Classification de fromages par apprentissage profond

DAUBE Victor
CentraleSupélec
victor.daube@student-cs.fr

GRÈS Paola
CentraleSupélec
paola.gres@student-cs.fr

Abstract

La gastronomie française est aujourd'hui reconnue dans le monde entier, le fromage étant un des ingrédients phares de cette richesse gustative. Depuis sa création, la recette du fromage s'est développée, aboutissant à plusieurs centaines de variétés de fromages différents. Aujourd'hui, avec les avancées technologiques, l'intelligence artificielle est devenue un outil puissant à mettre au service de la reconnaissance d'aliments, en particulier les fromages. Cet outil peut permettre de faire évoluer l'industrie agro-alimentaire ainsi que celle de la restauration autonome, en calculant par exemple le prix d'un plateau repas en fonction des éléments qui le composent, comme le type de fromage choisi. Dans ce projet, nous nous focalisons sur 43 variétés de fromages, et nous comparons les performances de plusieurs réseaux de neurones avec différents paramètres. Le nombre de classes considérées étant plutôt élevé par rapport à la quantité d'images par classe, nous aurons recours à l'augmentation de données pour élargir le dataset. Enfin, nous explorerons d'autres pistes pour améliorer les performances, comme le voting classifier.

1. Introduction

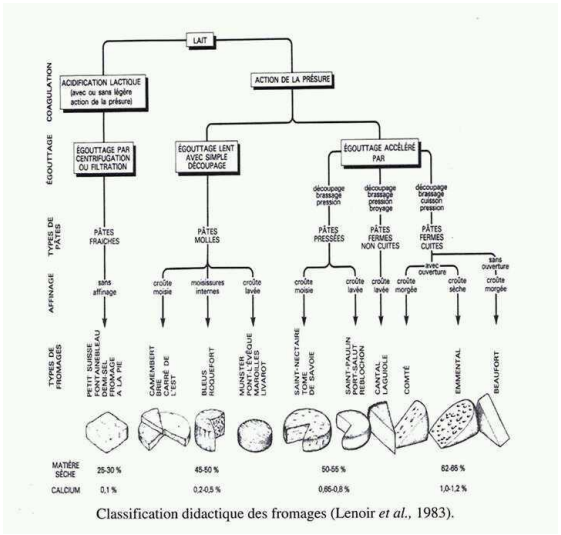
Dans ce projet, nous nous focalisons sur la reconnaissance de fromages. On estime que l'histoire du fromage remonte au néolithique, son invention pouvant être liée à la pratique du transport de lait dans des vessies faites de caillettes de ruminants, sources de présure. Arrivé en Europe durant la période héllénistique, le fromage est aujourd'hui devenu un élément central des repas quotidiens en France. Il est donc intéressant d'apprendre à les reconnaître, notamment dans le cadre des restaurants self-service. Le développement d'applications permettant aux acteurs industriels de facturer automatiquement des plateaux repas est en effet en plein essor [1]. On note par ailleurs l'émergence d'applications mobiles permettant aux utilisateurs de trouver rapidement l'origine d'un produit alimentaire grâce à la vision par ordinateur.

Il existe des centaines de variétés de fromages, de tous types : pâtes molles, pâtes dures, pâtes pressées, pâtes persillées, fromage frais, fromage blanc, etc [2]. Cependant, pour des questions de temps et de complexité, nous avons décidé de réduire le problème à un nombre limité de classes de fromages :

- A Filetta - Beaufort - Bethmale - Bleu d'Auvergne - Bleu de Gex - Bons Mayennais - Boulette d'Avesnes - Brie de Meaux - Brin d'amour - Cabécou - Cantal - Chaource - Charolais - Chaumes - Comté - Coulommiers - Couronne Lochoise - Crottin de Chavignol - Emmental de Savoie - Fourme d'Ambert - Gaperon - Gruyère français	- La vache qui rit - Mimolette - Morbier - Munster - Neufchâtel - Ossau-iraty - P'tit basque - Pont l'évêque - Pouligny-saint-Pierre - Raclette - Reblochon - Rocamadour - Rochebaron - Roquefort - Saint-nectaire - Sainte-maure - Tomette de brebis - Tomme de Savoie - Tomme des Pyrénées - Valençay - Vieux Boulogne

Ce projet a pour but de construire un classifieur capable de reconnaître les 43 types de fromages ci-dessus. Nous allons pour ce faire tester différentes approches, de la construction from scratch d'un réseau, à l'utilisation de réseaux pré-entraînés en transfert learning, avec ou sans

fine tuning. Nous allons également expérimenter l'augmentation de données ainsi que l'utilisation de voting classifier, afin d'affiner la précision de notre modèle.



Classification didactique des fromages (Lenoir *et al.*, 1983).

Figure 1 : Classification didactique des fromages (Lenoir et al 1983)

2. Etat de l'art

2.1. Classification d'images

La classification d'images est un sujet à l'étude depuis plusieurs décennies. Des approches basées sur l'extraction de features à l'aide de filtres puis la classification à l'aide d'algorithmes de machine learning (Support Vector Machine, k-Nearest Neighbours etc…) ont été proposées [3]. Toutefois, les modèles aboutissant aux résultats les plus spectaculaires sont aujourd'hui les réseaux de neurones comportant des couches convolutives. La popularité de ces modèles a explosé dans le cadre de la "troisième vague connexionniste" dans les années 2010. Cette troisième vague a été notamment rendue possible par la croissance des données disponibles et de la puissance de calcul (GPGPU). Ainsi, une grande variété d'architectures reposant sur les CNN (convolutive neural networks) ont vu le jour comme par exemple les ResNet [4]. Ces architectures permettent d'augmenter le nombre de couches du réseau sans dégradation de performance grâce à la modélisation du résidu. Les réseaux MobileNet [5] ont de leur côté permis d'optimiser l'utilisation des ressources de calculs grâce à la convolution séparée (depthwise separable convolution).

2.2. Classification de nourriture

Les études sur la classification d'images de fromages étant très rares, nous avons axé nos recherches sur la classification de plats. De nombreuses études ont été réalisées sur le sujet, car la classification de plats est très complexe compte tenue de la diversité de plats possible ainsi que l'étendue de variations possibles pour un même plat. La plupart des études se basent sur un réseau convolutif pré-entraîné. Une étude a été faite pour comparer différents réseaux sur des problèmes de classification de nourriture [6], les réseaux comparés étant ResNet-18, Inception-V3, Resnet-50, Densenet-121, Wide Resnet-50 et ResNext-50. Dans cette étude, le Resnet-50 apparaît comme le plus précis, mais les résultats d'accuracy restent très bas. En effet, une autre étude se focalise sur la classification de nourriture basée sur le Resnet-50 [7], et les résultats d'accuracy sont aux alentours de 40% pour 100 classes.

2.3 Ensemble Learning

Les méthodes d'ensemble learning [8] consistent en la combinaison de plusieurs modèles dont l'apprentissage a été réalisé dans des conditions différentes. Le concept est de "moyenner" plusieurs hypothèses afin de réduire le risque de se tromper en bénéficiant du phénomène de sagesse des foules. Un système de vote peut par exemple être utilisé entre différents modèles de machine learning afin de moyenner les décisions des modèles, et aboutir à des performances plus intéressantes que celles des modèles considérés individuellement [9].

3. Approche

3.1. Dataset

Nous avons utilisé un dataset d'images de fromages disponibles sur Kaggle. La dataset contenait initialement 252 fromages différents. Bien que nous n'ayons pas trouvé d'information sur la méthode de constitution de ce dataset, une exploration de celui-ci nous a amené à penser qu'il a été constitué dans une démarche de webly supervised learning (récolte d'images par requête dans un moteur de recherche sur le web). En effet, pour un nombre considérable de classes, les images recueillies étaient fortement bruitées. Par exemple, la classe pour le fromage "Cap Noir" était uniquement constituées d'images de casquettes noires, ce qui doit probablement être un résultat plus pertinent du point de vue d'un moteur de recherche en ligne.

Ainsi nous avons révisé l'ensemble du dataset et supprimé à la main les classes pour lesquelles les données semblaient anormalement bruitées. Nous avons également supprimé les classes ayant un nombre trop faible d'images. Ainsi nous avons conservé 43 classes de ce dataset, pour 3814 images au total. Les images contiennent chacune un ou plusieurs fromages, tranchés ou entiers. Nous ferons l'hypothèse qu'il n'y a qu'une seule classe de fromage présente dans chaque image.

Nous avons séparé le dataset comme suit :

- train : 2288 images (56%)
- val : 572 images (14%)
- test : 954 images (30%)

Figure 2 : Exemples d'images de fromages contenues dans le dataset

L'ensemble d'entraînement contient, finalement, peu d'images par classe (en moyenne 89 par classes, au minimum 81), nous allons donc utiliser de l'augmentation de données pour pallier ce problème. Nous comparerons également les performances avec et sans augmentation.

Les images du dataset sont globalement de bonne qualité dans la mesure où peu d'images sont floues. Du point de vue de la classification, les images sont de difficulté variable. Si sur certaines images le fromage apparaît très distinctement comme l'élément central, il peut être partiellement masqué sur d'autres images et occuper une place moins importante.

Figure 3 : Deux exemples d'images pour la classe Rocamadour avec un exemple plutôt "facile" à gauche et un exemple plus difficile à droite

3.2. Classification

Pour la classification, nous avons utilisé 5 réseaux différents, que nous avons entraînés sur le jeu de données de base, puis sur le jeu de données augmenté :

- Un CNN à 4 couches (deux couches convolutives suivies de deux couches complètement connectées).
- Un modèle MobileNet-v2 pré-entraîné, disponible dans la librairie pytorch, en transfert learning simple.
- Ce même réseau en fine tuning.
- Un modèle ResNet-18 pré-entraîné, également disponible dans la librairie pytorch, en transfert learning.
- Ce même réseau en fine tuning.

L'apprentissage en transfert learning simple consiste ici à régler finement uniquement les paramètres de la dernière couche du réseau préentrainé (couche complètement connectée). En fine tuning, en revanche, l'ensemble des paramètres du réseau sont réglés finement. Les réseaux ResNet 18 et MobileNet v2 ont été pré-entraînés avec les 1.2M images de la base de données ILSVRC (1000 classes).

3.3. Data augmentation

Compte tenu du faible nombre d'images contenues dans notre dataset en comparaison au nombre important de paramètres à optimiser, nous avons utilisé de l'augmentation de données, multipliant ainsi par 6 notre jeu d'entraînement. Nous avons utilisé pour cela des transformations simples :

- Couleur (saturation, contraste, luminosité, température)
- Rognage (induisant un zoom)
- Retournement (horizontal et vertical)
- Taille (induisant un “flou”)
- Rotation

Figure 4 : Illustration des transformations simples du dataset

Nous avons préféré ne pas introduire d’opérations de distorsion des images, et cela afin de ne pas introduire de biais dans le dataset. En effet, la classe d’un fromage est liée à ses proportions. Finalement nous obtenons le dataset suivant pour les expériences avec augmentation de données : train 13728 images (76%) ; val 3432 images (19%) et test 954 images (5%). L’ensemble de test est resté inchangé par rapport aux expériences précédentes.

3.4 Métrique

La métrique utilisée afin de mesurer la performance des modèles entraînés est l’accuracy. En effet nous sommes dans un problème de classification multiclasse et il n’y a pas de classe qui soit a priori plus importante qu’une autre dans notre problème. C’est pour cela que nous avons retenu l’accuracy qui reflète la performance globale sur l’ensemble des classes sans donner plus de poids à une classe qu’à une autre.

4. Expériences

4.1. Entraînement sans optimisation des hyperparamètres

Tous les entraînements ont été réalisés avec des epochs de taille 10 dans le but de limiter les temps d’entraînement. Pour certaines expériences, nous avons bien conscience qu’une prolongation de l’entraînement sur un nombre plus élevé d’epochs aurait pu augmenter l’accuracy obtenue. Par exemple, nous constatons sur le graphique ci-dessous que l’accuracy de validation ne semble pas converger même après 10 epochs.

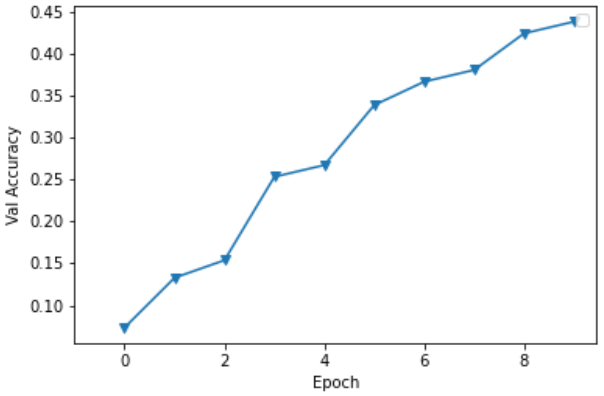


Figure 5 : Accuracy sur l’ensemble de validation lors de l’entraînement du réseau CNN

Les entraînements de nos modèles ont été réalisés en utilisant le GPU mis à disposition par Google Collab. Il s’agit d’accélérateurs NVIDIA Tesla K80.

Pour chacune des expériences présentées dans ce rapport, un early stopping round a été mis en place. Cela consiste en la mémorisation des poids ayant conduit à l’accuracy la plus élevée sur le dataset de validation au cours de l’entraînement. Cette technique nous permet notamment d’éviter des situations d’overfitting où le modèle surperforme sur les données d’entraînement mais affiche une performance moindre sur le dataset de validation. C’est par exemple ce que l’on peut observer sur le graphique ci-dessous.

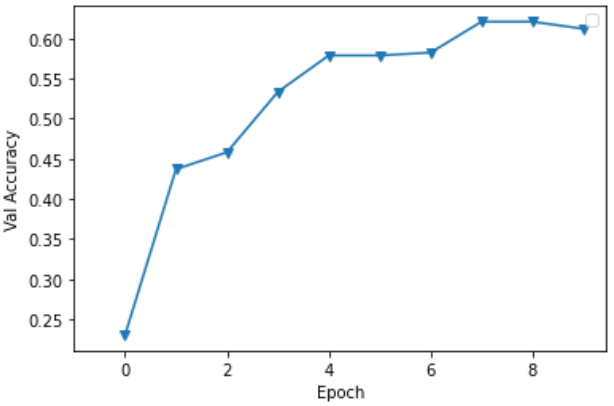


Figure 6 : Accuracy sur l’ensemble de validation lors de l’entraînement du réseau MobileNet en transfert learning simple. Diminution de l’accuracy à partir de l’epoch 8

Un premier apprentissage a été réalisé sans optimisation des hyperparamètres. Pour ce premier entraînement, nous avons arbitrairement choisi d'utiliser des minibatch de taille 32, un optimiseur à descente de gradient stochastique (optimiseur SGD), un taux d'apprentissage de 0.001 et un momentum de 0.9. Nous avons obtenu les résultats synthétisés dans le Tableau 1.

	Accuracy validation	**Accuracy test**
CNN	43,9%	**41,3%**
ResNet18 Transfer	52,8%	**51,2%**
ResNet18 Fine tuning	75%	**74,4%**
MobileNet Transfer	62,1%	**58,7%**
MobileNet Fine tuning	79,4%	**76,5%**

Tableau 1 : Accuracy de validation et de test pour le premier tour d'entraînement

Nous constatons que pour les réseaux ResNet et MobileNet, le fine tuning donne une accuracy significativement plus élevée. Ainsi l'apprentissage en transfer learning de la dernière couche complètement connectée semble beaucoup moins efficace que l'apprentissage de tous les paramètres du réseau. On peut en conclure que les featuresextraites par les différentes couches convolutives de la version pré-entraînée ne sont pas très pertinentes pour effectuer de la classification de fromage. En nous intéressant aux 1000 classes du dataset ILSVRC utilisé pour entraîner les modèles ResNet et MobileNet, nous avons observé que bien qu'il contienne des classes issues du domaine de l'alimentation (guacamole, ice-cream…), il n'y a pas vraiment de classe en lien avec le fromage. Ceci peut expliquer pourquoi nous observons cet écart entre le transfer learning simple et le fine tuning.

4.2. Tuning des hyperparamètres

Ayant des ressources de calcul limitées, nous avons décidé de réaliser l'optimisation des hyperparamètres les uns après les autres, retenant à la fin de chaque optimisation le meilleur hyperparamètre concerné pour passer au suivant.

Nous avons tout d'abord optimisé la structure du CNN, en modifiant la taille du noyau et de la première couche convolutive. Nous retenons un CNN avec une taille de noyau de 6 et une première couche convolutive de taille 32.

Nous avons ensuite testé sur chaque réseau les performances pour les tailles de batch suivantes : 8, 16 et 32.

	8 (val)	16 (val)	32 (val)	**test**
CNN	42,0%	43,7%	42,5%	**40,7%**
ResNet18 Transfer	59,6%	58,6%	52,8%	**57,3%**
ResNet18 Fine tuning	77,8%	77,4%	76,7%	**74,3%**
MobileNet Transfer	64,7%	65,6%	61,9%	**62,9%**
MobileNet Fine tuning	79,0%	80,0%	78,8%	**75,5%**

Tableau 2 : Meilleure accuracy par taille de batch pour chaque réseau, et accuracy sur l'ensemble de test pour la meilleure valeur de taille de batch.

Nous gardons donc une taille de batch de 8 pour le ResNet, 16 pour le MobileNet et 32 pour le CNN. Nous pouvons constater que pour certains modèles, l'accuracy obtenue est inférieure à celle du premier tour d'entraînement (cf Tableau 1) et cela malgré l'optimisation de la taille des minibatch. Nous pensons que cela est dû au fait que l'utilisation des GPU google Collab ne garantit pas une reproductibilité complète des résultats entre plusieurs sessions d'exécution. Cela signifierait donc que l'amélioration apportée par l'optimisation de la taille des minibatch est inférieure à la variation d'accuracy due au caractère stochastique de l'entraînement. Nous tenons à préciser que chaque expérience a été menée lors d'une unique session d'exécution, ce qui ne remet donc pas en cause les conclusions tirées à la fin de chaque expérience, la reproductibilité étant garantie sur une même session.

Nous avons ensuite testé les optimiseurs SGD et Adam, en faisant varier le taux d'apprentissage et le weight decay. Nous avons testé trois formats possibles pour l'évolution du taux d'apprentissage : un taux constant d'une valeur de 1e-3, un taux cyclique triangulaire avec des valeurs comprises entre 1e-5 et 2e-3 et un taux multiplicatif de valeur initiale 1.5e-3 et multiplié par 0.9985 à chaque epoch. Chaque configuration est itérée avec un weight decay prenant les valeurs 0 et 1e-4 pour chacun des réseaux, nous donnant 6 configurations à comparer pour chaque réseau.

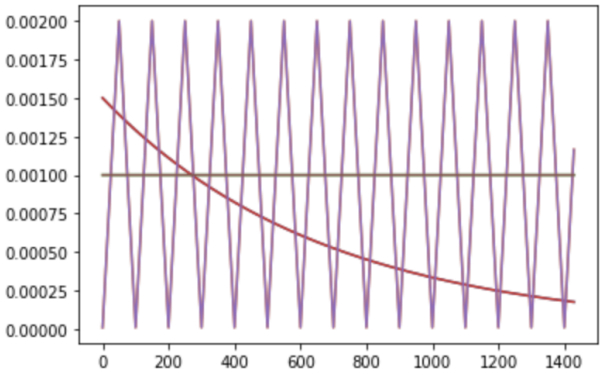


Figure 7 : Evolution du taux d'apprentissage selon 3 schedulers : cyclique (violet), multiplicatif (rouge) et sans scheduler (marron)

Comme observé précédemment lors de l'entraînement sans optimisation des hyperparamètres, les performances du CNN sont en dessous des réseaux pré entraînés sur l'ensemble de test.

Réseau	Optimizer	Batch size	Weight decay	Scheduler	Accuracy (test)
CNN	SGD	32	0	multi	40,5%
ResNet18 Transfer	Adam	8	0	cyclic	59,5%
ResNet18 Fine tuning	SGD	8	0.0001	No	74,9%
MobileNet Transfer	Adam	16	0	multi	68,2%
MobileNet Fine tuning	SGD	16	0.0001	multi	77,0%

Tableau 3 : Accuracy pour chaque réseau sur l'ensemble de test pour la meilleure configuration Batch size - Weight decay - Scheduler - Optimizer retenue

Pour chaque réseau, les performances sont également bien meilleures en fine tuning qu'en transfer learning simple. Le MobileNet et le ResNet ont des performances similaires en fine tuning, bien que le MobileNet soit légèrement au-dessus.

4.3. Classification avec data augmentation

Pour chacun des cinq réseaux présentés précédemment, nous avons retenu l'optimiseur qui aboutissait à la meilleure accuracy sur l'ensemble de validation. Nous avons relancé chacun des réseaux puis nous avons comparé les résultats d'accuracy avec les résultats sur le dataset de base.

Réseau	Accuracy (test)	Accuracy (test) sur dataset augmenté
CNN	40,5%	47,5%
ResNet18 Transfer	59,5%	64,2%
ResNet18 Fine tuning	74,9%	80,3%
MobileNet Transfer	68,2%	68,6%
MobileNet Fine tuning	77,0%	78,6%

Tableau 4 : Accuracy pour chaque réseau sur l'ensemble de test pour la meilleure configuration d'hyperparamètres retenue, après entraînement sur le dataset augmenté

L'augmentation de données améliore très peu l'accuracy pour le MobilNet, mais cette amélioration est bien plus évidente pour le ResNet et le CNN. Étonnamment, alors que le MobileNet était meilleur sur le dataset de base, le ResNet est ici meilleur sur le dataset augmenté. Cela peut-être dû au fait que l'augmentation de données ne suffit pas pour que l'architecture du MobileNet n'induise pas d'overfitting, d'autant plus que la taille de batch sélectionnée est 2 fois supérieure au ResNet.

Nous pouvons remarquer que l'accuracy n'est pas du tout homogène entre les classes (cf Figure 8). En effet, on note de faibles taux d'accuracy pour la classe Tomme des Pyrénées (35%) et les classes Tomette de brebis, Ossau-Iraty et Coulommiers (environ 50%). En observant les images de plus

près, on remarque que ces faibles taux concernent des classes dans lesquelles la représentation du fromage est très variée (cf Figure 9), comme des emballages très différents avec des textes différents, ou des images non classifiables par l'œil humain. Cependant, 56% des classes obtiennent un score d'accuracy supérieur à 80%, et seulement 9% en-dessous de 60%.

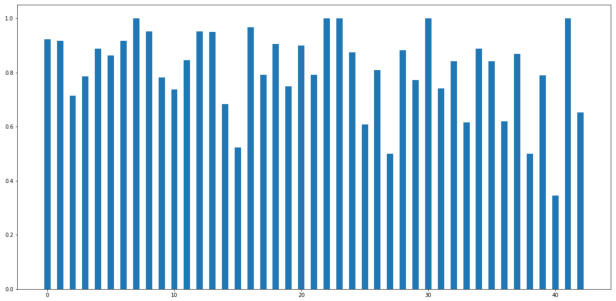


Figure 8 : Accuracy par classe sur l'ensemble de test (taille de classe allant de 16 à 31)

Figure 9 : Exemples de classes de fromages pour lesquelles l'accuracy est <= 50% : Coulommiers en haut, Tomme des Pyrénées en bas

4.4. Voting classifier

Nous avons finalement implémenté un voting classifier en utilisant les quatre réseaux suivants :ResNet Transfer Learning (TL), ResNet Fine Tuning (FT), MobileNet TL et MobileNet FT. Nous avons effectivement retiré le CNN qui affichait des résultats moins bons jusqu'ici. Les accuracy sur l'ensemble de test de ces quatre réseaux sont rappelées sur la figure ci-dessous :

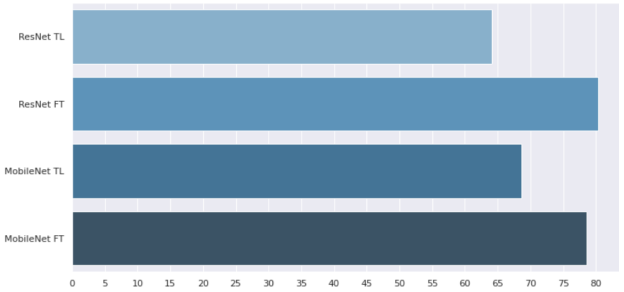


Figure 10 : Accuracy des 4 modèles retenus

Nous avons implémenté trois différents types de voting classifier. Tout d'abord nous avons défini un hard voting classifier : la classe choisie est la classe majoritaire parmi les prédictions des quatre modèles. Ensuite vient le soft voting classifier : pour chaque classe, nous ajoutons la probabilité que l'image appartienne à cette classe selon chaque modèle. La prédiction est la classe correspondant à la probabilité cumulée la plus élevée. Les prédictions en sortie des réseaux de neurones sont transformées en probabilités en utilisant la fonction softmax. Enfin a été implémenté un weighted soft voting : il s'agit d'un voting classifier de type soft voting dans lequel les probabilités d'appartenance à une classe sont pondérées par l'accuracy de chaque modèle. Nous obtenons les résultats sur la figure ci-dessous :

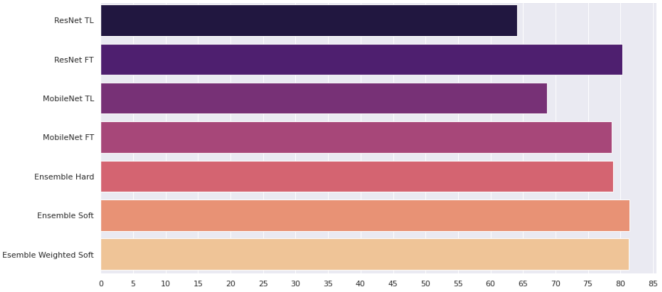


Figure 11 : Accuracy des voting classifiers

Si le hard voting ne présente pas d'intérêt car il aboutit à une accuracy inférieure à celle du meilleur modèle (78.8% pour le soft voting contre 80.3% pour ResNet FT), le soft voting et le weighted soft voting ont permis d'atteindre une accuracy supérieure à celle du meilleur modèle (respectivement 81.4% et 81.2%). Les deux modèles entraînés en transfer learning simple ayant donné une accuracy significativement moins bonne que les deux modèles en fine tuning, nous avons voulu tester ces différents voting classifiers en retenant seulement les modèles obtenus après fine tuning. Néanmoins, cela n'a pas donné de

meilleurs résultats que dans la version précédente utilisant les quatre modèles.

5. Conclusion

Pour conclure, notre principal frein sur ce projet était nos ressources limitées en calcul. Nous sommes conscients que nos performances auraient pu être améliorées grâce à l'implémentation d'un véritable GridSearch ou même d'un random search, si nous avions accès à une puissance de calcul plus conséquente. Cependant, nous avons eu l'occasion de tester 5 modèles différents et d'optimiser les hyperparamètres sur quelques valeurs. Les réseaux pré-entraînés ont montré des performances bien supérieures que le CNN implémenté from scratch, comme l'on s'y attendait, tandis que ces réseau donnaient de meilleurs résultats en fine-tuning qu'en transfer learning simple. Enfin, le réseau ResNet-18 a donné un score d'accuracy légèrement meilleur que le MobileNet, avec 80,3% contre 78,6%. L'augmentation de données a été décisive pour l'augmentation de ce score, bien qu'elle n'ait eu qu'un léger impact sur les performances du MobilNet. Pour finir, le voting classifier a permis d'améliorer encore ce score, passant de 80,3% à 81,4%.

Pour aller plus loin, nous aurions pu améliorer l'augmentation de données en utilisant un GAN, permettant de créer des images complètement nouvelles pour combler le dataset existant. Nous pourrions également tester un plus grand nombre de paramètres dans l'étape d'optimisation des hyperparamètres. Par exemple, nous n'avons pas fait varier la valeur initiale du taux d'apprentissage.

6. Références

[1] Sicara, a Theodo group company, Facturation automatique de plateaux repas, https://www.sicara.fr/etudes-de-cas/facturation-automatique-plateaux-repas

[2] Lenoir, J., Lamberet, G. & Schmidt, J. L. (1983). L'élaboration d'un fromage: l'exemple du Camembert. Pour la science, 30-42

[3] Sebastian Berisha, IMAGE CLASSIFICATION USING GABOR FILTERS AND MACHINE LEARNING, May 2009, Wake Forest University

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren et Jian Sun, Deep Residual Learning for Image Recognition, Dec 2015, arXiv:1512.03385v1

[5] Howard, Andrew & Zhu, Menglong & Chen, Bo & Kalenichenko, Dmitry & Wang, Weijun & Weyand, Tobias & Andreetto, Marco & Adam, Hartwig. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.

[6] S. Memiş, B. Arslan, O. Z. Batur and E. B. Sönmez, "A Comparative Study of Deep Learning Methods on Food Classification Problem," 2020 Innovations in Intelligent Systems and Applications Conference (ASYU), 2020, pp. 1-4, doi: 10.1109/ASYU50717.2020.9259904.

[7] Z. Zahisham, C. P. Lee and K. M. Lim, "Food Recognition with ResNet-50," 2020 IEEE 2nd International Conference on Artificial Intelligence in Engineering and Technology (IICAIET), 2020, pp. 1-5, doi: 10.1109/IICAIET49801.2020.9257825.

[8] Dietterich T. (2000) Ensemble Methods in Machine Learning. Proc. 1st workshop on Multiple Classifier Systems, Sardaigne, Italie

[9] Peppes, N.; Daskalakis, E.;Alexakis, T.; Adamopoulou, E.; Demestichas, K. Performance of Machine Learning-Based Multi-Model Voting Ensemble Methods for Network Threat Detection in Agriculture 4.0. Sensors 2021, 21, 7475. https://doi.org/10.3390/s21227475