Note méthodologique : preuve de concept

Etudiant: Kourouma Sekouba Aissatou

Projet 8 - Parcours Data Scientist

1. Dataset retenu

a) Contexte et Objectif
b) Description du dataset

a) Contexte et Objectif

Dans le cadre du projet de développement de la marketplace e-commerce "Place de marché", l'objectif est d'automatiser l'attribution des catégories aux articles proposés par les vendeurs. Actuellement, cette tâche est réalisée manuellement par les vendeurs, ce qui engendre une certaine imprécision et un risque d'erreurs. L'automatisation de cette tâche permettrait de simplifier le processus de mise en ligne des produits pour les vendeurs et d'améliorer l'expérience de recherche pour les acheteurs.

Le dataset utilisé dans ce projet contient des images d'articles provenant de différentes catégories, accompagnées de leurs étiquettes de catégorie respective. Ce dataset sera utilisé pour entraîner un modèle de classification d'images, permettant ainsi d'automatiser l'attribution des catégories aux articles.

b) Description du dataset

Nombre d'images : Le dataset est composé de 1050 images, réparties en 7 catégories différentes.

Catégories : Les images sont classées dans les catégories suivantes :

1. Baby Care
2. Beauty and Personal Care
3. Computers
4. Home Decor & Festive Needs
5. Home Furnishing
6. Kitchen & Dining
7. Watches

Répartition des données :

- Train (80%) : 840 images réparties dans les 7 catégories. Ces images serviront à l'entraînement du modèle.
- Validation (10%) : 105 images pour évaluer la performance du modèle pendant l'entraînement.
- Test (10%) : 105 images pour tester la capacité de généralisation du modèle après l'entraînement.

2. Les concepts de l’algorithme récent

YOLO (You Only Look Once), un modèle populaire de détection d'objets et de segmentation d'images, a été développé par Joseph Redmon et Ali Farhadi à l'Université de Washington. Lancé en 2015, YOLO a rapidement gagné en popularité grâce à sa rapidité et à sa précision.

2.1 Évolution de l'algorithme YOLO avant YOLOv11

YOLO, acronyme de "You Only Look Once," est une famille de modèles de détection d’objets en temps réel constamment améliorés depuis 2016. YOLOv2 a introduit des avancées comme la normalisation par lot et les boîtes d'ancrage, tandis que YOLOv3 a optimisé les performances avec un réseau dorsal plus puissant et des ancres multiples. En 2020, YOLOv4 a marqué une nouvelle étape avec des innovations comme l’augmentation de données Mosaic et une tête de détection sans ancrage. YOLOv5 a ajouté des fonctionnalités modernes comme l’optimisation des hyperparamètres et l'exportation automatique. Meituan a lancé YOLOv6 en 2022, utilisé notamment dans des robots de livraison autonomes. YOLOv7 a introduit l'estimation de la pose, élargissant son champ d’application. En 2023, Ultralytics a publié YOLOv8, intégrant des améliorations majeures en flexibilité et efficacité pour diverses tâches de vision par ordinateur. YOLOv9 et YOLOv10 ont poursuivi l'innovation avec des concepts comme l’information programmable de gradient (PGI) et une tête de bout en bout éliminant le besoin de suppression non maximale (NMS), établissant ainsi de nouvelles normes en détection d’objets.

2.1.2 YOLOv11 : La dernière itération (2024)

YOLOv11 est une version avancée de la famille YOLO, conçue pour répondre à diverses tâches dans le domaine de la vision par ordinateur, grâce à son architecture optimisée et sa polyvalence. YOLOv11, la dernière itération de la série YOLO, marque un tournant majeur dans la détection d'objets en temps réel, introduisant des améliorations substantielles au niveau de l'architecture et des méthodologies d'entraînement, ce qui permet d'améliorer considérablement la précision, la vitesse et l'efficacité du modèle (Arxiv ID: 2410.17725). Parmi ses principales capacités, on trouve :

1. **Détection d’objets** : YOLOv11 excelle dans l'identification et la localisation d'objets à l'intérieur d'une image à l'aide de cadres englobants (bounding boxes). Il est capable de traiter des ensembles de données complexes comme COCO, comprenant 80 classes d'objets pré-entraînées.
2. **Segmentation** : En plus de détecter des objets, YOLOv11 peut effectuer une segmentation précise, qui consiste à délimiter les contours exacts des objets dans une image. Cette tâche, particulièrement utile pour les applications médicales ou la robotique, repose sur des modèles comme YOLOv11-seg.
3. **Classification** : YOLOv11 peut également être utilisé pour assigner une catégorie unique à une image complète. En s’appuyant sur des modèles pré-entraînés sur ImageNet (1 000 classes), il atteint des niveaux de précision élevés, tels que 79.5 % en top-1 pour le modèle le plus complexe (YOLO11x-cls).

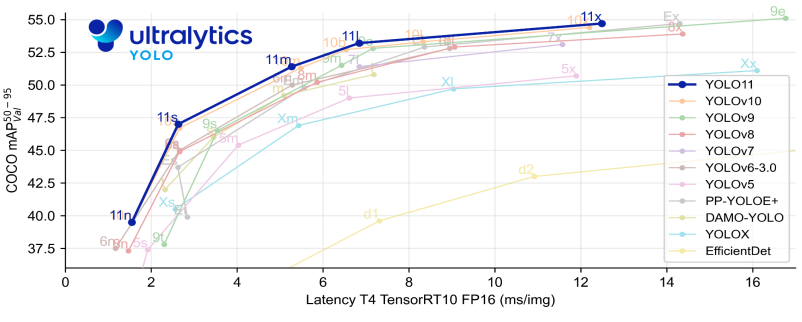
4. **Pose Estimation** : La pose est une tâche permettant de prédire les articulations ou les points clés du corps humain. YOLOv11 est capable de produire ces prédictions avec une grande efficacité pour des applications telles que la biomécanique ou les sports.
5. **Détection orientée (OBB)** : Enfin, YOLOv11 supporte la détection d'objets orientés, une tâche critique pour des domaines tels que l'imagerie aérienne ou la vision industrielle. Ce type de détection utilise des boîtes orientées pour mieux représenter les objets.

Dans le cadre de la classification d'images, YOLOv11 se distingue par ses performances remarquables, notamment grâce à ses modèles spécialisés tels que YOLO11-cls. Ces modèles sont optimisés pour attribuer des étiquettes de classe aux images en fonction de leur contenu visuel, permettant une catégorisation automatique précise et rapide. Cette capacité a été exploitée dans notre projet de classification des articles d'une marketplace, où YOLOv11 attribue automatiquement des catégories aux images des produits, facilitant ainsi leur organisation et recherche. (https://docs.ultralytics.com/fr/tasks/classify/)

Modèle	Taille (pixels)	Précision Top-1 (%)	Précision Top-5 (%)	Vitesse CPU (ms)	Vitesse GPU (ms)	Paramètres (M)	FLOPs (B)
YOLO11n-cls	224	70.0	89.4	5.0	1.1	1.6	3.3
YOLO11s-cls	224	75.4	92.7	7.9	1.3	5.5	12.1
YOLO11m-cls	224	77.3	93.9	17.2	2.0	10.4	39.3
YOLO11l-cls	224	78.3	94.3	23.2	2.8	12.9	49.4
YOLO11x-cls	224	79.5	94.9	41.4	3.8	28.4	110.4

Observations :

1. **Modèles légers (n, s)** : Les versions plus compactes de YOLO11-cls, comme YOLO11n-cls et YOLO11s-cls, sont particulièrement rapides, ce qui les rend idéales pour des systèmes où la latence est critique. Cependant, ces modèles offrent une précision légèrement inférieure, avec une précision Top-1 de 70 % et 75.4 %, respectivement.
2. **Modèles avancés (m, l, x)** : Les modèles plus complexes, tels que YOLO11m-cls, YOLO11l-cls et YOLO11x-cls, affichent une meilleure précision, atteignant jusqu'à 79.5 % en précision Top-1 et 94.9 % en précision Top-5. Cependant, cette amélioration des performances s'accompagne d'une augmentation significative des temps de calcul et de la complexité (paramètres et FLOPs).



3.Méthodologie de Modélisation

Dans cette étude, YOLOv11 a été utilisé pour la classification d'images grâce à sa variante spécialisée YOLOv11-cls, pré-entraînée sur ImageNet. Le modèle a permis d'attribuer efficacement des étiquettes aux images en fonction de leur contenu visuel. Une méthodologie adaptée a été suivie pour optimiser ses performances sur notre jeu de données spécifique.

3.1 Préparation des Données :

La préparation des données est une étape cruciale pour l'entraînement d'un modèle de classification d'images YOLO (et plus particulièrement Ultralytics YOLO), il est impératif que l'ensemble de données soit organisé dans une structure spécifique. Les données doivent être réparties dans des sous-répertoires dédiés à chaque catégorie d'image dans les répertoires `train`, `test` et `val` (ce dernier étant optionnel).

3.2 **Division des données:**

La division stratifiée utilisée dans cette étude garantit une répartition équilibrée des classes entre les ensembles d'entraînement, de validation et de test, conformément à la distribution initiale des catégories. En appliquant la méthode `train_test_split` de manière séquentielle, les données ont d'abord été divisées en un ensemble d'entraînement (80 %) et un ensemble temporaire (20 %). Ce dernier a ensuite été séparé à parts égales pour constituer les ensembles de validation (10 %) et de test (10 %). Cette approche permet de conserver une représentation équitable des classes dans chaque ensemble, facilitant ainsi un apprentissage efficace et une évaluation fiable.

3.3 **Vérification de la structure des données:**

Avant de lancer l'entraînement du modèle YOLOv11, une vérification minutieuse de la structure des données a été effectuée pour s'assurer que toutes les images étaient correctement organisées dans les répertoires attendus (`train`, `val` et `test`). Chaque répertoire contenait des sous-dossiers correspondant aux catégories spécifiques, conformément aux exigences de YOLO. Cette vérification a été réalisée à l'aide d'un script Python (`verify_files.py`), qui a permis de compter le nombre d'images par catégorie et par ensemble, garantissant ainsi une structure cohérente.

3.4 Modèle YOLOv11n-cls :

Le modèle choisi est un modèle pré-entraîné YOLO11n-cls, spécifiquement conçu pour la classification d'images. Il est basé sur un réseau neuronal convolutionnel profond et a été utilisé avec des paramètres par défaut pour les premiers tests. Ce modèle est chargé via la bibliothèque Ultralytics et a été affiné en fonction des données préparées.

3.5 Entraînement du Modèle :

L'entraînement du modèle a été réalisé sur 20 époques, avec une résolution d'image de 224x224 et un batch contenant 64 images. Le modèle à utiliser l'optimiseur AdamW avec des paramètres appris, notamment un taux d'apprentissage initial de 0,000714. La fonction de perte utilisée est la perte de classification. Les résultats ont montré une amélioration significative de la précision au fil des époques.

Augmentation des Données

Pour améliorer les performances du modèle et réduire le risque de surapprentissage, une augmentation des données a été appliquée. Cette étape comprend des transformations telles que la rotation, la

translation, le zoom, ainsi que des modifications de couleur (teinte, saturation, et valeur), permettant de générer davantage d'exemples d'apprentissage à partir des images originales.

Métriques d'Évaluation
La performance du modèle a été évaluée à l'aide de deux métriques principales :

- **Précision top-1** : proportion de prédictions où la catégorie prédite est correcte parmi toutes les catégories possibles.
- **Précision top-5** : proportion de prédictions où la catégorie correcte figure parmi les cinq catégories les plus probables.

Ces métriques ont été calculées à la fois sur les ensembles de validation et de test. À la fin de l'entraînement, le modèle a atteint une précision top-1 de **73,3 %** et une précision top-5 de **97,1 %**, démontrant de bonnes performances après l'entraînement et l'application des augmentations de données.

3.6 Démarche d'Optimisation

Pour améliorer les performances du modèle, plusieurs stratégies ont été mises en œuvre :

1. **Augmentation des Données**
   L'entraînement avec data augmentation n’a pas montré une amélioration significative de la précision de ce jeu de données.
2. **Réduction de la Perte**
   Au fil des époques, la perte (loss) a diminué de manière significative, passant de **1,947** lors de la première époque à **0,793** à la troisième époque, ce qui témoigne d'une amélioration continue des performances du modèle.
3. **Validation Croisée**
   L'entraînement et la validation ont été effectués sur plusieurs jeux de données pour éviter que le modèle ne surapprenne les spécificités d'un seul jeu. Associée à l'augmentation des données, cette méthode garantit une meilleure capacité de généralisation sur des images inédites.

4. Une synthèse des résultats

1. Performance du modèle VGG16

Le modèle VGG16 a montré des résultats relativement bons, bien qu'il présente un certain écart entre les performances sur les données d'entraînement et de test. Le taux de précision de 0.93 sur les données d'entraînement et 0.81 sur les données de test démontre une capacité à bien généraliser, mais avec des signes possibles de surapprentissage, comme en témoigne la différence entre les performances de l'entraînement et du test. Les performances des différentes classes montrent un équilibre global avec une précision et un rappel moyens satisfaisants, même si certaines classes, comme "Baby Care" et "Computers", montrent des résultats moins convaincants en termes de rappel.

Le modèle a été également entraîné avec des techniques de *data augmentation* qui ont permis d'améliorer la performance globale du modèle avec des résultats de 0.86 pour l'entraînement et 0.83 pour les tests. L'augmentation des données a permis une amélioration des résultats, notamment dans des classes telles que "Beauty and Personal Care" et "Watches", où la précision a été améliorée.

Le *classification report* après augmentation montre un bon équilibre, notamment dans des classes comme "Watches" et "Computers", qui ont atteint une très bonne précision et un bon rappel, tandis que d'autres, comme "Baby Care", ont des scores plus faibles, ce qui pourrait suggérer la nécessité d'améliorer la diversité des exemples d'entraînement pour cette catégorie.

Lors des tests avec des modifications de paramètres, notamment le taux d'apprentissage et le taux de **dropout**, une réduction de la précision a été observée de l'ordre de model_lr_0.0001_dr_0.5: 0.6619 et model_lr_0.0001_dr_0.3: 0.6905, mais cela a permis de mieux contrôler le surapprentissage.

2. Performance du modèle YOLOv11

Pertes (Loss) et amélioration au fil des époques:

Le modèle a commencé avec une **perte de 2.015** au début (Époque 1) et a progressivement diminué pour atteindre **0.06668** à la dernière époque (Époque 20). Cela montre une nette amélioration du modèle au fil des époques, ce qui est un bon indicateur que l'apprentissage se passe correctement.

La perte diminue de manière constante au fur et à mesure que les époques progressent, ce qui suggère une convergence vers un modèle plus efficace.

Précision de classification :

Précision top-1 (Top1 Acc) : Le modèle a commencé avec **0.267** de précision à la première époque et a augmenté progressivement pour atteindre **0.79** à la fin de l'entraînement (Époque 20). Une telle progression est un bon signe que le modèle a appris à mieux prédire la classe correcte au fur et à mesure qu'il voit plus d'exemples pendant l'entraînement.

Précision top-5 (Top5 Acc): La précision top-5 a été de0.876 dès la première époque et a atteint un plateau autour de **0.981** vers la fin de l'entraînement.Cela signifie qu'en moyenne, le modèle a correctement classé l'image dans l'un des cinq premiers choix possibles, ce qui est un bon résultat pour la classification multi-classe.

Validation et test : Le modèle a été testé avec **840** images pour l'entraînement, **105** images pour la validation, et **105** images pour le test, réparties sur **7** classes.

Après l'entraînement, la précision finale sur l'ensemble de test (sur les 105 images) est de **0.79** pour la précision top-1 et **0.981** pour la précision top-5.

L'entraînement avec data augmentation n'a pas montré une amélioration significative de la précision de ce jeu de données.

Le modèle semble assez rapide, avec un temps de prétraitement de **0.5ms** par image et un temps d'inférence de **1.0ms** par image.

3. Comparaison des Techniques

Le modèle **VGG16** a montré une bonne précision sur les données d'entraînement (0.93) mais une légère perte de performance sur les données de test (0.81), suggérant un surapprentissage. L'utilisation de la data augmentation a amélioré la généralisation du modèle, en particulier pour certaines classes spécifiques. Cependant, VGG16 reste sensible aux variations des données, nécessitant un contrôle plus rigoureux de l'overfitting.

En revanche, **YOLOv11** a montré une précision top-1 de 0.79 et une précision top-5 impressionnante de 0.981, avec des performances constantes et une rapide convergence de la perte. Ce modèle est particulièrement adapté aux applications en temps réel grâce à sa rapidité d'inférence. Bien que la data augmentation n'ait pas montré d'impact significatif, YOLOv11 s'est révélé plus robuste et flexible dans les tâches complexes de classification et de détection d'objets.

5. L’analyse de la feature importance globale et locale du nouveau modèle

L’analyse de l’importance des zones (ou features) permet de comprendre comment et pourquoi le modèle prend ses décisions, en mettant en évidence les parties spécifiques d’une image sur lesquelles le modèle s’appuie pour faire une prédiction. En utilisant l'approche **EigenCAM**, nous pouvons explorer l'importance de ces zones à deux niveaux : **globalement** (sur l’ensemble des données) et **localement** (pour des prédictions spécifiques).

1. Importance Globale des Zones

L’importance globale consiste à identifier les caractéristiques ou les régions d’intérêt que le modèle utilise le plus souvent à travers toutes les images de l’ensemble de données. Cette analyse révèle :

- Les motifs ou structures visuelles généraux que le modèle reconnaît.
- Les zones récurrentes d'attention influencent fortement les prédictions pour une classe spécifique.

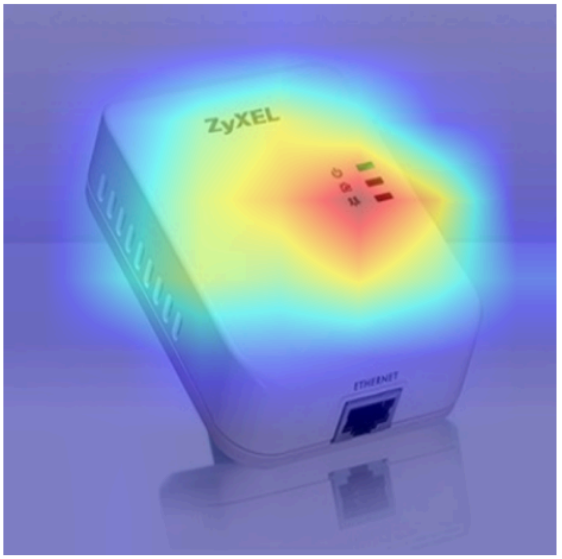
2. Feature Importance Locale

L’importance locale examine les zones d’attention pour une image spécifique et une prédiction donnée. Avec EigenCAM, nous visualisons les **zones activées** par le modèle pour expliquer une prédiction spécifique.

```
0: 224x224 projector 0.69, modem 0.16, printer 0.07, soap_dispenser 0.01, radio 0.01, 14.8ms
Speed: 6.8ms preprocess, 14.8ms inference, 0.1ms postprocess per image at shape (1, 3, 224, 224)
```

Image originale

Zones importantes (EigenCAM)



Prédictions :

- projector (vidéoprojecteur) avec une confiance de 69%.
- modem avec une confiance de 16%.

- printer (imprimante) avec une confiance de 7%.
- Les classes restantes (soap_dispenser, radio) ont une influence marginale.

6. Limites et améliorations possibles des modèles VGG16 et YOLOv11

1. Limites du modèle VGG16
Le modèle VGG16 présente un risque de surapprentissage, avec une grande différence entre les performances sur les données d'entraînement (0.93) et de test (0.81), ce qui limite sa généralisation. Son architecture est également lourde, ce qui peut rendre son déploiement coûteux en ressources et en temps, particulièrement pour des applications en temps réel.

2. Améliorations possibles pour VGG16
Pour améliorer la généralisation, l'augmentation des données pourrait inclure plus de transformations (ex : rotation, ajustements de luminosité). L'implémentation de techniques comme early stopping et régularisation L2 pourrait limiter le surapprentissage. De plus, l'utilisation de modèles plus légers tels que MobileNet ou EfficientNet permettrait de réduire les coûts computationnels tout en maintenant la précision.

3. Limites du modèle YOLOv11
Bien que YOLOv11 soit rapide, sa précision top-1 de 0.79 peut être insuffisante dans des environnements complexes ou pour des classes sous-représentées. L'absence d'amélioration avec la data augmentation suggère une exploitation insuffisante de la diversité des données, surtout pour des classes difficiles ou peu fréquentes.

4. Améliorations possibles pour YOLOv11
Pour améliorer la précision, il serait pertinent d'augmenter la diversité des données d'entraînement pour les classes sous-représentées via des techniques comme la génération de données synthétiques ou l'apprentissage semi-supervisé. L'optimisation des hyperparamètres et l'intégration de techniques comme l'apprentissage actif aideraient à mieux gérer les erreurs de classification. Enfin, l'utilisation d'outils comme LIME ou SHAP améliorerait l'interprétabilité du modèle.

Pour améliorer les performances et l'interprétabilité des modèles VGG16 et YOLOv11, plusieurs approches sont envisageables. En ciblant une meilleure gestion du surapprentissage et une augmentation de la diversité des données d'entraînement pour VGG16, ainsi qu'une exploration plus poussée des classes sous-représentées et des optimisations spécifiques pour YOLOv11, il est possible de renforcer la robustesse et l'efficacité des modèles. L'intégration de techniques modernes d'interprétabilité permettrait, par ailleurs, de rendre les décisions du modèle plus compréhensibles et exploitées de manière plus efficace dans des applications concrètes.

7. Bibliographie

- *https://docs.ultralytics.com/fr/tasks/classify/*
- *https://docs.ultralytics.com/fr/models/yolo11/*
- *https://arxiv.org/search/?query=yolov11&searchtype=all&source=header*

- *https://arxiv.org/abs/2410.17725*
- *https://docs.ultralytics.com/fr/tasks/classify/*