

HEIG-VD — ARN

Laboratoire 5 – Rapport

[redacted]

30 octobre 2023

1 Introduction

Dans le cadre de ce laboratoire, nous allons passer par toutes les étapes requises pour créer une application de classification d'objets. Dans cet objectif, nous nous sommes demandés ce qui pourrait être utile de classifier et nous nous sommes décidés sur quelques types de véhicules, à savoir les hatchback, les pickups, les suv, et les vans. En plus de nous permettre une première expérience du transfer learning, une telle application pourrait permettre la mise en place de caméras à des fins statistiques sur les types de véhicules circulant dans une région donnée, ou bien permettre à des amateurs de découvrir un peu plus le monde de l'automobile et les différents types de véhicules existants.

En nous basant sur un dataset pré-existant, retravaillé pour correspondre à nos besoins, et des photos prises par nos soins, nous allons appliquer le transfer learning à l'aide de `MobileNetV2` sur les poids de *ImageNet* dont les couches sont figées. Nous allons ensuite procéder au peaufinage du modèle afin d'obtenir les meilleurs résultats possibles avec notre dataset.

2 Problématique

Nous étions initialement partis dans l'idée de différencier les marques principales de véhicules en circulation, avant de nous rendre compte de la complexité de la collecte de telles données. En effet, outre le grand nombre de marques, plusieurs d'entre elles utilisent les mêmes châssis ou offrent une large gamme de types de véhicules, rendant une telle classification très difficile pour une première expérience du transfer learning.

Nous avons donc rabattu nos choix de classes sur les différents types de véhicules et plus précisément les hatchback, les pickup, les SUV, et les van. Comme toute classe de véhicules, elles possèdent des caractéristiques reconnaissables, suivant l'angle de prise de vue, tout en offrant des similitudes entre-elles et spécialement entre les SUV et les hatchback. Une version de profil des types de voitures sélectionnés se trouve dans la [figure 1](#).

Après plusieurs recherches sur internet, nous nous sommes rendus compte que des datasets existaient déjà pour la classification de véhicules et nous avons trouvé l'un d'eux sur Kaggle, offrant une séparation des images par type de véhicules. Ce dataset, basé sur le dataset d'entraînement du Stanford Cars Dataset, contient plus de 8'000 images réparties en 10 classes, dont font partie les classes de notre sélection. Le nombre d'images de ce dataset pour nos classes varient entre 250 et 1400 images. Nous avons donc conservé uniquement une partie d'elles, afin d'équilibrer et de réduire le dataset que nous allons utiliser.



FIGURE 1 – Visualisation des types de voiture sélectionnés pour la classification

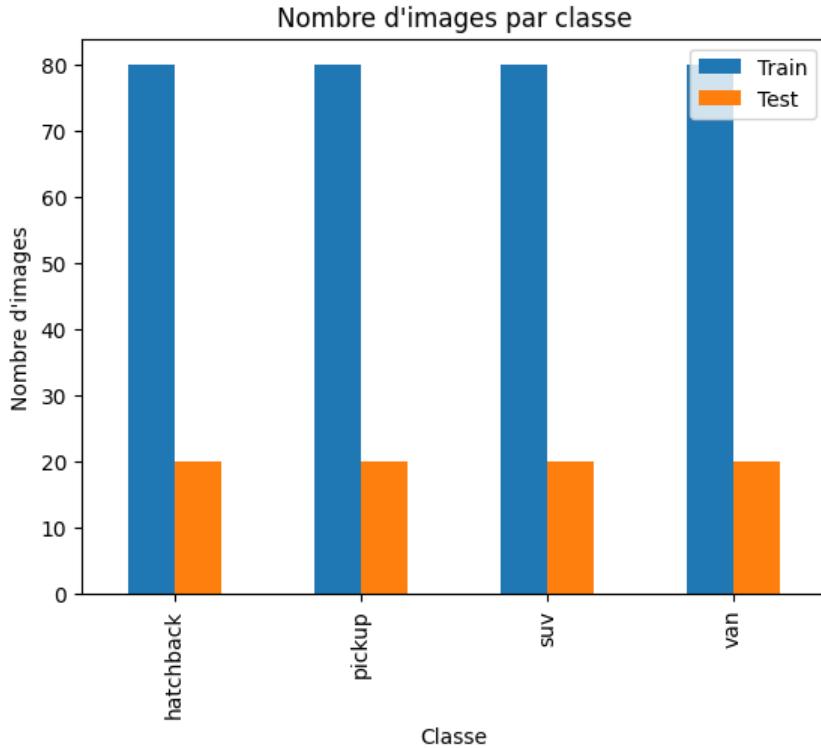


FIGURE 2 – Distribution par classe des images du dataset que nous avons sélectionnées

3 Préparation des données

Lors de la sélection des photos pour notre ensemble de données, nous avons conservé les photos ayant peu d'éléments qui se mélangeaient avec la voiture, comme d'autres voitures ou des bannières dans le fond.

Nous avons également essayé de mélanger des marques et modèles des voitures, pour que le modèle ait une meilleure chance de reconnaître les caractéristiques qui composent un type de voiture.

Pour la suite, nous avons sélectionné aléatoirement 20% des images pour notre ensemble de test, et le 80% restant pour l'entraînement. Les images d'entraînement auront un traitement additionnel pour ajouter des variations, que notre ensemble de validation n'aura pas.

Pour introduire des variations dans les données, nous avons appliqué des effets de miroir, de contraste, de zoom et de rotations pour une partie des images de test. Nos images n'étant pas nécessairement semblables, ces modifications apportent toutefois une possibilité au modèle de se concentrer sur les éléments des voitures qui caractérisent leur type, au-lieu de se concentrer uniquement sur la forme du châssis, qui reconnaîtrait plus une marque qu'un type.

Toutes les images sont redimensionnées au format 224x224 pixels pour avoir une taille uniforme en entrée du modèle.

4 Création du modèle

4.1 Architecture

Notre modèle part de MobileNetV2 avec les poids de *ImageNet*, sur lequel nous allons rajouter une couche dense de 128 neurones qui sera la seule couche que nous allons paramétriser, les couches du MobileNetV2 restant figées.

Étant donné l'ensemble de features important dans les photos du dataset, nous avons fait le choix d'ajouter une régularisation L1 et L2, respectivement *lasso regression* et *ridge regression*, sur notre unique couche dense en 10.2023

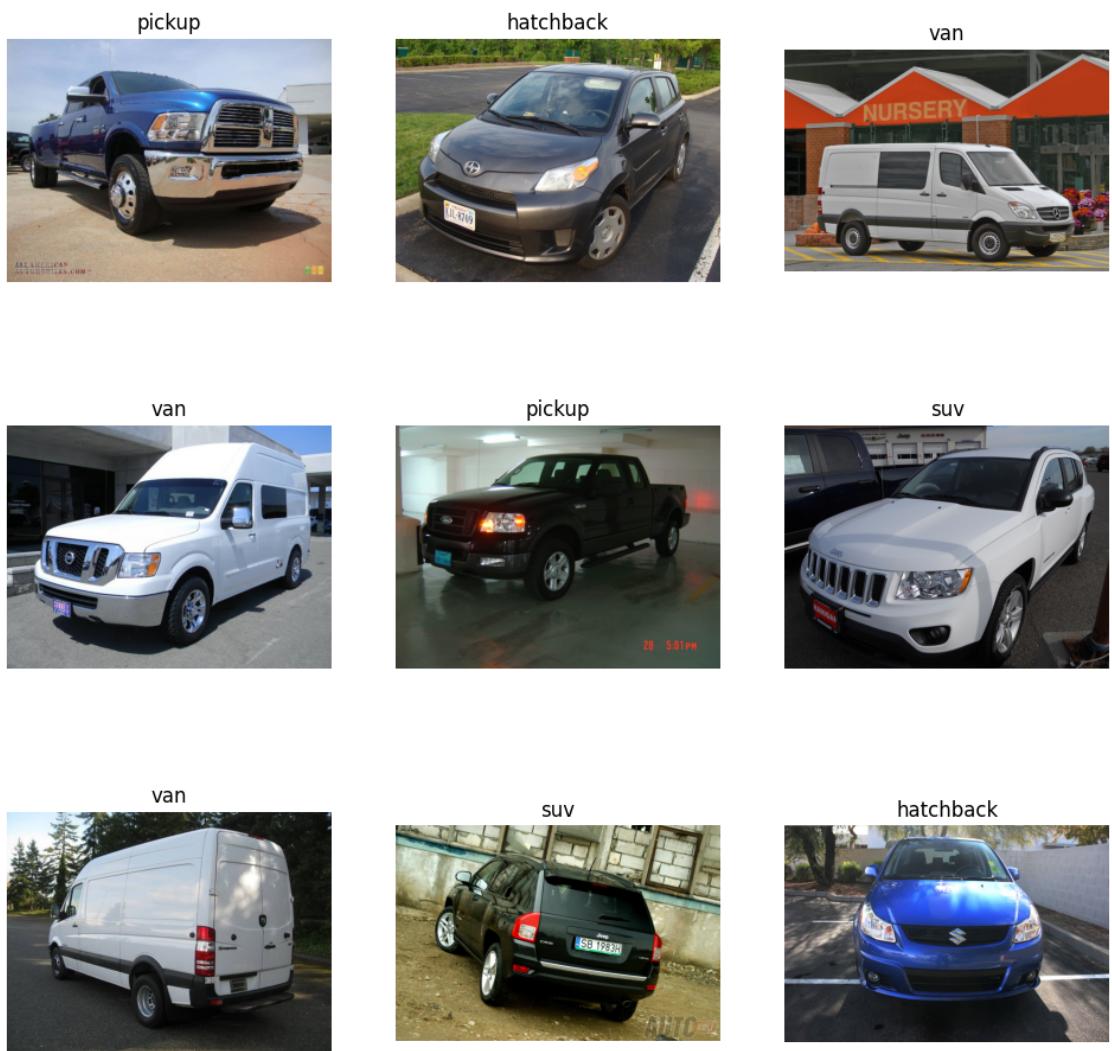


FIGURE 3 – Quelques exemples d’images du dataset

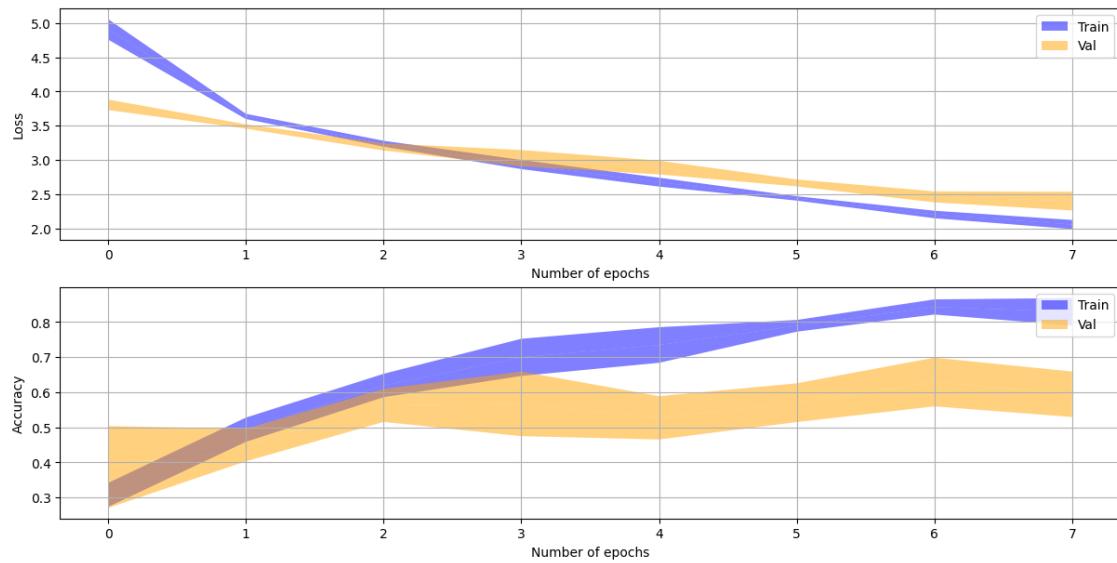


FIGURE 4 – Performance (précision et fonction de coût) de l’entraînement du modèle

sortie de MobileNetV2. Cet effet additionnel sur la fonction de coût permet d'aider le modèle à éliminer certaines features qui ne seraient pas utiles à la classification du problème en limitant aussi le risque d’overfitting.

Nous avons fait quelques essais avec notre ensemble de données en introduisant certaines variations dans les hyper-paramètres pour essayer de trouver un modèle donnant les meilleurs résultats. Le modèle suivant a été sélectionné.

- **Nombre d’epochs :** 8.
- **Optimizer :** RMSprop a été conservé du modèle fourni dans la première partie
- **Learning rate :** Celui par défaut pour RMSprop, 0.001
- **Layers dense :** 1 couche avec 128 neurones et une régularisation L1 et L2
- **Fonction d’activation :** relu

La performance de l’entraînement est donnée en figure 4. Nous notons que le modèle affiche un certain overfitting dans les dernières epochs, que nous avons cherché à réduire en ajoutant de la régularisation L1 et L2, ainsi qu’en introduisant un effet *dropout* après notre couche dense en sortie du modèle. Cela a déjà bien aidé la précision, en introduisant toutefois une variation plus élevée dans les résultats de la fonction de coût. Les expériences suivantes sur ce modèle restant quand même satisfaisantes, et par la difficulté de notre choix de classes initial, nous avons décidé que ce modèle serait suffisant.

Ce modèle possède donc un total de 2,422,468 paramètres, dont seulement 164,484 sont entraînables, le reste étant les poids du modèle *ImageNet*.

4.2 Transfer learning

Pour effectuer le transfer learning, nous avons utilisé le système MobileNetV2, en utilisant, comme mentionné précédemment, les poids de *ImageNet*. Étant donné la petite taille de notre dataset, tous les paramètres d’Imagenet ont été figés, ne laissant que la couche dense comme étant paramétrable. Notre dataset n’étant toutefois pas totalement similaire aux objets qui ont été utilisés pour *ImageNet*, nous aurions probablement pu laisser une partie des paramètres du modèle entraînables, mais nous avons décidé de nous concentrer principalement sur la modification des hyper-paramètres.

Le transfer learning dans notre cas est toutefois très utile. En effet, c'est un moyen efficace d'obtenir un modèle performant capable de différencier certaines features d'une photo et d'analyser de manière plus précise un petit ensemble d'images. Un modèle convolutif classique entraîné sur notre dataset aurait donné des résultats de très mauvaise qualité, et très probablement avec un overfitting inévitable.

Train Loss function: 2.0563

Train Accuracy: 0.8291

Val Loss function: 2.3986

Val Accuracy: 0.5942

F-score hatchback: 0.6207

F-score pickup: 0.8333

F-score suv: 0.5532

F-score van: 0.7857

F1 score global: 0.6982

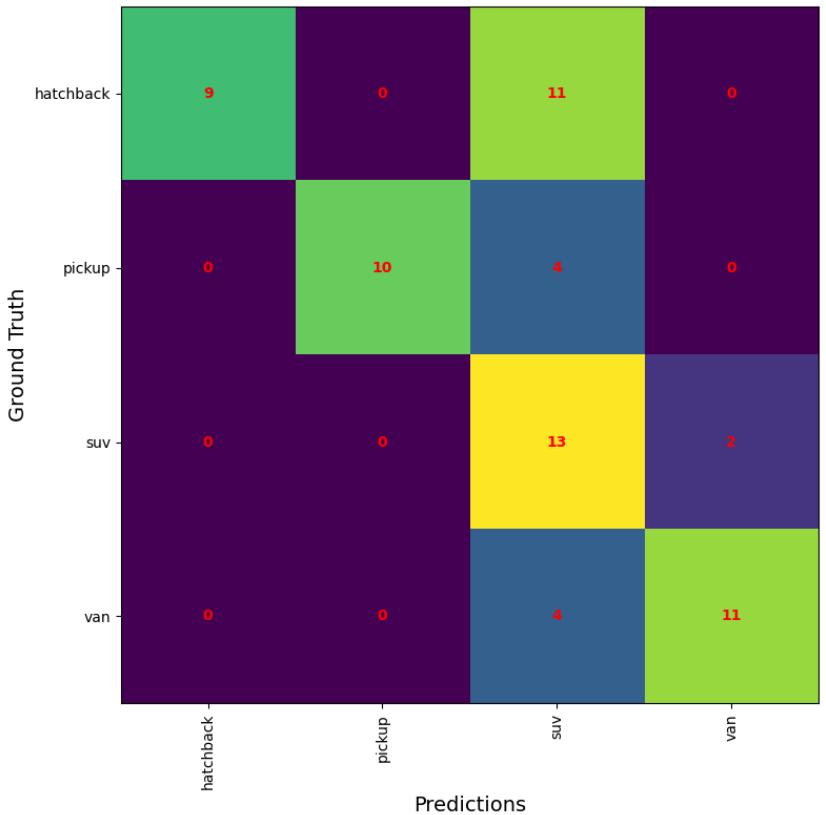


FIGURE 5 – Résultats de la validation du modèle et matrice de confusion

5 Résultats

La performance de l'entraînement est donnée dans la [figure 4](#). La matrice de confusion du système ainsi que les résultats de validation sont dans la [figure 5](#).

Comme nous pouvons l'observer, le modèle a plus de peine avec les prédictions de la classe des SUV. Cela nous semble être assez logique, car la définition d'un SUV ressemble beaucoup à celle d'un hatchback. La différence est principalement la taille, et souvent un hatchback a un chassis arrière plus coupé, tel que présenté dans la [figure 1](#). On pourrait définir le SUV comme étant un véhicule avec une position de conduite plus haute. En revanche, cela peut être difficile à modéliser dans le cadre d'une classification telle qu'elle a été effectuée dans ce laboratoire.

En tenant compte de la difficulté de notre ensemble de données, nous sommes quand même plutôt satisfaits de la performance de notre modèle qui a un f-score relativement bon pour les vans et pickups. Cela nous rassure quant à la possibilité du modèle à séparer les features spécifiques aux types de véhicule. Les SUV et hatchbacks pouvant souvent être confondus par même un humain, les f-score inférieurs ne nous choquent pas trop.

On note une fonction de coût relativement élevée qui est un résultat probable de la régularisation L1 et L2 appliquée dans la couche dense du modèle. Le facteur de différence entre le test et l'entraînement est cependant relativement correct, avec le coût de validation 1.2 fois plus élevée, ce qui n'indique pas forcément qu'il y a un overfitting.

En application réelle, le modèle affiche quand même quelques incertitudes, comme observé dans la [figure 6](#), les pourcentages pour le van et le pickup restent inférieurs à 50%, et les pourcentages fluctuaient beaucoup en fonction de la position de la caméra. Cela peut être dû à un overfitting des données, mais nous pensons que c'est plus parce que les caractéristiques que le modèle cherche à reconnaître pour certains types ne sont parfois pas mises en avant. Dans l'ensemble, une fois la position de la caméra stabilisée, nous avons quand même observé des pourcentages relativement peu changeants, bien que quelques hésitations soient encore parfois présentes.

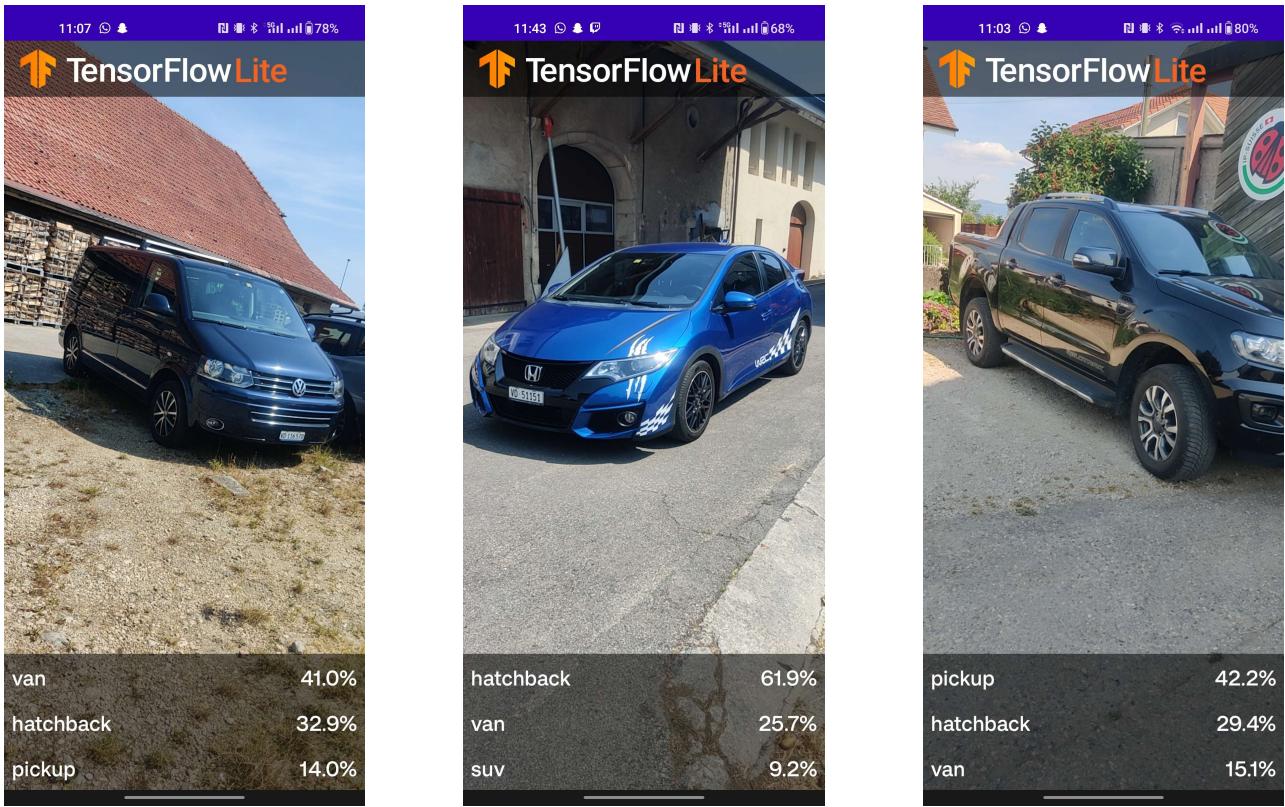


FIGURE 6 – Essais en vrai avec l’application. Catégories (gauche à droite) : van, hatchback, pickup.

Les résultats de la grad-cam, en [figure 7](#), sont plutôt bons. On remarque sur certaines images que le modèle regarde un peu plus que la voiture, et sur d’autres des parties trop spécifiques. Néanmoins, dans l’ensemble, le modèle a une tendance à analyser la voiture dans son ensemble, ce qui est pour nous indicatif que les observations sont faites en grande partie par rapport aux caractéristiques des voitures.

Les essais réels ont montré quelques soucis sur certains types de véhicules. En effet, un type de voiture très courant en Suisse est le *crossover SUV*, un SUV plus petit, beaucoup plus proche des hatchbacks. Nous pouvons voir dans la [figure 8](#) nos résultats pour ces *crossovers*, qui affichent en général une catégorie hatchback. Cela ajoute donc une certaine difficulté à notre modèle étant plus entraîné à classifier des SUV plus grands, comme on peut voir en Amérique. Cela entre en relation avec les résultats de la matrice de confusion en [figure 5](#), où l’on avait une confusion visible entre hatchback et SUV, ce qui pourrait aussi ajouter à la difficulté de la distinction entre les deux types.

On observe dans la grad-cam des mauvaises classifications en [figure 10](#) que le modèle se concentre dans l’ensemble bien les caractéristiques des voitures, avec parfois des régions trop spécifiques comme nous l’avons observé dans la [figure 7](#) des images d’entraînement. Cela nous indique que la mauvaise classification provient plus d’une mauvaise observation des caractéristiques de la voiture que d’une observation d’un élément n’étant pas le sujet de la photo. Cela pourrait être ajusté en ayant plus de données en entrée, ou en introduisant encore plus de photos modifiées.

Enfin, si on essaie de classifier des voitures n’étant pas des voitures, comme en [figure 11](#), nous pouvons voir que le modèle affiche des résultats appartenant souvent à la catégorie des vans. Une découverte assez étonnante, qui peut être liée au fait que les vans ressemblent à beaucoup d’autres types de véhicules et indiquerait que le modèle est moins capable de généraliser ce genre de véhicules et deviendrait du coup une solution « catch-all », prenant un certain manque de caractéristiques comme étant une caractéristique pour les vans, mais cela pourrait aussi être une coïncidence.

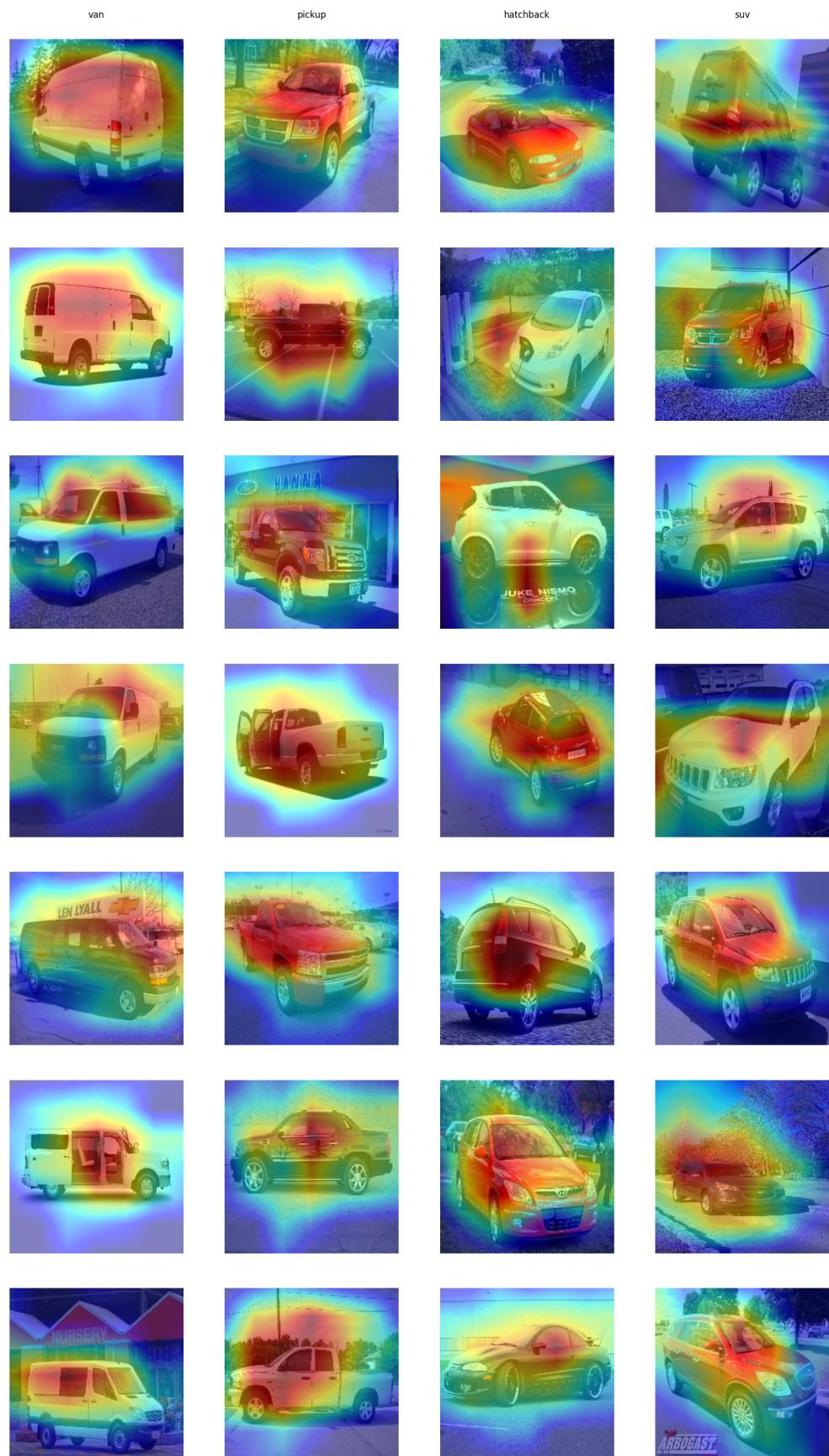


FIGURE 7 – Grad-cam de l'entraînement du modèle

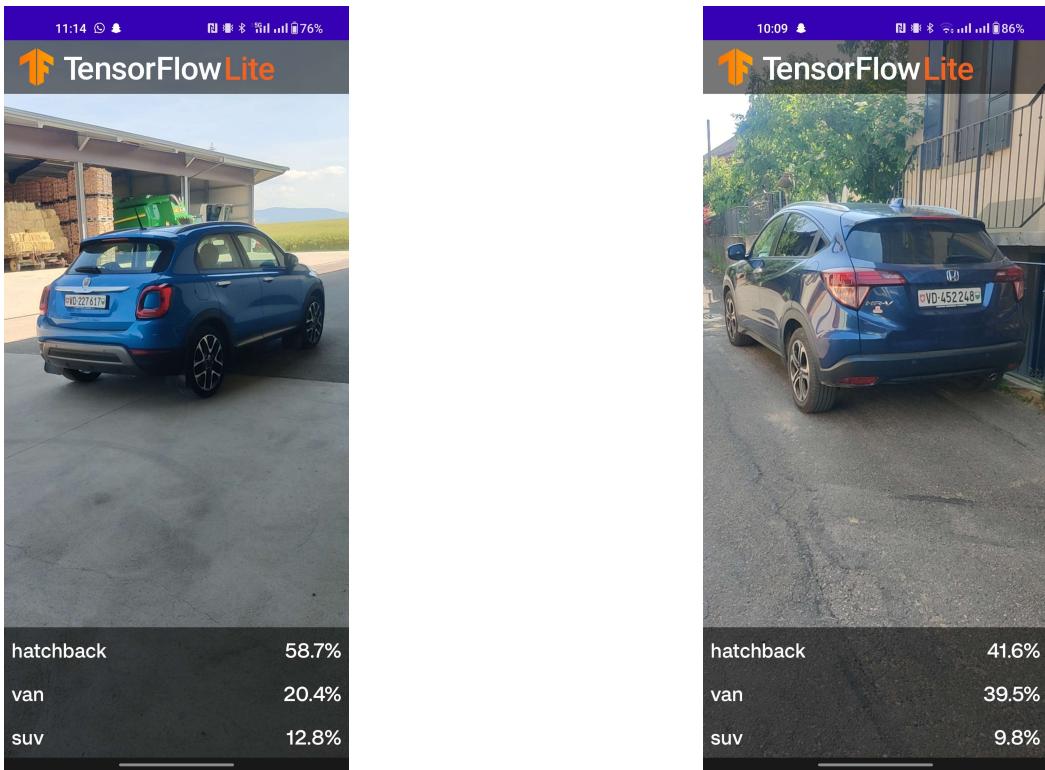


FIGURE 8 – Démonstration des erreurs de classification sur des véhicules de type *crossover SUV*

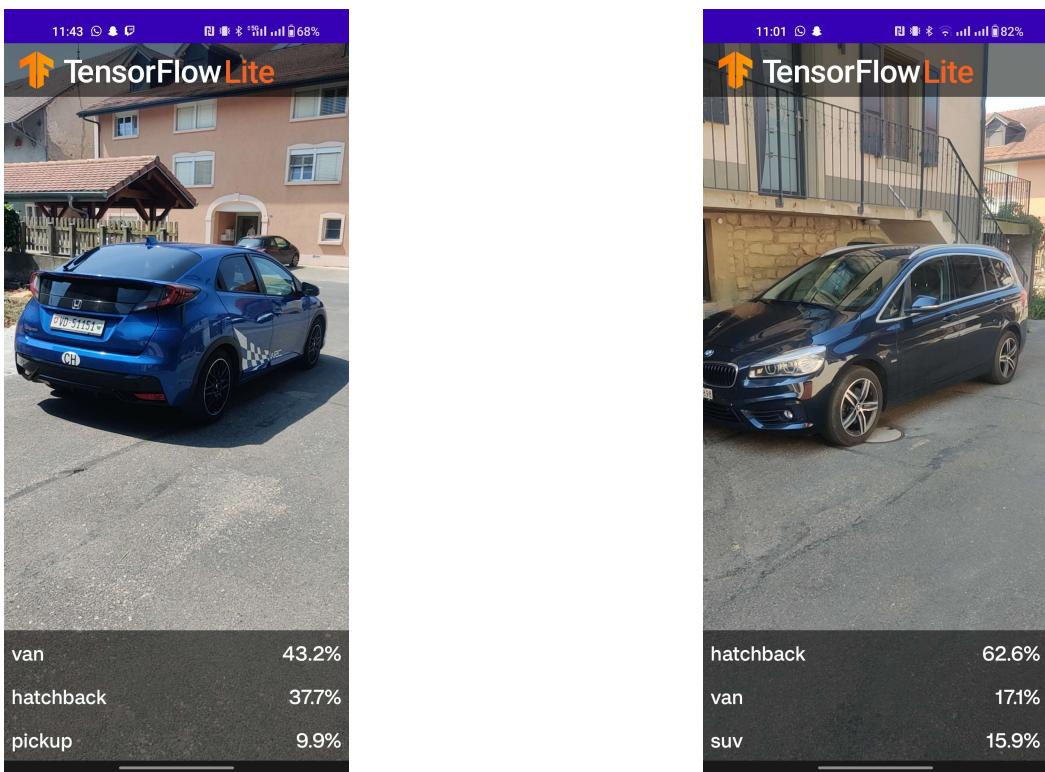


FIGURE 9 – Exemples de mauvaises classifications. Catégories réelles (gauche à droite) : hatchback, suv.

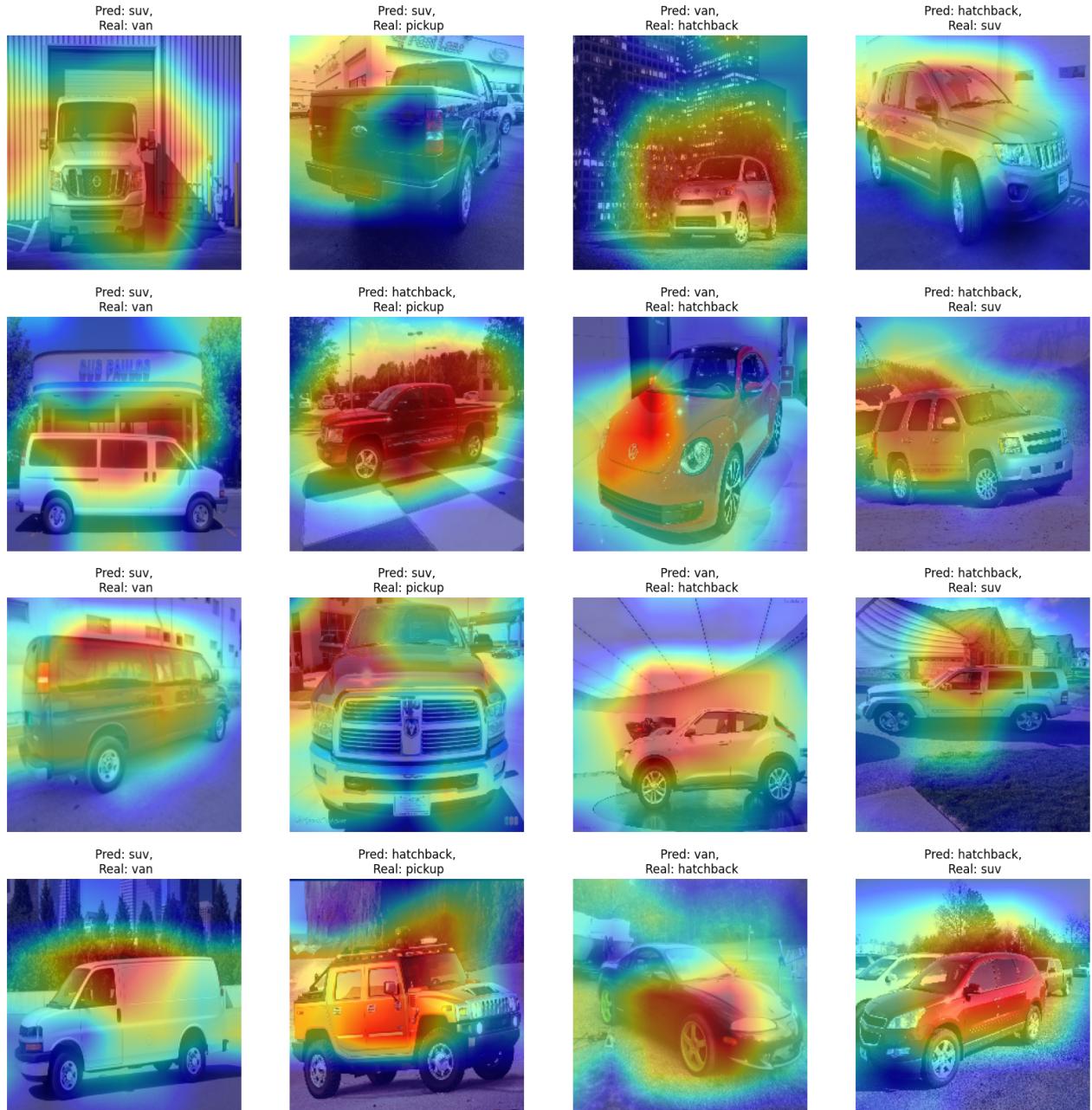


FIGURE 10 – Grad-cam des images de validations mal classifiées

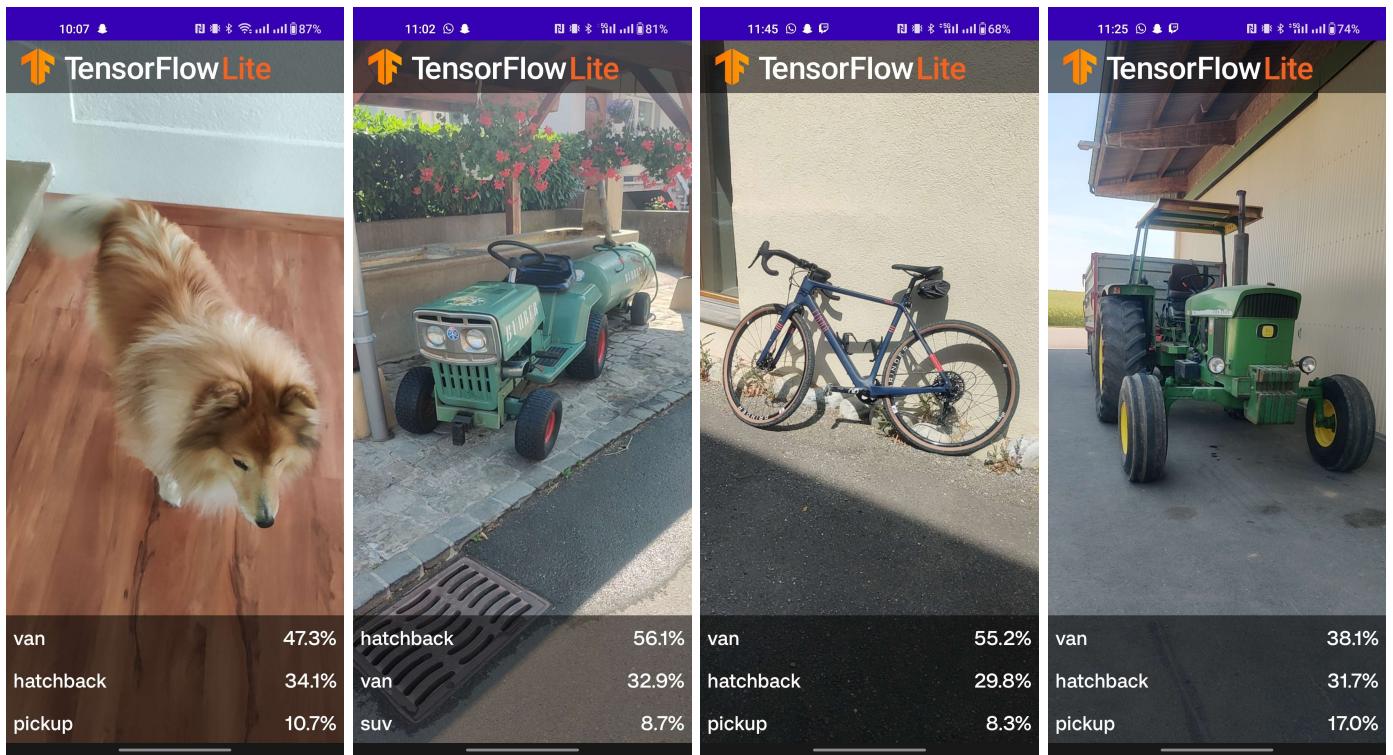


FIGURE 11 – Essais en vrai avec des sujets autres.

Pour conclure, les résultats obtenus dans l’application réelle correspondent assez bien à nos attentes, simultanément celles avant de commencer le projet, et celles après avoir analysé les résultats de la validation de l’entraînement. Nous sommes assez contents avec les régions que le modèle est capable d’analyser, et nous n’avons pas remarqué des difficultés avec des fonds différents (gravier, asphalte, béton, etc.).

Comme discuté précédemment, le modèle a quelques difficultés avec les hatchbacks et les SUV, et cela a aussi été visible dans les résultats de l’application réelle. Ces confusions sont probablement liées aux ressemblances de ces types de véhicules et n’est pas aidé par la présence d’un type que nous n’avons pas entraîné, les *crossovers*, qui devraient être catégorisés sous le type SUV étant donné que c’est un sous-type. Cependant, la taille étant différente, le modèle n’arrive pas à déterminer un type certain pour ce genre de véhicules. Cela pourrait être contré avec un entraînement sur plus d’images et notamment plus d’images de ce type de véhicules.

D’autres améliorations du dataset ont été données dans les paragraphes précédents. Probablement qu’avec plus de temps d’entraînement et en appliquant les suggestions susmentionnées, nous aurions pu obtenir un modèle ayant une meilleure capacité à distinguer les classes que nous avons choisi de garder.

6 Conclusion

Ce laboratoire aura été très intéressant, même si notre choix de classes a probablement été trop complexe. Un ensemble de classes plus facile et plus proche des objets reconnus par *ImageNet* aurait probablement causé moins de problèmes dans la classification. Nous avons rencontré plusieurs difficultés lors de la mise en place du dataset pour avoir quelque chose d’intéressant pour le modèle et pour la classification, et cela s’est évidemment répercuté sur les résultats finaux. Parmi ces difficultés, on retrouve évidemment le choix de nos classes, dont la diversité intra-classe et les similarités inter-classes rendent la classification plus compliquée que si l’on avait choisi des objets plus ordinaires.

Nous sommes quand même grandement satisfaits des résultats obtenus et pensons que l’expérience nous aura malgré tout formé sur l’utilisation du transfer learning ainsi que sur l’importance de la sélection initiale des classes et de l’ensemble de données initial.

Nous pensons donc que les résultats approuvent notre problématique initiale assez complexe, et, considérant que *ImageNet* est entraîné sur des objets ordinaires du quotidien, la classification finale est satisfaisante pour une première expérience sur le transfer learning et les réseaux convolutifs complexes.

La classification telle qu'effectuée ici vient quand même avec ses limites. En effet, les types de voiture autres que hatchback, SUV, pickup, et van ne pourront évidemment pas être classifiées correctement. L'utilisation de ce modèle reste donc très restreinte dans l'état actuel.

Un travail futur pourrait partir des observations effectuées en incorporant un set de données bien plus grand et avec un nombre bien plus élevé de types de véhicules, en considérant les modèles de voitures plus récents, que nous n'avons pas pu avoir dans notre dataset. Il pourrait également être utile de fournir une catégorie supplémentaire « autre » afin que les objets observés qui ne sont pas des véhicules y soient classés.

Nous sommes d'avis qu'en incorporant toutes les améliorations suggérées, notre modèle a le potentiel de s'améliorer encore et d'être moins sensible aux perturbations lorsqu'il s'agit de prédire le type d'une voiture à partir d'une image non conventionnelle. En outre, avec un peu plus de temps, nous aurions pu identifier un modèle plus efficace en modifiant le nombre de couches et de neurones dans chaque couche. Une option aurait pu être d'exécuter un notebook pendant la nuit avec un ensemble complet d'hyperparamètres à évaluer, ce qui nous aurait permis de faire une sélection plus précise de ces hyperparamètres.

En résumé, nous sommes extrêmement satisfaits de nous être engagés dans le développement d'un cycle complet d'une application de classification d'objets. La navigation au travers de la procédure complète aux évaluations finales sur un appareil mobile, nous a éclairés sur les subtilités de la création d'applications du transfer learning. En outre, la création d'une application qui a des applications tangibles dans le monde réel, qui peut immédiatement être mise à l'épreuve dans la vie quotidienne, a été très gratifiante.