

**Classification d’images avec MobileViTv2**

**Note méthodologique & preuve de concept**

Sommaire

[Dataset retenu 2](#_Toc203579287)

[Les concepts de l’algorithme récent 2](#_Toc203579288)

[Rappel sur le benchmark : ResNet50 2](#_Toc203579289)

[L’algorithme récent : MobileViTv2 4](#_Toc203579290)

[La modélisation 6](#_Toc203579291)

[Synthèse des résultats 7](#_Toc203579292)

[Rappels des résultats obtenus avec ResNet50 7](#_Toc203579293)

[Résultats obtenus avec MobileViTv2 9](#_Toc203579294)

[Feature importance 10](#_Toc203579295)

[Feature importance globale 10](#_Toc203579296)

[Feature importance locale 10](#_Toc203579297)

[Limites & améliorations possibles 11](#_Toc203579298)

[Références bibliographiques 12](#_Toc203579299)

# 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 et réparties en 3 sous-jeux d’entraînement (700 images), de validation et de test (175 images chacun).

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 de texte également disponibles) avec le modèle ResNet50; ces résultats s’étaient cependant avérés tout à fait modestes et inhomogènes dans leur capacite de prédiction des classes (voir rappel des résultats du modèle ResNet50 ci-dessous). Ce jeu de données offre donc une problématique intéressante, particulièrement en raison de sa taille très modeste (<1000 images), 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.

Le jeu d’images retenu 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 |

A pie chart with numbers and a graph

AI-generated content may be incorrect.

# Les concepts de l’algorithme récent

## Rappel sur le benchmark : ResNet50

À sa publication en 2016 par plusieurs chercheurs de Microsoft Research Asia, l’architecture ResNet a révolutionné l’optimisation de l’apprentissage et des performances des réseaux de neurones convolutifs (CNN) très profonds 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[[1]](#footnote-1) classiques, ResNet a permis l’entraînement de modèles beaucoup plus profonds, dont le plus communément utilisé est la version à 50 couches, qui a ouvert la voie à des architectures à 101 voire 152 couches sans dégradation notable de la performance[[2]](#footnote-2) sur des tâches complexes (classification, détection, segmentation) , tout en favorisant une réduction computationnelle et en améliorant la généralisation du réseau à d’autres jeux de données et tâches par rapport aux architectures « classiques » de l’époque comme VGG.

A diagram of a diagram

AI-generated content may be incorrect.

**Bloc résiduel avec connexion de saut**

**Source : https://paperswithcode.com/method/residual-block[[3]](#footnote-3)**

Chaque bloc 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.

A diagram of a machine

AI-generated content may be incorrect.

**Block résiduel inversé « bottleneck »**

**https://paperswithcode.com/method/bottleneck-residual-block**

Malgré sa puissance, ResNet50 nécessite une utilisation fine de la régularisation et de l’augmentation de données pour éviter les problèmes de sur-apprentissage sur de petits jeux de données[[4]](#footnote-4).

La combinaison résidu/connexion de saut introduite par ResNet a marqué un tournant dans l’évolution du deep learning appliqué aux images, et elle est désormais standard pour la plupart des architectures CNN ; elle a également inspiré les transformers, que les nouvelles générations de modèles comme MobileViT utilisent.

## L’algorithme récent : MobileViTv2

Initialement développé par Apple en s’appuyant sur MobileNet de 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), qui permettent de capter les caractéristiques et de modéliser les relations globales/ a longue distance dans l’image, avec des convolutions séparables en profondeur et des qui permettent de capturer également les caractéristiques locales de l’image ;

résiduels inverses

la ou les blocs résiduels dans ResNet50 reliaient des couches larges (dites « expanded »), MobileViTv2 relie des couches fines en entrée et en sortie du bloc, ce qui réduit la taille des données transitant par le réseau tout en conservant l’essentiel de l’information.

Alors que dans ResNet50, l’utilisation des couches de convolution visait à extraire des motifs visuels pertinents hiérarchiquement, à différents niveaux de profondeur (les premières couches détectent des bords et des textures, tandis que les couches profondes capturent des objets ou des concepts plus abstraits), dans MobileViTv2, leur combinaison avec des blocs transformeurs  permet, outre l’extraction des motifs locaux (petites régions de l’image) via le self-attention, d’identifier quelles régions de l’image sont les plus influentes pour la prédiction finale, même si elles sont éloignées spatialement, permettant une modélisation globale de l’image.

Les activations non-linéaires après les couches fines sont également supprimées au profit d’activations linéaires, contribuant encore davantage à l’efficience accrue de cette architecture par rapport à la « première génération » de visual Transformers[[5]](#footnote-5) (ViT). 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 dans les ViT, ce qui permet, outre une meilleure propagation de l’information, une accélération remarquable des temps de traitement[[6]](#footnote-6) 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[[7]](#footnote-7)) ; 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.

A diagram of a block diagram

AI-generated content may be incorrect.

**Visual transformers et MobileViT blocks**

**Source :** [**https://arxiv.org/pdf/2110.02178**](https://arxiv.org/pdf/2110.02178)

A diagram of a computer program

AI-generated content may be incorrect.

**Comparaison de l’architecture des blocs dans Mobile ViTv1 et MobileViTv2**

**From** [**https://arxiv.org/pdf/2209.15159**](https://arxiv.org/pdf/2209.15159)

A diagram of a flowchart

AI-generated content may be incorrect.

**\* La convolution en profondeur utilise un noyau de taille 3×3 pour encoder les représentations locales. Les opérations de « dépliage » (unfolding) et de « repliage » (folding) utilisent une hauteur et une largeur de patch de deux respectivement. Les couches d’auto-attention séparable et de feed-forward du transformeur sont répétées B fois avant d’appliquer l’opération de repliage (folding). Source : https://arxiv.org/pdf/2206.02680**

**MobileViTv2 block\***

Le **bloc MobileViT** combine le traitement local des CNN et le traitement global des transformers de la manière suivante :

1. **Entrée** : Une image (par exemple RGB = 3 canaux) passe au bloc.
2. **Convolution n×n** : Extraction initiale de motifs locaux.
3. **Convolution 1×1** : Combinaison linéaire des canaux.
4. **Unfolding** : Découpage de chaque canal en patchs aplatis.
5. **Transformer** : Passage des patches au travers d’une couche transformer pour extraction des dépendances globales.
6. **Folding** : Les patches sont “repliés” pour former une nouvelle image de dimension d.
7. **Pointwise convolution** : Application d’une convolution ponctuelle (1×1).
8. **Fusion** : Réintégration avec les canaux originaux.

**Résultat :** Le bloc peut modéliser à la fois des dépendances locales et globales tout en maintenant un nombre de paramètres très réduit.



A diagram of a graph

AI-generated content may be incorrect. A diagram of a graph

AI-generated content may be incorrect.

attention multi-tete attention separable

complexite O(k²)[[8]](#footnote-8) Complexité linéaire (O(k) au lieu de O(k²))

Suppression des multiplications de matrices par lots

Latence encore réduite, même sur les appareils mobiles

Amélioration à la fois de la précision et de la rapidité d’inférence

**Figure 3 : Différentes variantes d’unités d’auto-attention.**

(a) représente une auto-attention multi-tête standard (MHA) utilisée dans les transformers.  
(b) étend la MHA de (a) en introduisant des couches de projection de tokens, qui projettent k tokens vers un nombre prédéfini de tokens p, réduisant ainsi la complexité de O(k²) à O(k). Cependant, cette variante utilise toujours des opérations coûteuses (par exemple, des multiplications de matrices en batch) pour calculer l’auto-attention, ce qui affecte la latence sur les appareils à ressources limitées (voir Fig. 1).  
(c) correspond à la couche d’auto-attention séparable proposée : elle présente une complexité linéaire, c’est-à-dire O(k), et utilise des opérations élémentaires, ce qui permet une inférence beaucoup plus rapide. **Source : https://arxiv.org/pdf/2206.02680**

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[[9]](#footnote-9), 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 F1-score (score F1 macro), qui combine précision et rappel en une seule valeur synthétique. Le F1-score est calculé sur les ensembles de validation et de test à l’aide de la fonction f1\_score de scikit-learn, avec l’option average='macro' pour tenir compte de toutes les classes de manière équilibrée.

Le score F1 est la moyenne harmonique de la précision et du rappel :

A close-up of words

AI-generated content may be incorrect.

où :

A math equations with black text

AI-generated content may be incorrect.A close up of words

AI-generated content may be incorrect.

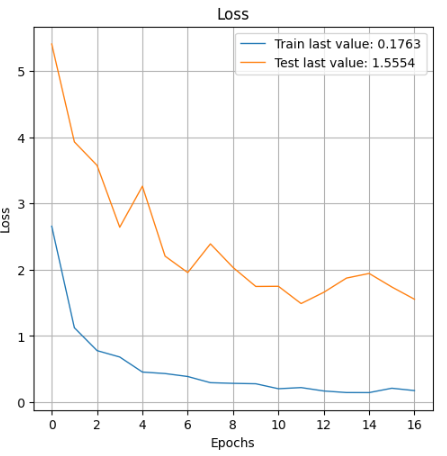
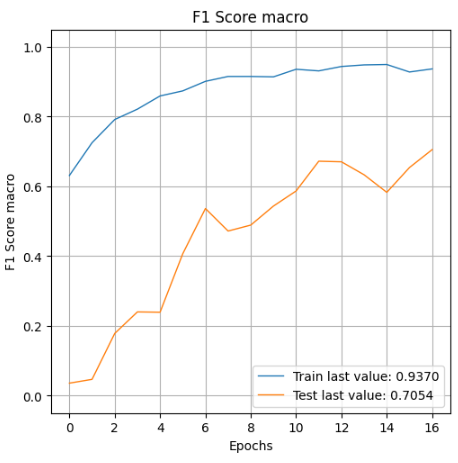
Le score F1 varie de 0 (le pire) à 1 (le meilleur) et un score élevé indique que le modèle a à la fois une haute précision et un haut rappel ; le score F1 fournit donc une métrique synthétique qui équilibre le compromis entre la précision et le rappel, ce qui en fait un choix courant pour l’évaluation des performances d’un modèle de classification multi-classes.

Le fit time a également été considéré dans l’évaluation, ainsi que la capacite 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.

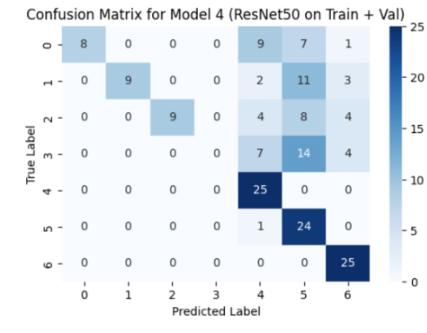
# Synthèse des résultats

## 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

A graph of a graph

AI-generated content may be incorrect.A graph with blue and orange lines

AI-generated content may be incorrect.

**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.

A graph of a graph

AI-generated content may be incorrect.

La matrice de confusion sur le jeu de test met en évidence des prédictions d’une qualité beaucoup plus homogène, 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 sont cohérentes avec la littérature scientifique, MobileViTv2 offrant un compromis optimal entre performance et efficacité sur les jeux d’entraînement de taille restreinte.

# Feature importance

## Feature importance globale

L’étude de la feature importance globale sur les modèles de classification d’images offre peu de données interprétables, étant attribuée au niveau de l’unité de représentation qu’est le pixel.

A graph of different colored squares

AI-generated content may be incorrect.

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 attribues 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 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 :

A gold statue of a buddha sitting in a lotus pose

AI-generated content may be incorrect.A screenshot of a cell phone

AI-generated content may be incorrect.A colorful statue of a buddha

AI-generated content may be incorrect.

A close up of a number

AI-generated content may be incorrect.

**Image originelle** **Feature importance locale Feature importance locale**

**avec SHAP 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.

L’utilisation de GradCAM[[10]](#footnote-10) offre ici une granularité supérieure à SHAP, qui ameliore l’auditabilite et la transparence des resultats du modele.

# Limites & améliorations possibles

* **Augmentation des données :** 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 (<1000 images). 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 les CNN « purs » comme ResNet50 ;
* **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 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 de classification et du coût de main-d’œuvre nécessaire, parmi d’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>

1. 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. [↑](#footnote-ref-1)
2. Le ResNet50 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. [↑](#footnote-ref-2)
3. À noter qu’en pratique, les blocs résiduels de type “bottleneck” sont utilisés dans les architectures ResNet plus profondes, comme ResNet-50 et ResNet-101, car ces blocs bottleneck sont moins coûteux en calcul. Un bloc résiduel “bottleneck” est une variante du bloc résiduel classique qui utilise des convolutions 1×1 pour créer un goulot d’étranglement. L’usage de ce goulot d’étranglement permet de réduire le nombre de paramètres et de multiplications de matrices. L’idée est de rendre les blocs résiduels aussi “fins” que possible afin d’augmenter la profondeur du réseau tout en limitant le nombre de paramètres.  
   Ces blocs ont été introduits dans l’architecture ResNet et sont employés notamment dans les ResNet profonds comme ResNet-50 et ResNet-101. Source : https://paperswithcode.com/method/bottleneck-residual-block [↑](#footnote-ref-3)
4. Pour les besoins de l’apprentissage profond et des CNN, tout jeu d’apprentissage comportant moins de 1000 images est considéré comme un petit jeu. [↑](#footnote-ref-4)
5. Une comparaison sur ImageNet fait apparaître que MobileViTv2 surpasse MobileViTv1 en précision tout en ne requérant que moities moins d’opérations et un tiers de paramètres en moins. Voir [insert ref to Paper sandler 2019] [↑](#footnote-ref-5)
6. La complexité quadratique de l’attention multi-tête étant remplacée par une complexité linéaire. Add link to paper Mehta Rastegari 2022. [↑](#footnote-ref-6)
7. Ref to paper 2024 [↑](#footnote-ref-7)
8. où k est le nombre de patchs dans l’image [↑](#footnote-ref-8)
9. Cite paper 2019 sandler here. [↑](#footnote-ref-9)
10. Add link to selvaraju 2017 paper here [↑](#footnote-ref-10)