

Rapport : Application de l'IA pour la Reconnaissance des Motifs



Projet de spécialité

Riham Faraj - Zakaria El Kazdam - Paul Leroux - Malak Charki

Ensimag

Encadré par Jean Sébastien Franco

Résumé

La détection précise d'objets dans les images revêt une importance capitale dans de nombreux domaines, de la surveillance à la conduite autonome. Cependant, malgré les avancées significatives dans le domaine de la vision par ordinateur, plusieurs défis persistent, notamment la détection robuste d'objets dans des scénarios complexes. Dans ce contexte, notre travail se concentre sur le développement d'un système de détection d'objets capable de localiser avec précision une seule instance d'objet dans une image, avec un accent particulier sur les classes de voitures et de motos. Nous proposons une approche basée sur des architectures de réseaux de neurones convolutionnels (CNN) avancées, notamment ResNet ..., pour résoudre cette tâche de détection d'objet. Notre solution repose sur l'utilisation d'un ensemble de données annoté spécifiquement pour les images ne contenant qu'une seule instance d'objet. Nous avons entraîné plusieurs modèles de détection d'objets sur cet ensemble de données et évalué leurs performances en utilisant des métriques telles que l'intersection sur union (IoU) et la précision de classification. Les résultats expérimentaux montrent que notre approche est similaire aux méthodes existantes sachant que notre modèle est moins complexe. Nous avons obtenu une précision de 1 et un IoU de 0.77, démontrant ainsi l'efficacité de notre système pour cette tâche spécifique. Ce travail apporte une contribution significative à la résolution du problème de la détection d'objets uniques dans des scénarios réels, ouvrant la voie à des applications pratiques dans des domaines tels que la reconnaissance de panneaux de signalisation, la surveillance de la circulation et bien d'autres encore.

1 Glossaire

Accuracy :

l'accuracy est une mesure permettant d'évaluer à quel point un modèle est correct pour la classification. Il représente le taux de prédictions correctes.

IoU :

une mesure couramment utilisée pour évaluer la précision des modèles de détection d'objets. Il est calculé en prenant le rapport de l'aire de l'intersection entre la prédiction de la bounding box et la vérité terrain (ground truth), sur l'aire de leur union. En pratique, un IoU supérieur à un seuil spécifique (par exemple, 0.5 ou 0.75) est souvent utilisé pour déterminer si une prédiction est considérée comme correcte ou non.

ResNet :

ResNet, ou Réseaux Résiduels, est une architecture de réseau de neurones profonds. L'idée principale derrière ResNet est l'introduction de connexions résiduelles, ou shortcuts, qui permettent de sauter des couches dans le réseau. Contrairement aux architectures traditionnelles où chaque couche apprend une représentation de l'entrée, dans ResNet, les couches apprennent les résidus, ou les écarts, par rapport à la représentation apprise par les couches précédentes.

GPU :

Les GPU (Graphics Processing Units) sont des processeurs spécialisés conçus pour effectuer rapidement des calculs sur des données graphiques et parallèles. Initialement utilisés principalement pour les tâches graphiques, tels que le rendu de jeux vidéo et la modélisation 3D.

SVM :

Le SVM (Support Vector Machine) est un algorithme d'apprentissage supervisé utilisé pour la classification et la régression. En classification, l'objectif du SVM est de trouver l'hyperplan qui

sépare au mieux les données en deux classes différentes. Cet hyperplan est choisi de manière à maximiser la marge, c'est-à-dire la distance entre l'hyperplan et les points de données les plus proches de chaque classe, appelés vecteurs de support.

HogFeatures :

Les caractéristiques HOG (Histogram of Oriented Gradients) sont une technique de description d'image largement utilisée en vision par ordinateur pour la détection et la classification d'objets. Cette méthode repose sur l'extraction des gradients d'intensité dans des régions locales de l'image. Cet algorithme permet d'avoir en sortie les "features" décrivant l'image donnée en entrée.

VGG11 : Architecture de réseau de neurones convolutionnels profonds conçue pour la reconnaissance et la classification d'images. Il a été développé par le Visual Geometry Group de l'Université d'Oxford. VGG est connu pour sa simplicité et son architecture uniforme, composée de multiples couches de convolution suivies de couches de max-pooling, et se terminant par des couches entièrement connectées.

Initialisation Xavier Uniforme : également connue sous le nom d'initialisation Xavier, est une méthode pour initialiser les poids des neurones dans les réseaux de neurones artificiels. Cette méthode vise à maintenir la variance des activations des neurones à peu près constante à travers différentes couches du réseau, ce qui favorise la stabilité de l'apprentissage. L'initialisation Xavier Uniforme tire les poids des neurones à partir d'une distribution uniforme centrée sur zéro.

Max Pooling :

Le max pooling est une opération de réduction de dimensionnalité couramment utilisée dans les réseaux de neurones convolutifs (CNN). Il consiste à diviser une image ou une carte de caractéristiques en régions non disjointes et à ne conserver que la valeur maximale (le "maximum") de chaque région.

Flattening :

Le flattening est une opération utilisée dans les réseaux de neurones, en particulier dans les architectures de réseaux entièrement connectés (ou "fully connected"). Cette opération consiste à transformer une matrice ou un tenseur multidimensionnel en un vecteur unidimensionnel, en "aplatissant" toutes les dimensions sauf la première.

Couche dense :

Une couche dense, dans un réseau de neurones artificiels, est une couche où chaque neurone est connecté à chaque neurone de la couche précédente, formant ainsi un réseau entièrement connecté.

Loss :

La loss, ou fonction de perte, est une mesure utilisée pour évaluer à quel point les prédictions d'un modèle sont éloignées des valeurs réelles lors de l'apprentissage supervisé. Pour la perte de cross-entropy en classification binaire :

$$\text{Cross-Entropy} = -\frac{1}{n} \sum_{i=1}^n (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$$

Où :

- y_i est la vraie étiquette de l'échantillon i ,
- \hat{y}_i est la prédiction du modèle pour l'échantillon i ,

— n est le nombre total d'échantillons dans le jeu de données.

Epoch :

Une époque (epoch) dans le contexte de l'apprentissage automatique, en particulier dans l'entraînement des réseaux de neurones, fait référence à une seule itération complète de l'ensemble de données d'entraînement à travers le modèle. Cela signifie qu'une époque est atteinte lorsque chaque exemple d'entraînement a été utilisé une fois pour mettre à jour les poids du modèle.

Techniques d'augmentation :

Cette augmentation est classiquement réalisée en effectuant des opérations modifiant l'aspect de l'image, sans pour autant en modifier la sémantique : par exemple, en changeant la luminosité, en effectuant une rotation / un effet miroir, en changeant l'échelle, ou encore en ajoutant du bruit.

Table des matières

Résumé	1
1 Glossaire	1
Glossaire	1
2 Introduction	5
3 Description conceptuelle	5
4 Présentation de la solution	5
4.1 Ressources	5
4.2 Méthodes et Outils	6
5 Résultat	9
6 Discussion	11
6.1 Synthèse et interprétation des résultats	11
6.2 Points à retenir et limites	11
6.3 Pistes à explorer	11
7 Conclusion	12

2 Introduction

L'avènement de l'intelligence artificielle et plus particulièrement des réseaux de neurones convolutionnels (CNN) a bouleversé le champ de la vision par ordinateur, ouvrant de nouvelles perspectives pour la localisation d'un objet dans une image. Contrairement aux méthodes traditionnelles basées sur le traitement des textures ou des contours, les CNN exploitent la capacité de modéliser des abstractions plus élevées dans les données visuelles. Ce projet se penche sur l'exploitation de CNN avancés pour améliorer la localisation précise d'objets, notamment les voitures et les motos, dans diverses conditions d'images, en utilisant des architectures profondes telles que ResNet.

Nous abordons ici le défi de développer un modèle capable de localiser précisément une instance unique, en mettant l'accent sur l'efficacité et la rapidité nécessaires. L'objectif est d'optimiser les performances de détection tout en minimisant les erreurs et les coûts computationnels associés à des architectures de réseau plus complexes.

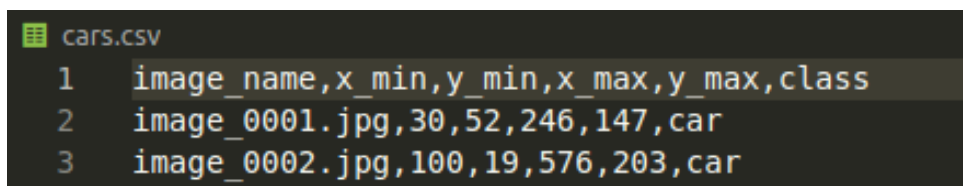
3 Description conceptuelle

Le défi qui se présente concerne la classification et la localisation des diverses catégories d'objets liés à la route, comme les voitures, les motos, etc., en utilisant des boîtes englobantes (aussi appelées bounding box). Cela nous permettra de déterminer si le modèle arrive bien à situer l'objet dans l'image. Dans ce projet, notre objectif principal est donc d'explorer une large gamme de méthodes afin de créer et de classifier différents modèles adaptés à cette tâche spécifique en faisant un benchmark. Nous prévoyons ainsi de concevoir une série de réseaux de neurones, avec des architectures variant en termes de profondeur et de complexité. De plus, afin d'aller plus loin, nous avons exploré des techniques de fine-tuning de différents modèles (Resnet) afin de les adapter spécifiquement à notre jeu de données et à notre problème. Le fine-tuning consiste à utiliser les poids d'un modèle déjà existant et de le réentraîner sur des nouvelles données afin de le rendre plus efficace sur une tâche précise. C'est pour cela qu'une grosse partie de notre approche consistait à effectuer une revue exhaustive de la littérature dans l'optique de comprendre l'état de l'art actuel dans le domaine de la détection d'objets.

4 Présentation de la solution

4.1 Ressources

1. Données : Nous avons utilisé un ensemble de données qu'on a collecté de différentes base de données ; se constituant d'images avec leurs annotations qui sont représentés par des fichiers .csv, contenant les informations suivantes :



```
cars.csv
1 image_name,x_min,y_min,x_max,y_max,class
2 image_0001.jpg,30,52,246,147,car
3 image_0002.jpg,100,19,576,203,car
```

FIGURE 1 – Annotations des images

L'organisation de nos données respecte la forme suivante :

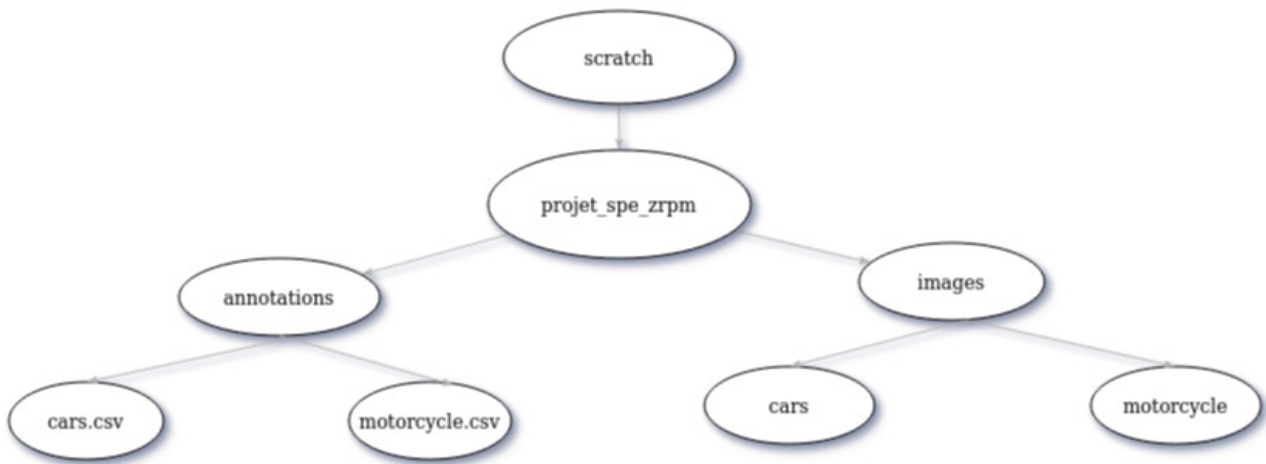


FIGURE 2 – organisation des données

2. Environnement de développement : Le projet a été développé en utilisant Python comme langage de programmation principal. Nous avons également utilisé des bibliothèques telles que TensorFlow et PyTorch pour implémenter les modèles de détection d'objets, Scikit-learn pour certaines tâches de prétraitement et d'évaluation, ainsi que Cv2 pour le traçage des boîtes englobantes lors de la vérification et la prédiction.
3. Matériel : Les expériences ont été menées sur des machines de processeur normal pas très performant, mais on exécutait le code sur des GPU.

4.2 Méthodes et Outils

1. Modèles initiaux :

- Modèle SVM avec HogFeatures : Dans un premier temps, on a décidé de découvrir l'état de l'art dans le domaine de la vision par ordinateur en combinant les HogFeatures des images pour extraire les "features" qu'on va utiliser après pour le modèle SVM qui est dédié uniquement à la première tâche qui est la classification des images. Nous avons entraîné le modèle sur le contenu des bounding box puis fait le test aussi sur le contenu des bounding box et nous avons obtenu une précision de 0.7, or dans la réalité nous n'avons pas de bounding box pour pouvoir classifier leur contenus. Malheureusement, lorsque nous faisons le test sur des images sans bounding box, nous obtenons seulement 0.5 ce qui n'est pas satisfaisant. De plus, la méthode des svm présente plusieurs limites : cette méthode est sensible aux données mal étiquetées ce qui peut entraîner une mauvaise classification, et elle peut aussi devenir lente à entraîner lorsque le nombre de données est important. Les hog features présentent aussi des limites : ils ne sont pas invariants par rotation ni par l'échelle. Si l'objet dans l'image est présenté selon une direction ou une taille différente, alors la méthode ne sera pas efficace. De plus, les hog features sont aussi sensibles au bruit. Avec des images de mauvaise qualité, cela peut alors conduire à une performance réduite. Pour toutes ces raisons, nous ne pouvons pas garder ce modèle et sommes obligé d'en trouver un nouveau.
- Modèle CNN simple : Pour commencer, on a construit un modèle CNN (Convolutional Neural Network) basique pour comprendre un peu plus comment le modèle se comporte face à notre Data et comment on peut l'améliorer. Nous avons utilisé une couche de convolution avec 6 filtres dans la première couche et puis 3 filtres dans la

deuxième couche. Le pas de convolution a été fixé à 4 pour la première couche, tandis que les valeurs par défaut ont été utilisées pour la deuxième couche, en plus, on a utilisé une couche de pooling (*MaxPooling*), une couche de flattening et enfin une couche dense. Le modèle a obtenu une accuracy de 41.46% sur l'ensemble de test, démontrant ainsi sa capacité de classer les images, mais cette valeur est faible et ne représente pas totalement les performances du modèle.

- **Modèle AlexNet** : Nous avons découvert ensuite AlexNet, une architecture CNN qui a concouru dans le défi ImageNet de 2012 et a surpassé les autres réseaux de manière significative. Un modèle vieux de douze ans pourrait ne pas sembler pertinent pour nous maintenant, mais la performance de l'architecture sur un ensemble de données avec plus d'un million d'images réparties sur 1 000 classes a depuis influencé la conception de nombreux modèles d'apprentissage profond en vision par ordinateur. L'architecture du modèle d'AlexNet est la suivante : 5 couche de convolution + 2 couches dense + une couche pour le flattening. Mais avant d'utiliser ce modèle, on était obligé de prétraiter notre data, en effet lorsqu'on travaille avec un ensemble de données comportant plusieurs caractéristiques, ces caractéristiques peuvent avoir des plages de valeurs différentes. Une caractéristique pourrait avoir un impact disproportionné sur les performances du modèle par rapport à une autre. Nous standardisons nos caractéristiques d'entrée afin que toutes puissent être sur la même échelle, réduisant ainsi tout biais potentiel que le modèle pourrait avoir envers une caractéristique particulière. Après avoir implémenté ce modèle, on a obtenu une loss égale à 0.456 et une précision de 0.852, ce qui n'est pas négligeable pour une tâche de classification. Le graphe ci-dessus représente le loss calculé sur le jeu de données de l'entraînement et celui sur le jeu de données de la validation en fonction du nombre d'epoch :

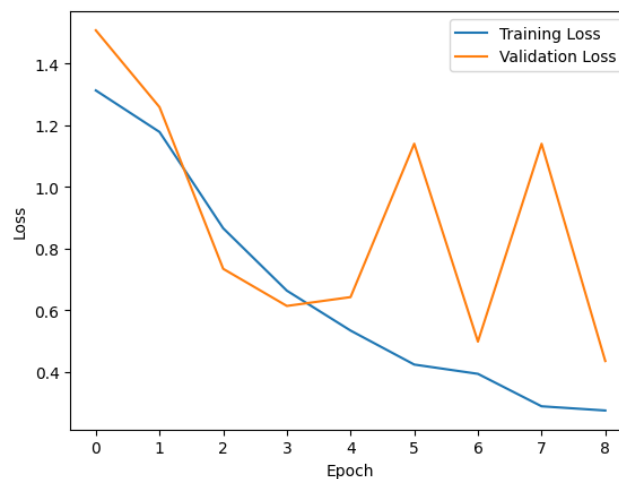


FIGURE 3 – La Loss en fonction du nombre d'epochs

2. Solution finale :

Les modèles mentionnés précédemment performaient d'une manière plus ou moins satisfaisante au niveau de la classification, en particulier le modèle d'AlexNet. Cependant, notre but n'est point limité à la classification, mais plutôt à la localisation d'un objet au sein des images aussi. Pour cela, on construira trois architectures différentes pour réaliser ces deux tâches :

Initialisation des Poids et Normalisation par Lots :

- **Initialisation des Poids** : Avant la mise en place du réseau, une fonction d'initialisation des poids est déterminée, qui utilise la méthode d'initialisation Xavier Uniforme pour régler les poids des différentes couches du réseau de façon adéquate afin de favoriser la convergence lors de la phase d'entraînement..
- **Normalisation par Lots (Batch Normalization)** : Tout au long du chemin des caractéristiques, des couches de Batch Normalization sont utilisées après chaque couche de convolution. La normalisation par lots normalise les activations en entrée de chaque couche, ce qui accélère l'entraînement et améliore la stabilité du modèle.

SimpleDetector :

SimpleDetector est un réseau de neurones convolutifs (CNN) inspiré de VGG11, conçu pour la classification et la localisation d'objets dans les images. Il comprend trois couches convolutionnelles suivies de fonctions d'activation ReLU, ainsi qu'une série de couches entièrement connectées pour la classification, et une branche de régression pour la localisation des objets.

DeeperDetector :

DeeperDetector est une architecture de CNN plus profonde, développée pour des tâches de classification et de localisation d'objets. Composé de cinq couches convolutionnelles avec des activations ReLU, il intègre également des couches entièrement connectées pour la classification, ainsi qu'une branche de régression pour la localisation des objets.

ResnetObjectDetector :

ResnetObjectDetector est basé sur l'architecture ResNet18 pré-entraînée, largement utilisée dans le domaine de l'apprentissage profond. Il utilise les couches convolutionnelles de ResNet18, comprenant 18 couches au total, avec des activations ReLU. Il comporte également des couches entièrement connectées pour la classification des objets, ainsi qu'une branche de régression pour la localisation des objets, tirant parti de la sortie de la partie convolutionnelle de ResNet18

Entraînement du modèle :

L'entraînement du modèle se déroule sur 20 époques avec un batch size de 32. À chaque époque, les données sont divisées en mini-lots de taille 32 et le modèle est entraîné sur ces mini-lots successivement. Pour chaque mini-lot, une passe avant est effectuée pour obtenir les prédictions du modèle, suivie du calcul de la perte. La perte est calculée à partir des prédictions du modèle et des annotations de l'ensemble d'entraînement. Une fois la perte calculée, une rétropropagation est effectuée pour ajuster les poids du modèle en fonction de la perte, grâce à l'algorithme d'optimisation Adam.

Après chaque époque d'entraînement, la performance du modèle est évaluée sur l'ensemble de validation. Cela permet de surveiller la capacité du modèle à généraliser à des données qu'il n'a pas vues pendant l'entraînement. Les métriques évaluées incluent la perte, la précision de classification et l'indice (IoU) pour évaluer la qualité des prédictions de localisation. Le modèle est également sauvegardé à chaque amélioration de la performance sur l'ensemble de validation, garantissant que seul le meilleur modèle est conservé. Une fois les 20 époques terminées, le modèle final est sauvegardé et le temps total d'entraînement est enregistré pour évaluer l'efficacité du processus d'entraînement.

Evaluation : Une évaluation supplémentaire a été ajoutée pour le dernier modèle conservé à la fin de l'entraînement (le meilleur). Cette évaluation se concentre sur la performance au sein de chaque classe afin de vérifier que le modèle ne s'adapte pas à une seule classe dominante. Les résultats montrent que les modèles ont une meilleure performance pour la localisation des

motos que des voitures. Cependant, en ce qui concerne la classification, le modèle reconnaît parfaitement les deux classes.

```
*** train set accuracy
Mean accuracy for all labels: 1.0
Mean iou for all labels: 0.8183961682717917

Mean accuracy for label motorcycle: 1.0
| 639 over 639 samples

Mean IoU for label motorcycle: 0.8499840615091363

Mean accuracy for label car: 1.0
| 638 over 638 samples

Mean IoU for label car: 0.7867587642300002
```

FIGURE 4 – Évaluation sur le train set

```
*** validation set accuracy
Mean accuracy for all labels: 1.0
Mean iou for all labels: 0.7752301637803375

Mean accuracy for label motorcycle: 1.0
| 78 over 78 samples

Mean IoU for label motorcycle: 0.830900585636748

Mean accuracy for label car: 1.0
| 82 over 82 samples

Mean IoU for label car: 0.722275372258386
```

FIGURE 5 – Évaluation sur le validation set

```
*** test set accuracy
Mean accuracy for all labels: 0.99375
Mean iou for all labels: 0.7794418593091788

Mean accuracy for label motorcycle: 1.0
| 80 over 80 samples

Mean IoU for label motorcycle: 0.8218686831713191

Mean accuracy for label car: 0.9875
| 79 over 80 samples

Mean IoU for label car: 0.7370150354470384
```

FIGURE 6 – Évaluation sur le test set

5 Résultat

pour chaque modèle nous nous sommes appuyés sur trois indicateurs 0 : l' accuracy, la loss ainsi que l'IoU.

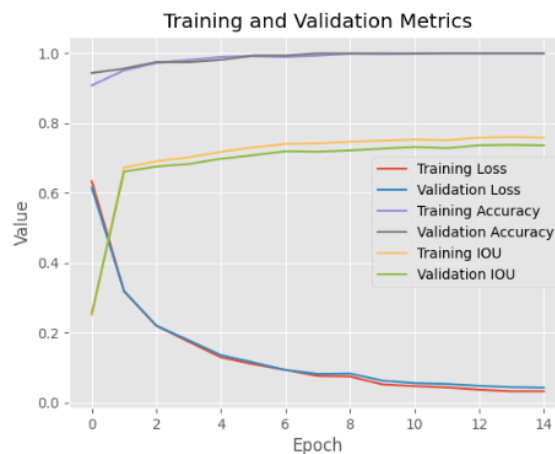


FIGURE 7 – Indicateurs en fonction du nombre d'époch pour le modèle simple

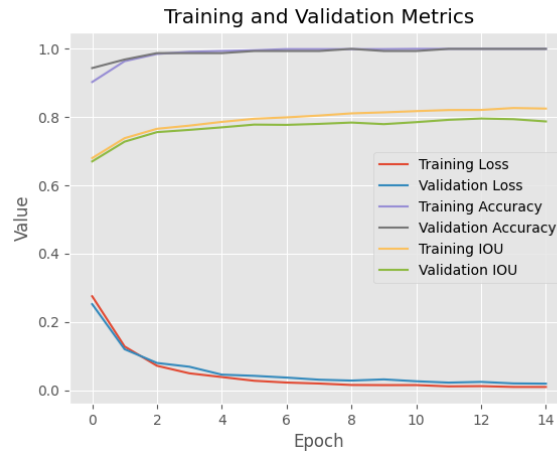


FIGURE 8 – Indicateurs en fonction du nombre d'époch pour le modèle profond

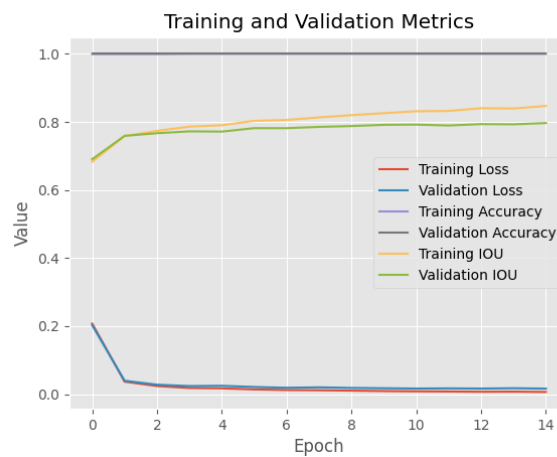


FIGURE 9 – Indicateurs en fonction du nombre d'époch pour Resnet

Avec ces courbes, nous voyons bien que l'accuracy de la validation est inférieure un peu de celle de l'entraînement, donc on est sûr qu'on n'est pas dans le cas d'overfitting. Finalement, les loss tendent vers 0 et les accuracy vers 1, ce qui est bien le comportement attendu.

Faisons ensuite un benchmark des modèles. Pour cela, nous utilisons le même ensemble de test sur chaque modèle :

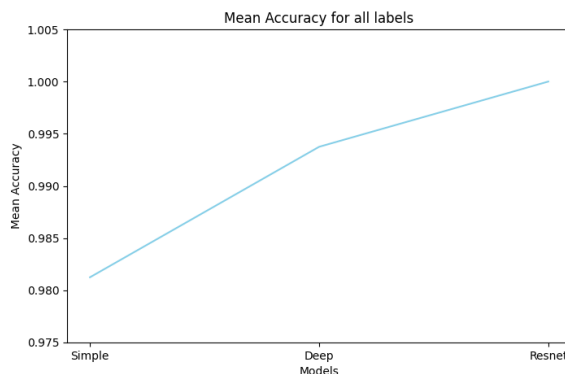


FIGURE 10 – Performance des modèles pour la classification

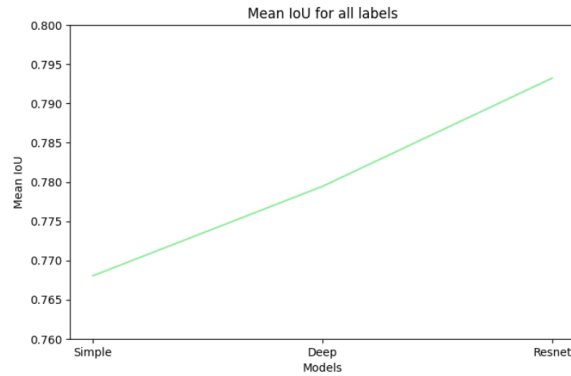


FIGURE 11 – Performance des modèles pour la localisation

Nous pouvons alors affirmer que le modèle avec Resnet est plus efficace que le modèle profond, lui-même plus efficace que le modèle simple.

6 Discussion

6.1 Synthèse et interprétation des résultats

Nos recherches et expérimentations ont conduit à l'élaboration de plusieurs modèles basés sur des CNN, dont le DeeperDetector et le ResnetObjectDetector. Ces modèles ont montré une capacité supérieure à localiser et classer des objets avec précision, réalisant un IoU de 0.77 et une précision de classification atteignant 1.0 pour le modèle basé sur ResNet.

Le DeeperDetector, spécifiquement, a prouvé être un outil puissant dans notre arsenal, avec sa configuration profonde permettant une extraction de caractéristiques plus détaillée, ce qui se traduit par des performances améliorées par rapport à des architectures moins profondes. Ce modèle a réussi à équilibrer efficacité et complexité, offrant une alternative viable à des solutions plus gourmandes en ressources.

6.2 Points à retenir et limites

L'étude met en lumière l'importance cruciale de l'initialisation appropriée des poids et de la normalisation par lots pour stabiliser et accélérer de la stabilité et de la vitesse de convergence des modèles. Toutefois, malgré les succès obtenus, certaines limites persistent. Par exemple, bien que les modèles performant bien sur les données de test, la généralisation à des scénarios réels non représentés dans l'ensemble de formation reste un défi. De plus, la dépendance à des architectures complexes comme ResNet peut introduire des contraintes de ressources lors de déploiements sur des systèmes avec des capacités de calcul limitées.

6.3 Pistes à explorer

Pour améliorer davantage la robustesse et l'efficacité de la localisation de motifs, plusieurs pistes pourraient être explorées :

- **Augmentation des données :** Enrichir l'ensemble de formation avec des images générées par des techniques d'augmentation pour améliorer la généralisation du modèle.
- **Optimisation des architectures :** Développer des architectures plus légères qui maintiennent une haute précision tout en étant plus adaptées à une exécution en temps réel

sur des dispositifs à faible consommation d'énergie.

En conclusion, nos travaux nous ont permis de découvrir la technologie de détections de motifs grâce à l'utilisation efficace des CNN, mais des efforts continus sont nécessaires pour surmonter les défis existants et maximiser l'applicabilité des modèles dans des applications pratiques variées, notamment des possibilités d'amélioration peuvent être la détection de plusieurs objets dans une même image ou encore de la détection en temps réels.

7 Conclusion

Dans cette étude, nous avons exploré diverses approches de détection d'objets dans des images en utilisant des architectures de réseaux de neurones convolutionnels (CNN). Notre objectif était de développer des modèles capables de localiser et classifier précisément des objets dans des scénarios variés, avec un accent particulier sur les classes de voitures et de motos.

Après avoir mis en œuvre plusieurs modèles, SimpleDetector, DeeperDetector, et ResnetObjectDetector, nous avons évalué leurs performances sur un ensemble de données dédié. Les résultats ont montré une amélioration significative par rapport aux méthodes initiales, avec une précision maximale de classification de 1.0 et un IoU de 0.77 pour le modèle ResnetObjectDetector.

Une analyse détaillée des résultats a révélé que les modèles étaient capables de localiser et classifier avec précision les objets dans des conditions variées, démontrant ainsi l'efficacité de l'approche basée sur les CNN. De plus, l'utilisation de techniques telles que l'initialisation des poids et la normalisation par lots a contribué à améliorer la stabilité et la vitesse de convergence des modèles.

En conclusion, cette étude a permis de mettre en évidence les avantages des approches basées sur les CNN pour la détection d'objets dans des images. Bien que des défis subsistent, nos résultats montrent un potentiel prometteur pour l'application de ces modèles dans une variété de domaines, de la surveillance à la conduite autonome, ouvrant ainsi la voie à des applications pratiques et innovantes dans le domaine de la vision par ordinateur.