout en réduisant la dépendance à l'expertise humaine et aux données parfaitement annotées. Le chemin reste ouvert vers une automatisation complète et robuste des campagnes GPR en contexte réel.

Annexe A

Rôle des fonctions noyau

Lorsque les données ne sont pas linéairement séparables dans l'espace d'origine, il est possible de généraliser les SVM en utilisant une *fonction noyau*. L'idée consiste à projeter les données dans un espace de dimension plus élevée, où une frontière de décision linéaire devient réalisable. Cette projection est effectuée implicitement grâce au *truc du noyau* (*kernel trick*), qui permet de calculer les produits scalaires dans l'espace transformé sans avoir à expliciter les coordonnées des données dans cet espace.

Dans ce contexte, x et x' désignent deux **échantillons** (ou **vecteurs de caractéristiques**) issus de la base d'apprentissage. La fonction noyau $K(x, x')$ mesure une forme de similarité entre ces deux observations.

Parmi les noyaux les plus couramment utilisés, on peut citer :

- **Noyau linéaire** :
$$K(x, x') = \langle x, x' \rangle,$$
correspondant au produit scalaire entre les deux vecteurs ;
- **Noyau polynomial** :
$$K(x, x') = (\langle x, x' \rangle + c)^d,$$
où c et d contrôlent respectivement le biais et le degré du polynôme ;
- **Noyau gaussien (RBF)** :
$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right),$$
qui mesure une similarité décroissante avec la distance euclidienne entre x et x' ;
- **Noyau sigmoïde** :
$$K(x, x') = \tanh(\alpha \langle x, x' \rangle + c),$$
inspiré des neurones à activation sigmoïde.

Le choix du noyau et de ses paramètres (par exemple d, σ, c ou α) est déterminant pour la performance du modèle. Un réglage inadéquat peut conduire à un sur-apprentissage ou, au contraire, à un modèle insuffisamment expressif. En pratique, ces paramètres sont généralement optimisés par validation croisée.

Annexe B

Géométrie Riemannienne

Cette annexe fournit quelques notions de géométrie riemannienne en présentant les principales définitions. Il décrit aussi l'algorithme de descente du gradient Riemannien qui est utilisé dans l'étape de backpropagation du réseau SPDNet dans les chapitres 3 et 4.

A Notions de base sur les variétés riemanniennes

On va définir les sous-variétés, notées $\mathcal{M}$, des espaces linéaires $\mathcal{E}$. Informellement, ce sont des sous-ensembles de $\mathcal{E}$ qui sont soit ouverts, soit définis par des contraintes $h : \mathcal{E} \to \mathbb{R}^k$ avec $k > 0$. Dans ce dernier cas, un point $x \in \mathcal{E}$ appartient à $\mathcal{M}$ si et seulement si $h(x) = 0$.

Définition A.1 (Définition 3.10 de [58])**.** Soit $\mathcal{E}$ un espace linéaire de dimension d. Un sous-ensemble non vide $\mathcal{M}$ de $\mathcal{E}$ est un sous-milieu lisse intégré de $\mathcal{E}$ de dimension q si soit

1. $q = d$ et $\mathcal{M}$ est ouvert dans $\mathcal{E}$ - nous l'appelons aussi sous-modèle ouvert ou
2. $q = d - k$ pour $k \geq 1$ et, pour chaque $x \in \mathcal{M}$, il existe un voisinage U de x dans $\mathcal{E}$ et une fonction lisse $h : U \to \mathbb{R}^k$ telle que
   (a) Si y est dans U, alors $h(y) = 0$ si et seulement si $y \in \mathcal{M}$; et
   (b) $\text{rank}(\text{D}h(x)) = k$ avec $\text{D}h(x)$ est le différentiel de h à x

Un outil primordial en géométrie différentielle est l'espace tangent à x, qui est défini pour un élément $x \in \mathcal{M}$. Informellement, il correspond à une linéarisation de $\mathcal{M}$ en x. Les espaces tangents sont d'une importance capitale. En effet, nous verrons plus loin que ce sont des espaces linéaires. Ainsi, les opérations classiques, telles que l'addition ou la multiplication, ainsi que les opérations liées aux produits intérieurs, sont possibles sur l'espace tangent, contrairement à la variété, qui n'est généralement pas un espace linéaire.

Théorème A.1 (Théorème 3.15 de [58])**.** *Soit $\mathcal{M}$ un sous-milieu intégré de $\mathcal{E}$. Considérons $x \in \mathcal{M}$ et l'ensemble $T_x\mathcal{M}$. Si $\mathcal{M}$ est un sous-milieu ouvert, alors*

$$T_x\mathcal{M} = \mathcal{E} \tag{B.1}$$

Sinon,

$$T_x\mathcal{M} = \ker(\text{D}h(x)) = \{\xi \in \mathcal{E} : \text{D}h(x)[\xi] = 0\} \tag{B.2}$$

avec h toute fonction de définition locale à x et $\text{D}h(x)[\xi]$ est la dérivé directionnelle de h à x dans la direction ξ.

Finalement, pour avoir une variété Riemannienne, il est nécessaire de définir un produit scalaire sur le plan tangent $T_x\mathcal{M}$.

Définition A.2 (Définition 3.52 de [58]). Une métrique $\langle .,. \rangle_x$ sur $\mathcal{M}$ est une métrique riemannienne si elle varie de manière lisse avec x, au sens où pour tous les champs de vecteurs lisses ξ, η sur $\mathcal{M}$ la fonction $x \mapsto \langle \xi(x), \eta(x) \rangle_x$ est lisse de $\mathcal{M}$ à $\mathbb{R}$.

La métrique permet de définir la norme d'un vecteur, ainsi que la géodésique $\gamma(t)$ qui est le chemin le plus court entre deux points de la variété. Nous pouvons ensuite définir, à partir de cette métrique et de la géodésique, deux applications très importantes qui sont l'exponentielle et le logarithme.

Définition A.3 (Définition 10.16 de [58]). Pour tout $(x, \xi) \in T_x\mathcal{M}$, on appelle $\gamma_\xi : I \to \mathcal{M}$ soit l'unique géodésique avec $\gamma_\xi(0) = x, \dot{\gamma}_\xi(0) = \xi$ et $\gamma_\xi(0) = \xi$ et $\gamma_\xi(0) = \xi$ aussi grand que possible. Considérons le sous-ensemble suivant du faisceau tangent :

$$\mathcal{O} = \{(x, \xi) \in T_x\mathcal{M} : \gamma_\xi \text{ est défini sur un intervalle contenant } [0, 1]\} \tag{B.3}$$

L'exponentielle $\exp^{\mathcal{M}} : \mathcal{O} \to \mathcal{M}$ est définie par

$$\exp^{\mathcal{M}}(x, \xi) = \exp_x^{\mathcal{M}}(\xi) = \gamma_\xi(1) \tag{B.4}$$

D'un point de vue pratique, cet opérateur permet de ramener un point du plan tangent sur la variété. Il sera évidemment très utile dans les algorithmes d'optimisation.

Lorsqu'elle existe, l'inverse de l'exponentielle est appelé logarithme. Étant donné $x, y \in \mathcal{M}$, elle retourne le vecteur tangent $\xi \in T_x\mathcal{M}$ tel que $\exp_x^{\mathcal{M}}(\xi) = y$. Elle est introduite dans la définition suivante.

Définition A.4 (Définition 10.20 de [58]). Pour $x \in \mathcal{M}$, notons $\log_x^{\mathcal{M}}$ la carte logarithmique à x,

$$\log_x^{\mathcal{M}}(y) = \underset{\xi \in \mathcal{O}_x}{arg\,\min} \|\xi\|_x \text{ subject to } \exp_x^{\mathcal{M}}(\xi) = y \tag{B.5}$$

avec un domaine tel que celui-ci est défini de manière unique.

B Principe de l'algorithme du gradient

Dans cette section, nous présentons un algorithme de minimisation des fonctions lisses sur les variétés, c'est-à-dire

$$\underset{x \in \mathcal{M}}{\text{minimize}}\ f(x)$$

où $\mathcal{M}$ est un manifold et f est un champ scalaire lisse, c'est-à-dire une fonction lisse de $\mathcal{M}$ à $\mathbb{R}$, appelée la fonction de coût. Nous commençons par la définition du gradient riemannien, qui étend la définition du gradient sur les espaces euclidiens aux variétés riemanniennes.

Définition B.1 (Définition 3.58 de [58]). Soit $f : \mathcal{M} \to \mathbb{R}$ soit lisse sur une variété Riemannienne $\mathcal{M}$. Le gradient riemannien de f est le champ de vecteurs $\text{grad}_{\mathcal{M}} f$ sur $\mathcal{M}$ défini uniquement par les identités suivantes :

$$\forall (x, \xi) \in T_x\mathcal{M}, \quad \mathrm{D}f(x)[\xi] = \langle \text{grad}_{\mathcal{M}} f(x), \xi \rangle_x \tag{B.6}$$

où $\mathrm{D}f(x)$ la dérivé directionnelle de f dans la direction ξ et $\langle ..., . \rangle_x$ est la métrique riemannienne définie au point x.

Pour trouver les points critiques qui annulent le gradient, les algorithmes d'optimisation fondés sur le gradient dans les espaces euclidiens sont adaptés aux variétés Riemanniennes $\mathcal{M}$. La principale difficulté vient de la non-linéarité (en général) de $\mathcal{M}$. En effet, une étape de descente de gradient ne retourne pas nécessairement un point sur $\mathcal{M}$, c'est-à-dire que pour un itéré donné $x^{(l)} \in \mathcal{M}$ et une taille de pas $\alpha > 0$

$$x^{(l+1)} = x^{(l)} - \alpha \,\text{grad}_{\mathcal{M}}\, h\left(x^{(l)}\right) \notin \mathcal{M}(\text{en général}). \tag{B.7}$$

Pour surmonter ce problème, nous cherchons des algorithmes itératifs qui se déplacent le long de la variété $\mathcal{M}$. Une solution est de trouver un opérateur nous permettant, à chaque itération, de revenir sur la variété. L'opérateur défini dans la section précédente, $\xi \in T_x\mathcal{M}, t \mapsto f\left(\exp_x^{\mathcal{M}}(t\xi)\right)$, est un bon candidat. En général, pour diverses raisons, on préfère utiliser une approximation de cet opérateur $\exp_x^{\mathcal{M}}$, appelée rétraction, définie ci-dessous.

Définition B.2 (Définition 3.47 de [58])**.** Une rétraction sur une variété $\mathcal{M}$ est une carte lisse

$$R_x^{\mathcal{M}} : T_x\mathcal{M} \to \mathcal{M} : (x, \xi) \mapsto R_x^{\mathcal{M}}(\xi) \tag{B.8}$$

telle que chaque courbe $c(t) = R_x^{\mathcal{M}}(t\xi)$ satisfasse $c(0) = x$ et $\dot{c}(0) = \xi$.

Comme dit précédemment, il convient de noter que les applications exponentielles $\exp_x^{\mathcal{M}}$ sont des rétractions. Par conséquent, les rétractions généralisent les cartes exponentielles et ne respectent que les propriétés importantes pour l'optimisation. Les rétractions peuvent être développées lorsque l'application exponentielle n'est pas disponible sous forme fermée, qu'elle est trop coûteuse à calculer ou qu'elle n'est pas stable numériquement.

Avec cet outil de rétraction, on peut enfin définir l'algorithme de descente du gradient Riemannien. Comme dans le cas euclidien, la direction de descente est $\xi = -\operatorname{grad}_{\mathcal{M}} f\left(x^{(l)}\right)$. Ensuite, il faut utiliser l'opérateur de rétraction pour revenir sur la variété. Ces opérations sont répétées jusqu'à la convergence. Pour le pas, une *line-search* est souvent effectuée à chaque étape. Un résumé de l'algorithme est montré dans l'algorithme 1.

Algorithm 1: Descente du gradient riemannien

Data: Initialisation : $x^{(0)} \in \mathcal{M}$
Result: $x^{(l)} \in \mathcal{M}$

for $l = 0$ *jusqu'à convergence* **do**
  $\xi^{(l)} = -\operatorname{grad}_{\mathcal{M}} f\left(x^{(l)}\right)$;
  $\alpha = \text{Linesearch}\left(x^{(l)}, \xi^{(l)}\right)$;
  $x^{(l+1)} = R_{x^{(l)}}^{\mathcal{M}}\left(\alpha\xi^{(l)}\right)$;
end

C Exemples

Dans cette dernière section, on donne les valeurs des outils définis précédemment (plan tangent, métrique, $\exp_x^{\mathcal{M}}$ pour les deux variétés utilisées dans cette thèse : la variété des matrices symétriques définies positives (SPD) et celle de Stiefel (ensemble des matrices orthogonales rectangulaires).

C.1 Variété des matrices symétriques définies positives (SPD)

Définition C.1 (Variété SPD)**.** La variété des matrices symétriques définies positives de taille $n \times n$ est définie par :

$$\mathcal{S}_{++}^n = \{X \in \mathbb{R}^{n\times n} : X = X^T \text{ et } X \succ 0\}$$

où $X \succ 0$ signifie que X est définie positive.

Définition C.2 (Espace tangent)**.** L'espace tangent à $\mathcal{S}_{++}^n$ en un point $X \in \mathcal{S}_{++}^n$ est :

$$T_X\mathcal{S}_{++}^n = \{Z \in \mathbb{R}^{n\times n} : Z = Z^T\}$$

c'est-à-dire l'espace des matrices symétriques.

Définition C.3 (Métrique riemannienne)**.** La métrique riemannienne affine-invariante sur $\mathcal{S}_{++}^n$ est définie par :

$$\langle Z_1, Z_2 \rangle_X = \mathrm{tr}(X^{-1} Z_1 X^{-1} Z_2)$$

pour $Z_1, Z_2 \in T_X \mathcal{S}_{++}^n$.

Définition C.4 (Application exponentielle)**.** L'application exponentielle $\mathrm{Exp}_X : T_X \mathcal{S}_{++}^n \to \mathcal{S}_{++}^n$ est définie par :

$$\mathrm{Exp}_X(Z) = X^{1/2} \exp(X^{-1/2} Z X^{-1/2}) X^{1/2}$$

où $\exp(\cdot)$ est l'exponentielle matricielle standard.

Définition C.5 (Forme alternative de l'exponentielle)**.** Une forme équivalente de l'application exponentielle est :

$$\mathrm{Exp}_X(Z) = X \exp(X^{-1} Z)$$

Cette formulation évite le calcul explicite de $X^{1/2}$.

Définition C.6 (Application logarithmique)**.** L'application logarithmique $\mathrm{Log}_X : \mathcal{S}_{++}^n \to T_X \mathcal{S}_{++}^n$ est l'inverse de l'application exponentielle :

$$\mathrm{Log}_X(Y) = X^{1/2} \log(X^{-1/2} Y X^{-1/2}) X^{1/2}$$

où $\log(\cdot)$ est le logarithme matriciel.

C.2 Variété de Stiefel

Définition C.7 (Variété de Stiefel)**.** La variété de Stiefel $\mathrm{St}(n, p)$ avec $n \geq p$ est définie par :

$$\mathrm{St}(n, p) = \{X \in \mathbb{R}^{n \times p} : X^T X = I_p\}$$

où I_p est la matrice identité de taille $p \times p$. Elle représente l'ensemble des matrices orthogonales rectangulaires.

Définition C.8 (Espace tangent)**.** L'espace tangent à $\mathrm{St}(n, p)$ en un point $X \in \mathrm{St}(n, p)$ est :

$$T_X \mathrm{St}(n, p) = \{Z \in \mathbb{R}^{n \times p} : X^T Z + Z^T X = 0\}$$

c'est-à-dire l'ensemble des matrices Z telles que $X^T Z$ est antisymétrique.

Définition C.9 (Métrique riemannienne canonique)**.** La métrique riemannienne canonique sur $\mathrm{St}(n, p)$ est définie par :

$$\langle Z_1, Z_2 \rangle_X = \mathrm{tr}(Z_1^T Z_2)$$

pour $Z_1, Z_2 \in T_X \mathrm{St}(n, p)$. Cette métrique est héritée de la métrique euclidienne de $\mathbb{R}^{n \times p}$.

Définition C.10 (Projection sur l'espace tangent)**.** La projection d'une matrice $A \in \mathbb{R}^{n \times p}$ sur l'espace tangent $T_X \mathrm{St}(n, p)$ est :

$$P_X(A) = A - X(X^T A)$$

Cette projection assure que $X^T P_X(A)$ est antisymétrique.

Définition C.11 (Application exponentielle)**.** L'application exponentielle $\mathrm{Exp}_X : T_X \mathrm{St}(n, p) \to \mathrm{St}(n, p)$ est définie par :

$$\mathrm{Exp}_X(Z) = XV \cos(\Sigma) V^T + U \sin(\Sigma) V^T$$

où $Z = U \Sigma V^T$ est la décomposition en valeurs singulières de Z, avec $U \in \mathbb{R}^{n \times p}$, $V \in \mathbb{R}^{p \times p}$ orthogonale, et $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_p)$.

Définition C.12 (Formulation alternative de l'exponentielle)**.** Une formulation équivalente utilise la décomposition QR :

$$\mathrm{Exp}_X(Z) = [X + Z, X_\perp] \exp \begin{pmatrix} X^T Z \\ -Z^T X \end{pmatrix} \begin{pmatrix} I_p \\ 0 \end{pmatrix}$$

où $X_\perp$ est une base orthonormale du complément orthogonal de l'espace colonne de X.

Définition C.13 (Rétraction)**.** Une rétraction couramment utilisée sur $\mathrm{St}(n, p)$ est la rétraction QR :

$$R_X(Z) = \mathrm{qf}(X + Z)$$

où $\mathrm{qf}(\cdot)$ dénote le facteur Q de la décomposition QR, garantissant que $R_X(Z) \in \mathrm{St}(n, p)$.

Annexe C

Prétraitement des données pour Faster R-CNN

Ce prétraitement a pour objectif d'adapter les données issues des radargrammes aux exigences du modèle Faster R-CNN. L'ensemble des étapes décrites ici permet de transformer les détections d'hyperboles en un jeu de données annoté au format COCO, directement exploitable par l'implémentation `torchvision`.

A Génération des annotations à partir des détections d'hyperboles

Les objets d'intérêt dans les radargrammes sont identifiés sous forme d'hyperboles détectées automatiquement (voir Chapitre 1.7.2). Chaque hyperbole est traduite par une boîte englobante, et ces informations sont regroupées dans un fichier nommé `position.json`, dont un extrait est proposé dans le listing C.1.

Listing C.1 – Extrait du fichier `position.json`

```
{
    "IRADAR__005.h5": [
        [
            [1107, 1483, 237, 0],
            [1845, 2221, 237, 0],
            [1476, 1852, 624, 387],
            [0, 376, 753, 516]
        ],
        ["EMPTY", "EMPTY", "EMPTY", "ABRI_EN_BOIS"]
    ],
    "IRADAR__006.h5": [
        [
            [1146, 1541, 239, 0],
            [1910, 2305, 239, 0],
            [1528, 1923, 626, 387]
        ],
        ["EMPTY", "EMPTY", "NON_METALLIQUE"]
    ]
}
```

Ce fichier est structuré sous forme de dictionnaire, où :

- chaque clé représente un nom de fichier radar (`.h5`) ;
- chaque valeur est une liste composée d'un ensemble de boîtes englobantes et des classes associées.

Les boîtes englobantes sont définies par leurs coordonnées $(x_{\min}, x_{\max}, y_{\max}, y_{\min})$, et les classes sont exprimées sous forme de chaînes de caractères.

B Liste des classes utilisées

Les différentes classes annotées dans les images radar sont les suivantes :

- `EMPTY` : région ne contenant pas d'objet détecté, utilisée comme classe par défaut dans l'annotation.
- `ABRI_EN_BOIS` : abri souterrain ou structure en bois.
- `NON_METALLIQUE` : objet enfoui non métallique.
- (d'autres classes peuvent être ajoutées selon le contexte).

Remarque importante : la classe `EMPTY` n'a pas été prise en compte dans les simulations. En effet, elle représente des zones sans hyperbole et n'a donc pas vocation à être détectée ou classifiée dans le cadre d'un entraînement supervisé. Seules les classes associées à de véritables objets d'intérêt ont été conservées pour la détection avec Faster R-CNN.

C Conversion vers le format COCO

Pour permettre l'utilisation de Faster R-CNN, les annotations sont converties au format COCO. Les opérations suivantes sont effectuées :

- Conversion des coordonnées en format COCO : $(x_{\min}, y_{\min}, \text{largeur}, \text{hauteur})$.
- Association d'un identifiant numérique unique à chaque classe textuelle.
- Ajout des métadonnées nécessaires (`id`, `iscrowd` $= 0$, `segmentation` vide, etc.).

Chaque image est finalement représentée par un dictionnaire contenant :

- `boxes` : liste de boîtes englobantes au format $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$,
- `labels` : étiquettes des objets annotés (entiers),
- `iscrowd` : indicateur binaire (0 dans notre cas).

D Définition d'un Dataset personnalisé pour PyTorch

Un `Dataset` personnalisé est défini en héritant de `torch.utils.data.Dataset`, afin de charger les images radar avec leurs annotations. La méthode `__getitem__()` retourne, pour chaque indice, une image au format tensor et un dictionnaire contenant les annotations.

E Utilisation d'un DataLoader

Le chargement par lots est assuré via un `DataLoader`, combiné à une fonction `collate_fn()` permettant de gérer la variabilité du nombre d'objets par image.

Grâce à ce pipeline, les données radar peuvent être intégrées sans modification dans une architecture Faster R-CNN, assurant une compatibilité totale avec le format COCO utilisé pour la détection d'objets.

Annexe D

Métriques

A Les métriques IoU, TP, FP, TN et FN dans un contexte de détection

L'**Intersection over Union (IoU)**, ou *indice de Jaccard*, mesure le **chevauchement** entre une boîte englobante prédite et une boîte de vérité terrain (annotation manuelle). Il s'agit du **rapport entre l'aire de l'intersection** des deux boîtes et **l'aire de leur union** :

$$IoU = \frac{\text{Aire(Intersection)}}{\text{Aire(Union)}}$$

Exemple illustré (Figure D.1) La Figure D.1 montre un exemple de détection d'un oiseau dans une image :

- La boîte rouge représente la **vérité terrain** (position réelle de l'objet, étiquetée manuellement) ;
- La boîte bleu ciel représente la **prédiction du modèle**.

Le **chevauchement** entre ces deux boîtes constitue l'*intersection*, tandis que leur surface combinée forme l'*union*. Le score IoU est compris entre 0 (aucun chevauchement) et 1 (chevauchement parfait). En pratique, une détection est considérée comme correcte si l'IoU est supérieure ou égale à 0,5.

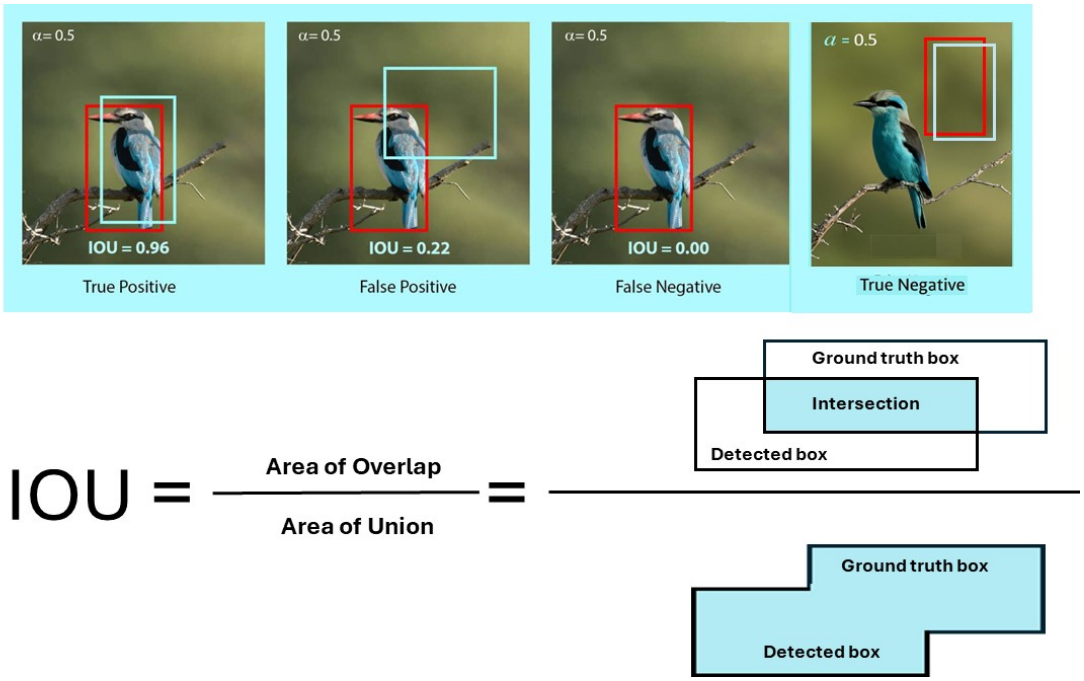


FIGURE D.1 – Exemple de calcul de l'IoU. La boîte rouge représente la vérité terrain (sauf cas particulier Vrai Négatif, la boîte bleue ciel est la prédiction du modèle (Source).

Score de confiance En plus de l'IoU, les modèles de détection d'objets attribuent à chaque prédiction un **score de confiance**, qui reflète :

- la **probabilité** qu'un objet soit présent dans la boîte,
- la **fiabilité** de la localisation de cette boîte.

Ce score est comparé à un **seuil de confiance**, typiquement fixé manuellement. Seules les prédictions dont le score dépasse ce seuil sont conservées pour l'évaluation.

Définitions des cas TP, FP, FN, TN Pour évaluer un modèle de détection, on utilise les métriques de **précision** et de **rappel**, qui reposent sur les cas suivants :

- **Vrai Positif (True Positive, TP)** : une boîte prédite est considérée comme correcte si :
  - son **score de confiance** dépasse le seuil, *et*
  - son **IoU** avec la boîte de vérité terrain est supérieur au seuil d'IoU.

  Cela signifie que l'objet est correctement détecté et bien localisé.
- **Faux Positif (False Positive, FP)** : une boîte prédite est incorrecte dans les cas suivants :
  - Aucun objet correspondant n'existe dans la vérité terrain (IoU = 0) ;
  - L'IoU est inférieur au seuil requis ;
  - L'étiquette de classe est incorrecte, même si la boîte est bien placée.
- **Faux Négatif (False Negative, FN)** : un objet est bien présent dans l'image, mais n'est **pas détecté** (le modèle n'a fait aucune prédiction valide pour cet objet).
- **Vrai Négatif (True Negative, TN)** : ce cas est rarement utilisé dans les métriques de détection, car il concerne des zones sans objet. Il correspond à :
  - l'absence de boîte prédite,
  - en l'absence également de vérité terrain.

  Remarque : dans ce cas, l'IoU n'est pas défini car aucune boîte n'existe pour le calcul.

B Évaluation d'un modèle Faster R-CNN

L'évaluation d'un modèle de détection d'objets repose sur des métriques qui permettent de quantifier la qualité des prédictions effectuées. L'une des plus répandues est la **précision moyenne** ou *Average Precision* (AP), qui résume les performances d'un modèle en combinant deux éléments fondamentaux : la *précision* (le taux de bonnes détections parmi toutes les détections) et le *rappel* (le taux de bonnes détections parmi tous les objets présents).

Pour calculer l'AP, on commence par construire une **courbe précision-rappel**, obtenue en faisant varier un seuil de score de confiance appliqué aux prédictions du modèle. À chaque seuil, on mesure la précision et le rappel, ce qui donne une série de points. Toutefois, cette courbe peut être irrégulière ou bruyante, en particulier à cause des prédictions mal classées. C'est pourquoi on applique une **interpolation** pour la lisser.

L'interpolation consiste à remplacer, pour chaque niveau de rappel r, la précision réelle par la meilleure précision obtenue pour un rappel supérieur ou égal à r. La formule de cette interpolation est la suivante :

$$p_{\text{interp}}(r) = \max_{\tilde{r} \geq r} p(\tilde{r})$$

Une fois la courbe lissée, l'aire sous la courbe interpolée représente la précision moyenne (AP). Cette aire peut être calculée à l'aide de la formule suivante :

$$AP = \sum_{i=1}^{n} (r_i - r_{i-1}) \cdot p_{\text{interp}}(r_i) \tag{D.1}$$

Dans certains benchmarks, comme *Pascal VOC 2007*, une version simplifiée de cette méthode, appelée **interpolation à 11 points**, est utilisée. Elle consiste à évaluer la précision interpolée à 11 niveaux de rappel également espacés entre 0 et 1, puis à en faire la moyenne :

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} p_{\text{interp}}(r)$$

Les benchmarks les plus récents, tels que COCO, utilisent une interpolation plus fine à **101 points**, avec des rappels variant de 0 à 1 par pas de 0.01.

Enfin, pour obtenir une évaluation globale des performances d'un modèle sur l'ensemble des classes d'objets, on calcule la moyenne des AP sur toutes les classes. Cette moyenne est appelée **moyenne des précisions moyennes** ou *mean Average Precision* (mAP) :

$$mAP = \frac{1}{k} \sum_{i=1}^{k} AP_i$$

où k est le nombre total de classes et AP_i est la précision moyenne pour la classe i.

Il est également courant d'évaluer les performances d'un modèle à l'aide des formules classiques de précision, de rappel et de F1-score pour chaque classe :

$$P_i = \frac{TP_i}{TP_i + FP_i}$$
$$R_i = \frac{TP_i}{TP_i + FN_i}$$
$$F1_i = \frac{2 \cdot P_i \cdot R_i}{P_i + R_i}$$

La précision globale du modèle, aussi appelée *accuracy*, est calculée comme suit :

$$\text{Accuracy} = \frac{\sum_{j=1}^{n} p_{jj}}{\sum_{j=1}^{n} \sum_{k=1}^{n} p_{jk}}$$

Exemple de calcul détaillé de l'AP Considérons un exemple pour une seule classe. Le tableau suivant présente une série de couples (r_i, p_i) représentant le rappel et la précision obtenus à différents seuils :

Rappel (r)	**Précision** (p)
0.10	0.80
0.15	0.60
0.20	0.75
0.35	0.70
0.35	0.50
0.40	0.55
0.50	0.40
0.75	0.45
0.90	0.30
0.92	0.35

TABLE D.1 – Points de rappel et précision bruts

Pour lisser cette courbe, nous appliquons l'interpolation définie plus haut. On obtient alors les valeurs suivantes :

Rappel (r)	**Précision interpolée** $p_{\textbf{interp}}(r)$
0.10	0.80
0.15	0.75
0.20	0.75
0.35	0.70
0.40	0.55
0.50	0.45
0.75	0.45
0.90	0.35
0.92	0.35

TABLE D.2 – Précisions interpolées pour les niveaux de rappel

On peut maintenant appliquer la formule (D.1) pour calculer l'AP :

$$
\begin{aligned}
(0.15 - 0.10) \cdot 0.80 &= 0.04 \\
(0.20 - 0.15) \cdot 0.75 &= 0.0375 \\
(0.35 - 0.20) \cdot 0.75 &= 0.1125 \\
(0.40 - 0.35) \cdot 0.55 &= 0.0275 \\
(0.50 - 0.40) \cdot 0.55 &= 0.055 \\
(0.75 - 0.50) \cdot 0.45 &= 0.1125 \\
(0.90 - 0.75) \cdot 0.45 &= 0.0675 \\
(0.92 - 0.90) \cdot 0.35 &= 0.007
\end{aligned}
$$

En additionnant tous les termes :

$$AP = 0.04 + 0.0375 + 0.1125 + 0.0275 + 0.055 + 0.1125 + 0.0675 + 0.007 = \boxed{0.4595}$$

Comparaison avec l'AP à 11 points En appliquant la méthode Pascal VOC à 11 points sur les mêmes données (en prenant la précision interpolée à 11 niveaux de rappel), on obtient :

$$AP = \frac{2 \cdot 0.8 + 0.75 + 0.7 + 0.55 + 3 \cdot 0.45 + 2 \cdot 0.35 + 0}{11} = \frac{5.65}{11} = \boxed{0.5136}$$

Conclusion La métrique AP (et par extension, mAP) constitue un indicateur robuste pour évaluer la performance d'un modèle de détection. Elle prend en compte à la fois la capacité du modèle à détecter tous les objets pertinents (rappel) et à éviter les fausses détections (précision). Des mesures complémentaires, comme le F1-score et l'accuracy permettent d'avoir une vue plus complète sur la qualité des résultats produits.

Annexe E

Liste des acronymes

ACP Analyse en Composantes Principales

AP Average Precision

AR Average Recall

BiMap Bilinear Mapping

CDF Cumulative Distribution Function

COCO Common Objects in COntext

CNN Convolutional Neural Network

CPEV Cumulative Proportion of Explained Variance

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DL Deep Learning

EVD EigenValue Decomposition

EM Electromagnétique

FC Fully Connected

FN False Negative

FP False Positive

FPN Feature Pyramid Network

GAP Global Average Pooling

GLRT Generalized Likelihood Ratio Test

GPS Global Positioning System

GPR Ground Penetrating Radar

GSSI Geophysical Survey Systems, Inc.

IoU Intersection over Union

LogEig Logarithm of Eigenvalues

mAP mean Average Precision

mAR mean Average Recall

MIMO Multiple-Input Multiple-Output

ML Machine Learning

MLE Maximum Likelihood Estimation

MPN-COV Matrix Power Normalized COVariance pooling

NMF Normalized Matched Filter

NMS Non-Maximum Suppression

OvA One vs All

PEHD Polyéthylène haute densité

PVC Polychlorure de vinyle

RCNet Residual Covariance Network

R-CNN Region-based Convolutional Neural Network

RC R-CNN Residual Covariance Region-based Convolutional Neural Network

ReEig Rectification of Eigenvalues

ReLU Rectified Linear Unit

ResNet Residual Network

RF Random Forest

RFT ResNet34 fine-tuned

RGB Red-Green-Blue

RoI Region of Interest

RPN Region Proposal Network

RRT ResNet34 retrained

RTK Real-Time Kinematic

RSB Rapport Signal sur Bruit

SAR Synthetic Aperture Radar

SCM Sample Covariance Matrix

S-CNN Shallow Convolutional Neural Network

SGD Stochastic Gradient Descent

SP90 Spectra Precision 90

SPD Symmetric Positive Definite

SPDNet Riemannian Symmetric Positive Definite Matrix Network

SRCNet Stacked Residual Covariance Network

SVC Support Vector Classifier

SVD Singular Value Decomposition

SVM Support Vector Machine

TN True Negative

TP True Positive

Bibliographie

[1] Q. Hoarau, G. Ginolhac, A. M. Atto, and J. M. Nicolas, "Robust adaptive detection of buried pipes using GPR," *Signal Processing*, vol. 132, pp. 293–305, Mar. 2017. iii, 1, 21, 22, 77

[2] D. J. Daniels, *Ground Penetrating Radar*. IEE, 2004. viii, 7

[3] F. Giovanneschi, K. V. Mishra, M. A. Gonzalez-Huici, Y. C. Eldar, and J. H. G. Ender, "Dictionary learning for adaptive gpr landmine classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 12, pp. 10036–10055, 2019. viii

[4] G. Terrasse, J.-M. Nicolas, E. Trouvé, and Drouet, "Sparse decomposition of the GPR useful signal from hyperbola dictionary," in *2016 24th European Signal Processing Conference (EUSIPCO)*, pp. 2400–2404, 2016. viii, 21

[5] M. Gallet, A. Mian, G. Ginolhac, E. Ollila, and N. Stelzenmuller, "New robust sparse convolutional coding inversion algorithm for ground penetrating radar images," *IEEE Transactions on Geoscience and Remote Sensing*, 2023. viii

[6] H. Zhou, X. Feng, Y. Zhang, E. Nilot, M. Zhang, Z. Dong, and J. Qi, "Combination of support vector machine and h-alpha decomposition for subsurface target classification of gpr," in *2018 17th International Conference on Ground Penetrating Radar (GPR)*, pp. 1–4, IEEE, 2018. viii, 53

[7] W. Shao, A. Bouzerdoum, S. L. Phung, L. Su, B. Indraratna, and C. Rujikiatkamjorn, "Automatic classification of gpr signals," in *Proceedings of the XIII Internarional Conference on Ground Penetrating Radar*, pp. 1–6, IEEE, 2010. viii, 53

[8] M. Almaimani, D. Wu, Y. Liang, L. Yang, D. Huston, and T. Xia, *Classifying GPR Images Using Convolutional Neural Networks*. Jan. 2018. viii, 58, 62, 78, 89, 90

[9] H. Zhou, X. Feng, Y. Zhang, E. Nilot, M. Zhang, Z. Dong, and J. Qi, "Combination of Support Vector Machine and H-Alpha Decomposition for Subsurface Target Classification of GPR," in *2018 17th International Conference on Ground Penetrating Radar (GPR)*, pp. 1–4, June 2018. ISSN : 2474-3844. viii

[10] M. Elsaadouny, J. Barowski, and I. Rolfes, "ConvNet Transfer Learning for GPR Images Classification," in *2020 German Microwave Conference (GeMiC)*, pp. 21–24, Mar. 2020. ISSN : 2167-8022. viii

[11] M. Gallet, A. Mian, G. Ginolhac, and N. Stelzenmuller, "Classification of GPR signals via covariance pooling on CNN features within a riemannian framework," in *IGARSS 2022*, pp. 365–368, IEEE, 2022. viii, 57

[12] Z. Huang and L. Van Gool, "A riemannian network for SPD matrix learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017. ix, x, 76, 78, 80, 83

[13] P. Li, J. Xie, Q. Wang, and W. Zuo, "Is second-order information helpful for large-scale visual recognition ?," in *Proceedings of the IEEE ICCV*, pp. 2070–2078, 2017. ix, 106, 107

[14] P. Li, J. Xie, Q. Wang, and Z. Gao, "Towards faster training of global covariance pooling networks by iterative matrix square root normalization," in *Proceedings of the IEEE CVPR*, pp. 947–955, 2018. ix, 107

[15] D. Brooks, O. Schwander, F. Barbaresco, J.-Y. Schneider, and M. Cord, "Riemannian batch normalization for SPD neural networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019. ix, 76, 82, 86

[16] D. Jafuno, A. Mian, G. Ginolhac, and N. Stelzenmuller, "Classification d'objets enfouis par un modèle du second d'ordre en utilisant des données gpr," in *GRETSI*, 2023. xi

[17] D. Jafuno, A. Mian, G. Ginolhac, and N. Stelzenmuller, "Buried object classification from gpr data by using second order deep learning models," in *IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium*, pp. 6513–6516, IEEE, 2024. xi, 77

[18] D. Jafuno, A. Mian, G. Ginolhac, and N. Stelzenmuller, "Classification of buried objects from ground penetrating radar images by using second order deep learning models," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2025. xi

[19] A. Ramadhan, E. Ali, and A. Pramudita, "The effect ricker wavelet of duty cycle adjustment on gpr detection result," *JMECS (Journal of Measurements, Electronics, Communications, and Systems)*, vol. 8, no. 1, pp. 17–22, 2021. 6

[20] Y. Wang, "Frequencies of the Ricker wavelet," *Geophysics*, vol. 80, pp. A31–A37, 02 2015. 6

[21] A. Annan, "Ground-penetrating radar," 2005. 7

[22] M. E. Everett, *Near-surface applied geophysics*. Cambridge University Press, 2013. 10, 12

[23] Geophysics for Practicing Geoscientists, "Ground penetrating radar (gpr), survey and data," 2024. 11, 13

[24] C. Fauchard, P. Pothérat, P. Côte, and M. Mudet, "Détection de cavités souterraines par méthodes géophysiques," *Guide technique, Laboratoire Central des Ponts et Chaussées*, 2004. 12

[25] M. G. Amin and F. Ahmad, *Through-the-Wall Radar Imaging : Theory and Applications*. E-reference Signal Processing, Elsevier, 2013. 19

[26] C. P. Kao, J. Li, Y. Wang, H. Xing, and C. Liu, "Measurement of layer thickness and permittivity using a new multilayer model from gpr data," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 45, no. 8, pp. 2463–2470, 2007. 19

[27] Z. L. Huang and J. Zhang, "Determination of parameters of subsurface layers using gpr spectral inversion method," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 52, no. 12, pp. 7527–7533, 2014. 19, 20

[28] J. F. Valerio and P. E. S. S. Filho, "Enhancing gpr data using image processing techniques," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 47, no. 2, pp. 659–666, 2009. 19

[29] R. Wu, K. Gu, J. Li, M. Bradley, J. Habersat, and G. Maksymonko, "Propagation velocity uncertainty on gpr sar processing," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 39, no. 3, pp. 849–861, 2003. 20

[30] B. Recht, W. Xu, and B. Hassibi, "Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization," in *2008 47th IEEE Conference on Decision and Control*, pp. 3065–3070, 2008. 21

[31] A. M. Plattner, "Gprpy : Open-source ground-penetrating radar processing and visualization software," *The Leading Edge*, vol. 39, no. 5, pp. 332–337, 2020. 28

[32] C. Warren, A. Giannopoulos, and I. Giannakis, "gprmax : Open source software to simulate electromagnetic wave propagation for ground penetrating radar," *Computer Physics Communications*, vol. 209, pp. 163–170, 2016. 31

[33] M. Almaimani, "Classifying GPR images using convolutional neural networks," 2018. 47, 60, 61

[34] M. Zhang, X. Feng, Y. Zhang, E. Nilot, Z. Dong, and H. Zhou, "Full-polarimetric ground penetrating radar underground objects classification using random forest," in *2018 17th International Conference on Ground Penetrating Radar (GPR)*, pp. 1–5, IEEE, 2018. 53

[35] N. Barkataki, S. Mazumdar, R. Talukdar, P. Chakraborty, B. Tiru, and U. Sarma, "Prediction of size of buried objects using ground penetrating radar and machine learning techniques," in *2020 International Conference on Computational Performance Evaluation (ComPE)*, pp. 781–785, IEEE, 2020. 53

[36] I. T. Jolliffe, *Principal Component Analysis*. Springer Series in Statistics, New York : Springer-Verlag, 2 ed., 2002. 53

[37] C. Cortes and V. Vapnik, "Support-vector networks," in *Machine Learning*, vol. 20, pp. 273–297, Springer, 1995. 54

[38] B. Schölkopf and A. J. Smola, *Learning with Kernels : Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, 2002. 54

[39] J. Weston and C. Watkins, "Multi-class support vector machines," tech. rep., Technical Report CSD-TR-98-04, Department of Computer Science, Royal ..., 1998. 54

[40] T. K. Ho, "Random decision forests," in *Proceedings of 3rd international conference on document analysis and recognition*, vol. 1, pp. 278–282, IEEE, 1995. 54

[41] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001. 54

[42] S. Akodad, L. Bombrun, J. Xia, Y. Berthoumieu, and C. Germain, "Ensemble learning approaches based on covariance pooling of CNN features for high resolution remote sensing scene classification," *Remote Sensing*, vol. 12, no. 20, p. 3292, 2020. 57, 124

[43] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Classification of covariance matrices using a Riemannian-based kernel for BCI applications," *Neurocomputing*, vol. 112, pp. 172–178, 2013. 57

[44] Antoine Collas, Florent Bouchard, Guillaume Ginolhac, Arnaud Breloy, and Jean-Philippe Ovarlez, "On the Use of Geodesic Triangles Between Gaussian Distributions for Classification Problems," *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2022. 57

[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012. 63, 64, 65, 74

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. 63, 64, 65, 67, 74

[47] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2 : Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018. 63, 64, 67, 74

[48] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet : A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009. 63

[49] A. G. Howard, "Mobilenets : Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv :1704.04861*, 2017. 67, 124

[50] P. Enkvetchakul and O. Surinta, "Effective data augmentation and training techniques for improving deep learning in plant leaf disease recognition," *Applied Science and Engineering Progress*, vol. 15, no. 3, pp. 3810–3810, 2022. 69

[51] E. Ollila, D. E. Tyler, V. Koivunen, and H. V. Poor, "Complex elliptically symmetric distributions : Survey, new results and applications," *IEEE Transactions on Signal Processing*, vol. 60, no. 11, pp. 5597–5625, 2012. 77, 124

[52] S. Akodad, L. Bombrun, M. Puscasu, J. Xia, C. Germain, and Y. Berthoumieu, "Deep Ensemble Learning Model Based on Covariance Pooling of Multi-Layer CNN Features," in *2022 IEEE International Conference on Image Processing (ICIP)*, (Bordeaux, France), pp. 1081–1085, IEEE, Oct. 2022. 77

[53] M. Gallet, A. Mian, G. Ginolhac, and N. Stelzenmuller, "Classification of GPR signals via covariance pooling on CNN features within a riemannian framework," in *IGARSS 2022 - 2022 IEEE International Geoscience and Remote Sensing Symposium*, pp. 365–368, 2022. 77

[54] P. Li, J. Xie, Q. Wang, and W. Zuo, "Is second-order information helpful for large-scale visual recognition ?," in *Proceedings of the IEEE international conference on computer vision*, pp. 2070–2078, 2017. 77

[55] O. Ledoit and M. Wolf, "A well-conditioned estimator for large-dimensional covariance matrices," *Journal of multivariate analysis*, vol. 88, no. 2, pp. 365–411, 2004. 79

[56] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout : A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014. 81

[57] C. Ionescu, O. Vantzos, and C. Sminchisescu, "Training deep networks with structured layers by matrix backpropagation," 2015. 82, 83

[58] N. Boumal, *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023. 84, 127, 128, 129

[59] A. Collas, A. Breloy, G. Ginolhac, C. Ren, and J.-P. Ovarlez, "Robust geometric metric learning," in *Proceedings of EUSIPCO 2022*, Aug. 2022. 90

[60] A. Collas, A. Breloy, C. Ren, G. Ginolhac, and J.-P. Ovarlez, "Riemannian optimization for non-centered mixture of scaled gaussian distributions," *IEEE Transactions on Signal Processing*, vol. 71, pp. 2475–2490, 2023. 92

[61] M.-T. Pham and S. Lefèvre, "Buried object detection from b-scan ground penetrating radar data using faster-rcnn," in *IGARSS 2018-2018 IEEE international geoscience and remote sensing symposium*, pp. 6804–6807, IEEE, 2018. 98

[62] Z. Fang, Z. Shi, X. Wang, and W. Chen, "Roadbed defect detection from ground penetrating radar b-scan data using faster rcnn," in *IOP Conference Series : Earth and Environmental Science*, vol. 660, p. 012020, IOP Publishing, 2021. 98

[63] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014. 98, 99

[64] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," in *International Journal of Computer Vision (IJCV)*, vol. 104, pp. 154–171, Springer, 2013. 98

[65] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015. 98, 99

[66] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster r-cnn : Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015. 99

[67] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2117–2125, 2017. 100

[68] C. Ionescu, O. Vantzos, and C. Sminchisescu, "Training deep networks with structured layers by matrix backpropagation," *arXiv preprint arXiv :1509.07838*, 2015. 107

[69] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco : Common objects in context," in *Computer vision–ECCV 2014 : 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pp. 740–755, Springer, 2014. 109

[70] K. Chen, W. Lin, J. Li, J. See, J. Wang, and J. Zou, "Ap-loss for accurate one-stage object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 3782–3798, 2020. 111

[71] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, pp. 303–338, 2010. 115

[72] Z. Cai and N. Vasconcelos, "Cascade r-cnn : Delving into high quality object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6154–6162, 2018. 124

[73] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra r-cnn : Towards balanced learning for object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 821–830, 2019. 124

[74] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017. 124

[75] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer : Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021. 124

[76] M. Tan and Q. Le, "Efficientnet : Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, pp. 6105–6114, PMLR, 2019. 124

[77] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet : An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6848–6856, 2018. 124

[78] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy, "Joint distribution optimal transportation for domain adaptation," *Advances in neural information processing systems*, vol. 30, 2017. 125

[79] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *International conference on machine learning*, pp. 97–105, PMLR, 2015. 125