Classification d'images avec MobileViTv2

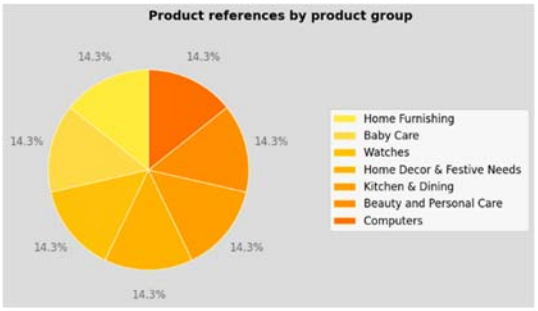
Note méthodologique & preuve de concept

Sommaire

Dataset retenu

Le jeu de données retenu est composé de 1,050 images non augmentées issues du Projet 6 – Classifiez Automatiquement des Biens de Consommation ; ce jeu d'images est celui sur lequel le meilleur résultat avait été obtenu dans le cadre de ce précédent projet pour la classification supervisée avec transfer learning sur les images seules (sans l'intégration des features issues de l'analyse du texte également disponible) avec le modèle ResNet50; ces résultats s'étaient cependant avérés tout à fait modestes et inhomogènes dans leur capacite de prédiction des classes. Ce jeu d'images offre donc une problématique intéressante, particulièrement en raison de sa taille très modeste[1], et il conviendra donc d'étudier si des perspectives d'amélioration significatives de la classification automatique de ces images existent avec l'emploi de techniques de modélisation plus récentes.

Ce jeu d'images présente une répartition équilibrée dans les 7 catégories suivantes :



Product Category	Label
Baby Care	0
Beauty and Personale Care	1
Computers	2
Home Décor & Festive Needs	3
Home Furnishing	4
Kitchen & Dining	5
Watches	6

Les concepts de l'algorithme récent

Rappels sur le benchmark : ResNet50

À sa publication en 2016 par plusieurs chercheurs de l'équipe de Microsoft Research Asia, l'architecture ResNet avait révolutionné l'optimisation de l'apprentissage et des performances des réseaux de neurones convolutifs très profonds (ci-après « CNN ») en apportant une solution innovante à la dégradation des performances causée par le problème du *vanishing gradient*. En introduisant les blocs résiduels et les connexions de saut en plus des couches convolutives classiques[2], ResNet a permis l'entraînement de modèles beaucoup plus profonds, dont le plus communément utilisé est la version à 50 couches, et a ouvert la voie à des architectures à 101, voire 152 couches, sans dégradation notable de la performance[3] sur des tâches complexes (classification multi-classes et segmentation). Cette architecture novatrice présentait également l'avantage de permettre une réduction computationnelle qui a permis son élargissement à d'autres jeux de données et tâches que ceux jusque-là traités par les architectures « classiques » (VGG notamment), comme par exemple la détection d'objets.

Chaque bloc de ResNet50 apprend non pas une fonction directe de transformation, comme c'est le cas d'une couche de convolution classique, mais le résidu entre l'entrée et la sortie visée. Ces sauts (« skip connections ») permettent au gradient (et donc à l'information d'apprentissage) de traverser directement plusieurs couches intermédiaires lors de la rétropropagation, grâce à un « pont » entre l'entrée et la sortie du bloc qui conserve l'information, évitant ainsi la disparition du gradient.

[1] Pour les besoins de l'apprentissage profond, tout jeu d'entraînement comportant moins de 1000 images est considéré petit.
[2] Composant fondamental des CNN, la couche de convolution permet d'extraire automatiquement les caractéristiques locales d'une image en appliquant des filtres (ou noyaux), qui sont des petites matrices de poids, en glissement sur l'image en entrée pour calculer des sommes pondérées et générer des cartes de caractéristiques ("feature maps") qui permettent de détecter des motifs comme des bords, des textures ou des formes ; la succession de ces couches dans un CNN permet de construire des représentations hiérarchiques de plus en plus abstraites, en utilisant des poids partages et une connectivité locale.
[3] ResNet et ses variantes ont remporté la 1re place dans plusieurs compétitions majeures en 2015 (ILSVRC, COCO), affichant par exemple un taux d'erreur Top-5 de 3,57 % sur ImageNet.

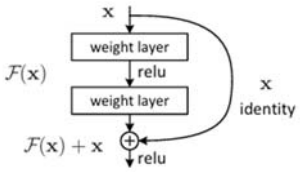


Figure 1 : Bloc résiduel avec connexion de saut

Il faut noter qu'en pratique, les blocs résiduels de type "bottleneck" sont utilisés dans les architectures ResNet plus profondes, comme ResNet-101 et ResNet-152, car ces blocs bottleneck sont moins coûteux en calcul. Un bloc résiduel "bottleneck" est une variante du bloc résiduel qui utilise des convolutions 1×1 pour créer un 'goulet d'étranglement' et permettre de réduire le nombre de paramètres et de multiplications de matrices.

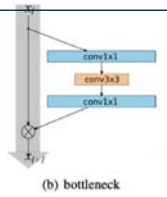


Figure 2 : Block résiduel inversé « bottleneck »

ResNet50 est un donc modèle puissant, mais qui nécessite cependant une utilisation fine de la régularisation et de l'augmentation de données pour éviter les problèmes de sur-apprentissage sur de petits jeux d'entraînement. La combinaison résidu/connexion de saut qu'il a introduite a marqué un tournant dans l'évolution du deep learning appliqué aux images, et elle est désormais standard pour la plupart des CNN ; elle a également inspiré certaines évolutions des transformers, structures issues des LLM[4] que les nouvelles générations de modèles comme MobileViT utilisent.

L'algorithme récent : MobileViTv2

Initialement développé par Apple en s'appuyant sur l'architecture MobileNet développée par Google et publié en 2022, le modèle MobileViTv2 est une architecture hybride optimisée pour les appareils mobiles et embarqués qui repose sur la combinaison de blocs transformeurs légers (les blocs « MobileViT ») et qui permet modéliser les relations globales à « longue distance » dans l'image tout en capturant également les caractéristiques locales avec des convolutions séparables en profondeur.

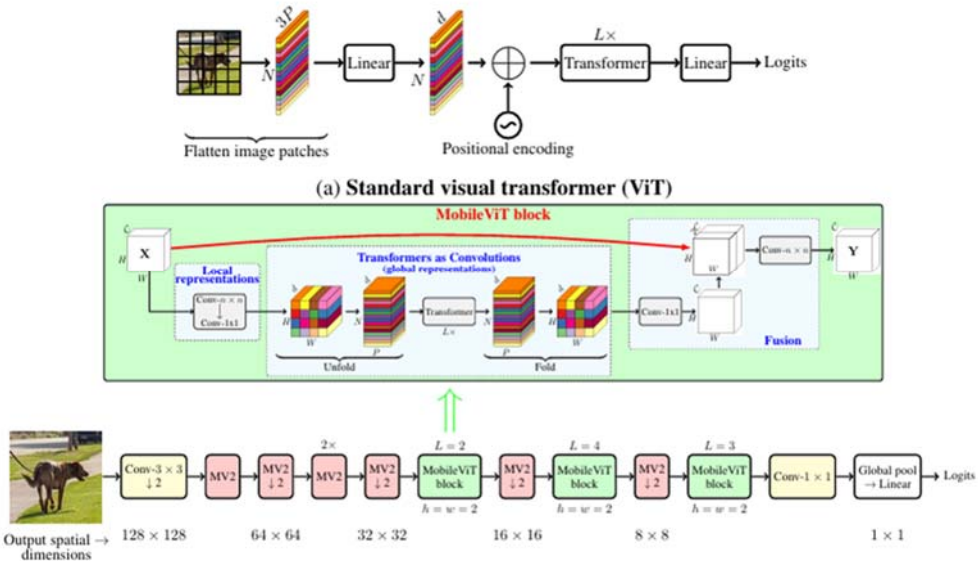


Figure 3 : Visual transformers et MobileViT blocs de « première génération »

[4] Large Language Models - réseaux de neurones spécialisés dans la compréhension et la génération de texte en langage naturel. Entraînés sur d'immenses corpus, ils prédisent et produisent des réponses cohérentes à partir d'instructions ou de contextes (dits « prompts »). Ces modèles sont à la base d'applications comme ChatGPT.

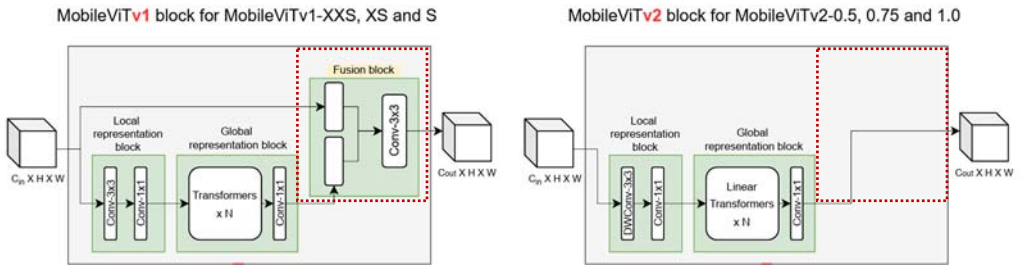


Figure 4 - Comparaison de l'architecture de Mobile ViTv1 et MobileViTv2 ; on note dans MobileViTv2 la disparition du fusion block, contribution majeure à l'efficience accrue du modèle.

Alors que dans ResNet50, l'utilisation des couches de convolution seules visait à extraire des motifs visuels pertinents hiérarchiquement à différents niveaux de profondeur[5], dans MobileViTv2, leur combinaison avec des blocs transformeurs permet d'y ajouter une modélisation globale de l'image en identifiant les régions les plus influentes pour la classification finale, même si celles-ci sont éloignées spatialement. Tout comme dans les LLM, l'utilisation des visual Transformers (ci-après « ViT ») dans le traitement d'images permet donc de capturer le contexte dans lequel sont inscrits les patches[6].

MobileViTv2 élabore et raffine également le concept des ViT, architectures traditionnellement coûteuses en calcul, en substituant une attention séparable (« separable self-attention ») à l'attention multi-tête (« multi-head self-attention ») utilisée jusque-là, ce qui permet, outre une meilleure propagation de l'information, une accélération remarquable des temps de traitement[7] qui rend cette famille de modèles particulièrement efficiente pour l'inférence en temps réel sur des appareils à ressources limitées (tels que les téléphones mobiles) ou sur des données bruitées (telles que des images satellite) ; cette accélération des temps d'inférence est atteinte avec des performances égales ou supérieures à celles des CNN purs comme ResNet50, et confère à ces modèles une excellente capacité de généralisation même sur des jeux d'entraînement de taille limitée. Enfin, dans l'architecture MobileViTv2 les activations non-linéaires après les couches fines ont également été supprimées au profit d'activations linéaires, contribuant encore davantage à l'efficience accrue de cette architecture par rapport à la « première génération » de ViTs[8].

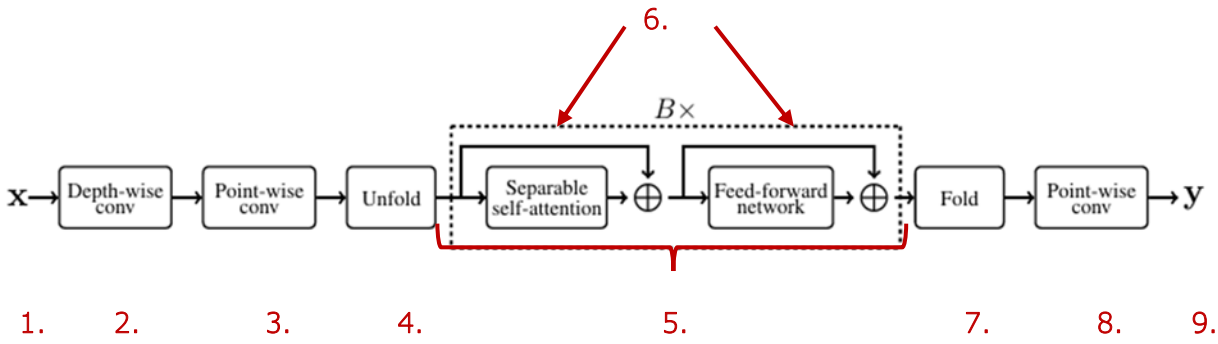


Figure 5 - Le bloc MobileViTv2

Le bloc MobileViTv2 affine le principe du bloc MobileViT en combinant modélisation locale (convolutions) et globale (attention séparable) selon les étapes suivantes :

1. Entrée X : L'image ou la carte de caractéristiques (features) d'entrée est transmise au bloc ;

[5] Les premières couches détectent des bords et des textures, tandis que les couches profondes capturent des objets ou des concepts plus abstraits.
[6] Unité de segmentation de l'image des visual transformers, équivalente au token dans les LLM.
[7] La complexité quadratique de l'attention multi-tête étant remplacée par une complexité linéaire.
[8] Une comparaison sur ImageNet fait apparaître que MobileViTv2 surpasse MobileViTv1 en précision tout en ne requérant que moitié moins d'opérations et un tiers de paramètres en moins. Voir [insert ref to Paper sandler 2019]

2. Convolution séparée en profondeur: utilise un noyau de taille 3×3 pour encoder les représentations locales et apprendre séparément des motifs dans chaque canal, ce qui extrait efficacement des caractéristiques locales avec peu de paramètres ;
3. Convolution ponctuelle: utilise un noyau de taille 1×1 et mélange l'information entre canaux, tout en maintenant l'efficacité paramétrique ;
4. Dépliage: Les cartes de caractéristiques sont découpées en k patchs plats (sous-blocs), rendant le flux compatible avec le traitement de type transformer en utilisant une hauteur et une largeur de patch de deux ;
5. Blocs transformeurs légers: Chaque patch passe à travers une auto-attention séparable (beaucoup plus légère que l'attention multi-tête classique, complexité linéaire O(k)), pour apprendre les dépendances globales entre patches, et un réseau feed-forward pour transformer davantage les représentations. Cette combinaison est répétée B fois[9] dans chaque bloc, selon la configuration du modèle ;
6. Sauts résiduels : Après chaque sous-bloc attention/feed-forward, un ajout résiduel ("skip connection") permet un apprentissage stable, même en profondeur ;
7. Repliage: Les patches transformés sont réassemblés ("repliés") pour reconstituer la carte de caractéristiques globale, préservant la structure spatiale. Tout comme le dépliage, le repliage utilise une hauteur et une largeur de patch de deux ;
8. Convolution ponctuelle : utilise un noyau de taille 1×1 et mélange à nouveau l'information entre canaux, ce qui prépare la carte pour les blocs suivants ou la sortie du réseau ;
9. Sortie y

Grâce à l'attention, les transformers peuvent détecter les caractéristiques sémantiquement pertinentes d'une image, agréger des informations locales et globales et s'adapter à de grandes images complexes sans perdre la capacité à se focaliser sur les détails clés. La self-attention « standard »[10] des transformers (dite aussi « multi-tête »), telle qu'utilisée par MobileViT est efficace mais coûteuse en calculs, particulièrement pour du matériel mobile. La self-attention séparée introduite par MobileViTv2 rend le calcul de l'attention beaucoup plus efficace.

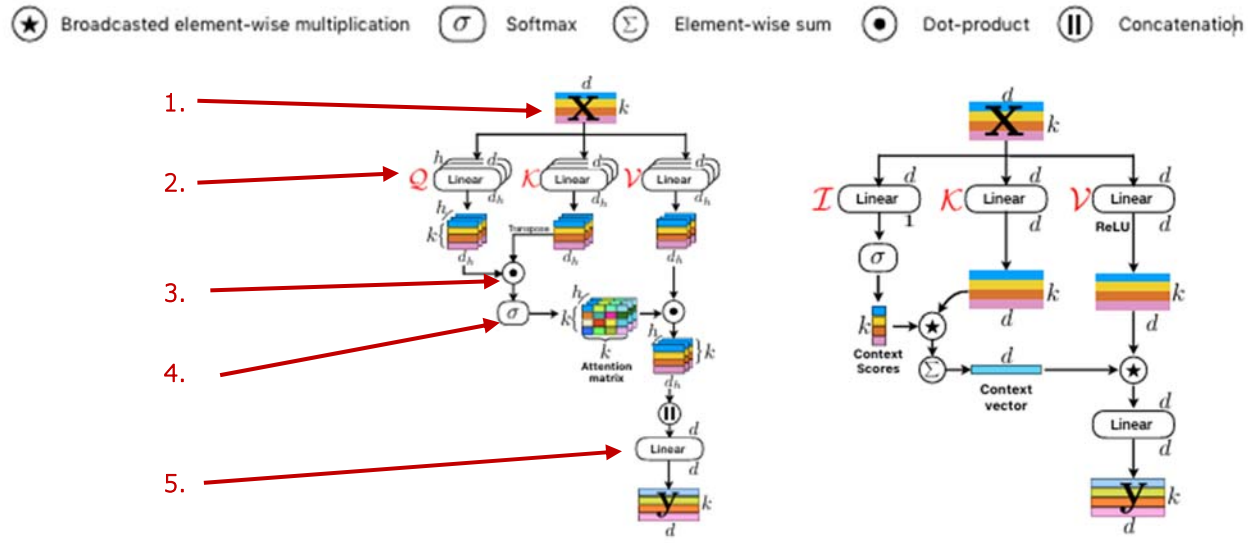


Figure 6 - Attention multi-tête – MobileViT et attention separable - MobileViTv2

Le schéma de gauche représente l'auto-attention multi-tête standard utilisée dans les transformers, et dont la complexité algorithmique est $O(k^2)$. Le schéma de droite correspond à la couche d'auto-attention séparable des transformeurs visuels de seconde génération, qui présente une complexité linéaire, c'est-à-dire O(k), et utilise des opérations élémentaires, notamment par la suppression des multiplications de matrices par lots, ce qui permet une inférence beaucoup plus rapide.
Ce mécanisme de l'attention s'articule autour des étapes suivantes :
1. Encodage de l'entrée : L'image est généralement découpée en patches et convertie en vecteurs ;
2. Vecteurs Query, Key, Value : Le vecteur de chaque patch est projeté sous trois formes séparées : Query (Q), Key (K) et Value (V) ;

[9] B est un hyperparamètre structurel qui vaut entre 2 et 4 selon la position du bloc dans le modèle.
[10] L'attention (« self-attention ») est un mécanisme central dans les modèles transformer, qui leur permet de se focaliser dynamiquement sur les parties les plus pertinentes de l'image d'entrée lors des prédictions, privilégiant certaines régions ou caractéristiques importantes d'une image, de façon similaire a une personne concentrant son regard sur des éléments distinctifs pour reconnaître une image.

3. Calcul de similarité : Le modèle calcule des scores d'attention en mesurant la correspondance entre la Query d'un patch et les Keys de tous les autres (produit scalaire). Cela détermine à quel point un patch doit "prêter attention" aux autres ;
4. Poids d'attention : Les scores sont normalisés (softmax), devenant des poids d'attention qui indiquent l'importance de chaque patch, avec :

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

où d_k est la dimension des vecteurs Key ;
5. Agrégation contextuelle : La sortie de chaque patch est une somme pondérée de tous les vecteurs Value, les poids reflétant les relations pertinentes.

Plutôt que de calculer une carte d'attention globale via des multiplications de matrices coûteuses, MobileViTv2 réalise donc des opérations élémentaires par patch permettant de garder le focus "global" ; cette architecture novatrice permet des performances élevées en classification d'image tout en limitant la complexité et le coût computationnel du modèle, et elle a été comparée de façon répétée dans la littérature scientifique à ResNet50, notamment du point de vue de sa précision en classification, qu'elle égale ou dépasse tout en requérant des ressources de calcul significativement moindres.

Synthèse de la comparaison :

Modèle	Précision (Imagnet)	Taille du modèle	Complexité de l'attention	Latence	Robustesse	Usage mémoire
ResNet50	Bonne à très bonne	Large	Inexistante (CNN pur)	Moyenne à haute	Sensible à l'over fit	Élevé
MobileViTv2	Très bonne	Compact	Séparable, linéaire O(k)	Faible	Excellente sur petit datasheet	Très faible

La littérature scientifique récente présentant MobileViTv2 comme une alternative moderne et efficiente à ResNet50, les performances des deux modèles ont donc été comparées sur le dataset retenu.

La modélisation

Afin d'assurer des résultats robustes, le jeu de données a été séparé en 3 sous-jeux d'entraînement (700 images), de validation et de test (175 images chacun).

Le contrôle de l'overfitting est assuré par un entraînement sur 50 epochs avec un critère d'early stopping associé à une patience de 5 epochs.

Afin d'assurer la comparabilité des résultats avec ceux obtenus avec ResNet50, aucune transformation (hors la normalisation et le resizing) ni augmentation n'ont été appliquées sur les images, le meilleur résultat ayant été obtenu avec ResNet50 sur les images originales non-augmentées.

La métrique principale retenue pour comparer les modèles est le score F1 macro, qui fournit une métrique synthétique qui équilibre le compromis entre la précision et le rappel (étant leur moyenne harmonique) ce qui en fait un choix courant pour l'évaluation des performances de modèles de classification multi-classes.

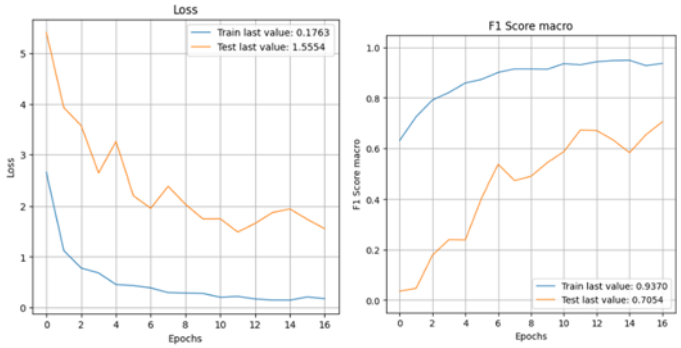
Le temps d'entraînement a également été considéré dans l'évaluation, ainsi que la capacité du modèle à prédire spécifiquement chaque classe, qui a été représentée dans une matrice de confusion.

La fonction de perte utilisée est la 'categorical_crossentropy', adaptée aux problèmes de classification multi-classes, et qui mesure la différence entre la distribution de probabilités prédite par le modèle et

la distribution connue ; plus la valeur est faible, plus la prédiction réalisée par le modèle est proche des « vrais labels » des classes.

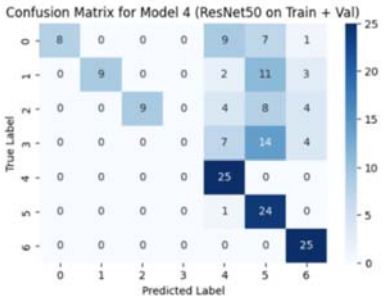
Synthèse des résultats

Rappels des résultats obtenus avec ResNet50



Fit time : 101.34 secondes

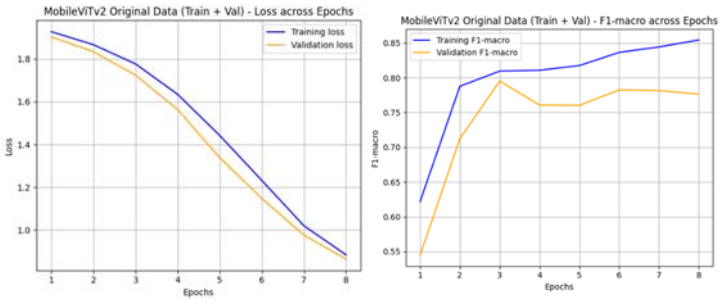
L'analyse de la fonction de perte montre qu'un plateau est atteint relativement rapidement sur le jeu d'entraînement, indiquant que le modèle « cesse d'apprendre ». Même si la fonction de perte sur le jeu de validation montre une tendance généralement décroissante, ses performances sont inégales selon les epochs, et l'écart entre les courbes d'entraînement et de validation ne se réduit pas de façon significative, restant relativement important. La capacité de généralisation du modèle s'en trouve affectée ; il est probable ici que la petite taille du jeu d'entraînement contribue au surapprentissage observé. Le score F1 montre quant à lui une amélioration irrégulière, et un écart relativement marqué entre les scores d'entraînement et de validation.



La matrice de confusion sur le jeu de test met en évidence des performances très inégales du modèle selon les classes :

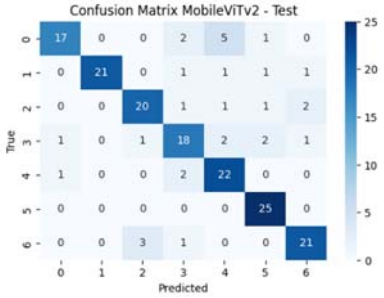
- Les classes 4 (Home Furnishing) et 6 (Watches) sont parfaitement prédites ;
- La classe 5 (Kitchen & Dining) est également très bien prédite ;
- Les classes 0 (Baby Care), 1 (Beauty & Personal Care) et 2 sont très mal prédites (environ 30% de prédictions correctes), étant généralement confondues avec les classes 4 à 6 …
- …de même que la classe 3 (Home Decor & Festive Needs), qui elle n'est en outre jamais correctement prédite.

Résultats obtenus avec MobileViTv2



Fit time : 23.13 secondes

Les courbes d'entraînement et de validation ne montrent pas de surapprentissage marqué pour MobileViTv2 ; les deux courbes diminuent progressivement avec le nombre d'epochs et la courbe de validation est relativement proche de celle d'entraînement, suggérant une bonne capacite de généralisation des résultats du modèle.



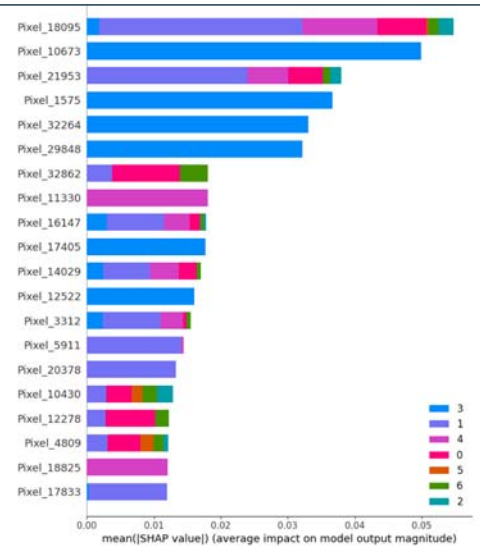
La matrice de confusion sur le jeu de test confirme ce résultat et met en évidence des prédictions d'une qualité beaucoup plus homogène qu'avec ResNet50, toutes les classes étant prédites correctement au moins 2 fois sur 3. La classe 0 (Baby Care) est la moins bien prédite ici ; la mieux (et parfaitement) prédite est la classe 5 (Kitchen & Dining).

MobileViTv2 a donc surpassé ResNet50 en termes de F1-score sur les ensembles de validation et de test, tout en nécessitant un temps d'entraînement divisé par 4. Ces observations empiriques sont cohérentes avec la littérature scientifique, MobileViTv2 étant conçu pour offrir un compromis optimal entre performance et efficacité sur les jeux d'entraînement de taille restreinte en environnement de ressources contraintes.

Feature importance

Feature importance globale

L'étude de la feature importance globale sur les modèles de classification d'images avec des packages comme SHAP offre peu de données interprétables, étant attribuée au niveau de l'unité de représentation qu'est le pixel du patch d'entrée.



L'exemple ci-contre pris sur les 20 premier pixels montre par exemple que les pixels en seconde et quatrième a sixième positions sont clairement attribués à la classe 3, alors que le pixel en avant dernière position est attribué à la classe 4.

C'est donc un outil peu informatif ici en l'absence de mapping des pixels aux patches et aux images auxquelles ils appartiennent.

Feature importance locale

L'étude d'un exemple de feature importance locale fournit des informations plus facilement exploitables pour la compréhension des résultats. En reprenant l'exemple de l'image f4d4c2eec77732f56e47722d7a355f2b.jpg qui appartient à la classe 3 (Home Decor & Festive Needs) et qui a été correctement prédite par le modèle, il est possible de visualiser les zones critiques pour la prédiction :

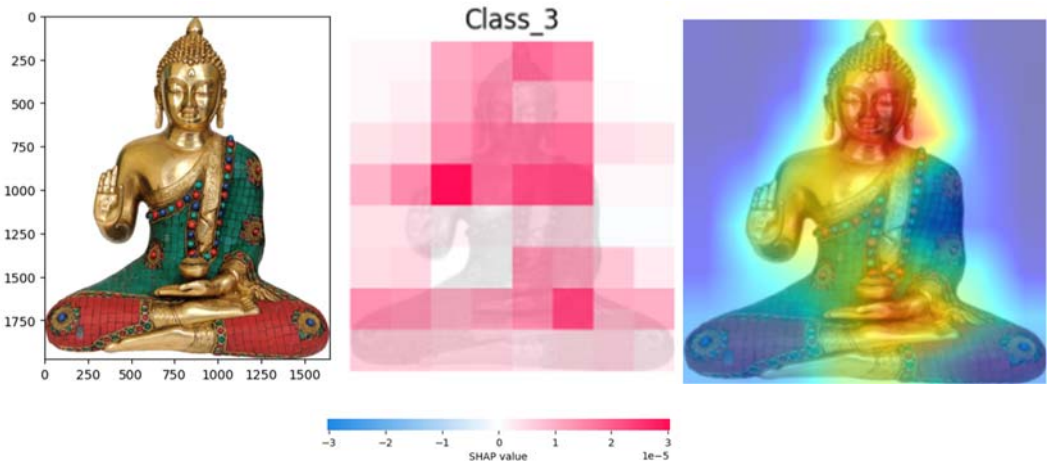


Image originelle | Feature importance locale avec SHAP | Feature importance locale avec GradCAM

Cet exemple met clairement en évidence les zones de l'image les plus déterminantes pour sa classification telles qu'activées par le modèle :

- Avec SHAP (image centrale), chaque patch est coloré en fonction de son importance dans la prédiction ;
- Avec GradCAM (image de droite), les gradients sont utilisés pour attribuer un poids à chaque pixel, et la heatmap offre des zones continues plus détaillées.

L'utilisation de GradCAM offre ainsi une granularité supérieure à SHAP, qui améliore l'auditabilité et la transparence des résultats du modèle et facilite leur interprétation par le métier.

Limites & améliorations possibles

- Augmentation des données : il serait nécessaire d'enrichir les jeux de données par des techniques avancées d'augmentation pour améliorer la robustesse du modèle peut offrir des perspectives d'amélioration supplémentaires de la qualité des prédictions ;
- Augmentation de la taille du dataset: Le défi principal ici est la taille très modeste du jeu d'entraînement. Un enrichissement du dataset par l'intégration d'articles et d'images supplémentaires dans chaque catégorie s'impose, particulièrement sur les catégories 0 à 3 qui étaient le moins bien identifiées par le modèle benchmark ;
- Inclusion d'un critère métier: à l'heure actuelle, les images sont classifiées manuellement par les équipes de Place de Marché et le pourcentage d'erreur humaine dans ce processus est inconnu. Avant toute décision d'implémentation de ces modèles en production, qui sera par nature chronophage et coûteuse, il est impératif d'intégrer cette donnée métier pour déterminer le coût d'opportunité du statu quo, afin d'établir si l'utilisation d'une classification automatique permet une réduction effective du taux d'erreur et du coût de main-d'œuvre nécessaire, entre autres facteurs à intégrer.

Références bibliographiques

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). *An image is worth 16x16 words: Transformers for image recognition at scale* (arXiv:2010.11929). Proceedings of the International Conference on Learning Representations (ICLR). https://arxiv.org/abs/2010.11929

He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep residual learning for image recognition* (arXiv:1512.03385). Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778. https://doi.org/10.48550/arXiv.1512.03385

Hugging Face. (n.d.). *MobileViTV2*. Hugging Face Transformers documentation. https://huggingface.co/docs/transformers/en/model_doc/mobilevitv2

Le, T.-D., Ha, V. N., Nguyen, T. T., Eappen, G., Thiruvasagam, P., Garces-Socarras, L. M., Chou, H.-F., Gonzalez-Rios, J. L., Merlano-Duncan, J. C., & Chatzinotas, S. (2024). *On-board satellite image classification for Earth observation: A comparative study of pre-trained Vision-Transformer models*. arXiv. https://arxiv.org/abs/2409.03901

Mehta, S., & Rastegari, M. (2021). *MobileViT: Light-weight, general-purpose, and mobile-friendly vision transformer* (arXiv:2110.02178). arXiv. https://arxiv.org/abs/2110.02178

Mehta, S., & Rastegari, M. (2022). *Separable self-attention for mobile vision transformers* (arXiv:2206.02680). arXiv. https://arxiv.org/abs/2206.02680

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2019). *MobileNetV2: Inverted residuals and linear bottlenecks* (arXiv:1801.04381v4). arXiv. https://arxiv.org/abs/1801.04381

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (pp. 618–626). https://doi.org/10.1109/ICCV.2017.74

Wadekar, S. N., Kan, K., Patel, K. K., Kudugunta, S., & Mehta, S. (2022). *MobileViTv3: Mobile-friendly vision transformer with simple and effective fusion of local and global features* (arXiv:2209.15159). arXiv. https://doi.org/10.48550/arXiv.2209.15159