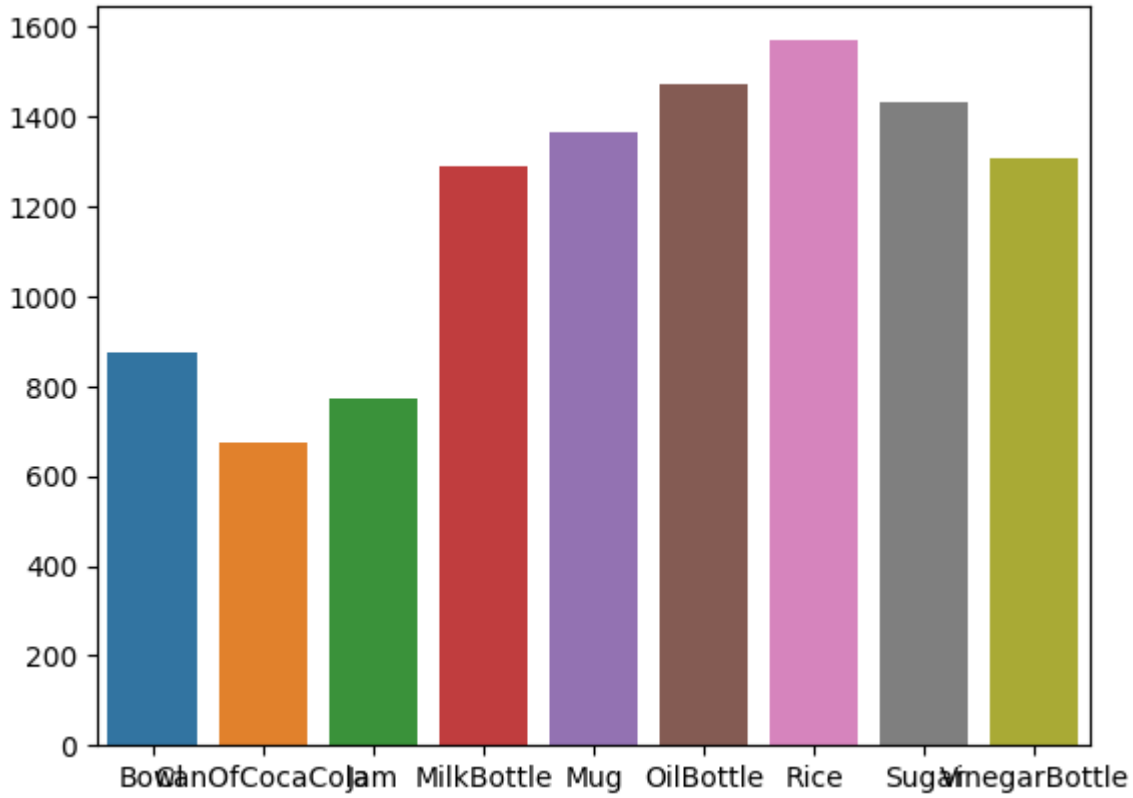
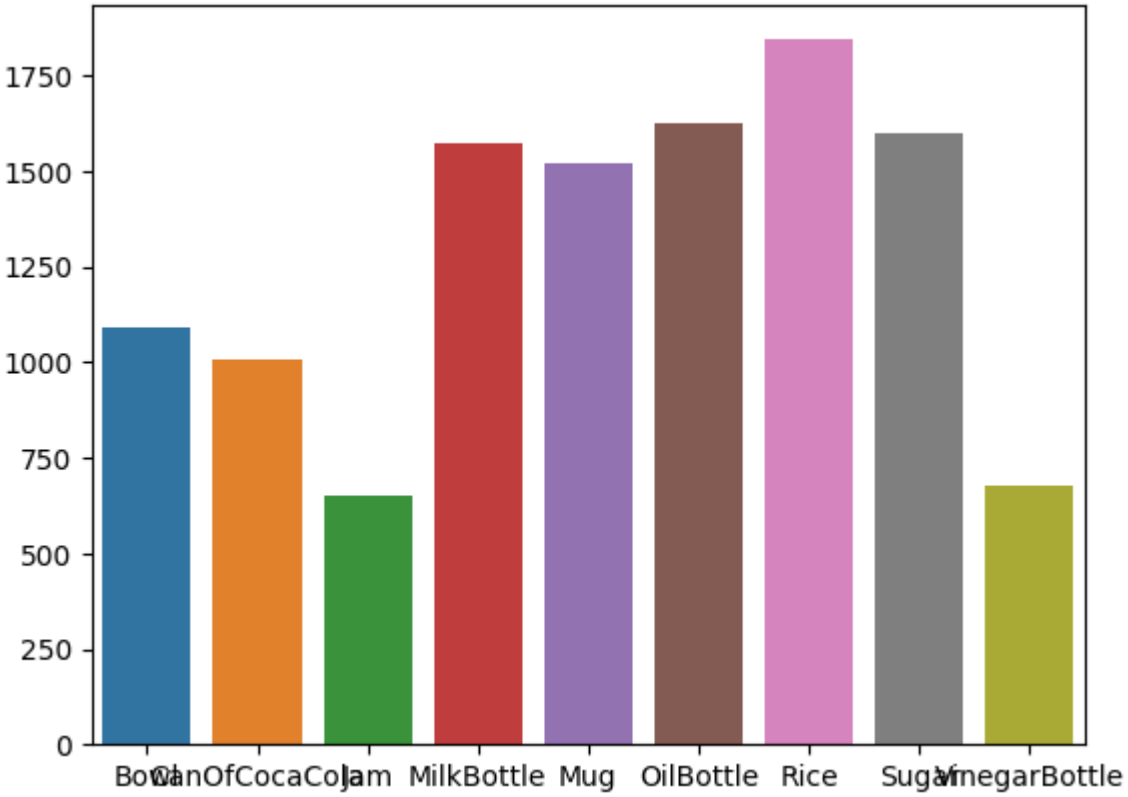
# Analyse de vidéo

*présenté par Bardisbanian Lucas et Kafak Jacques*

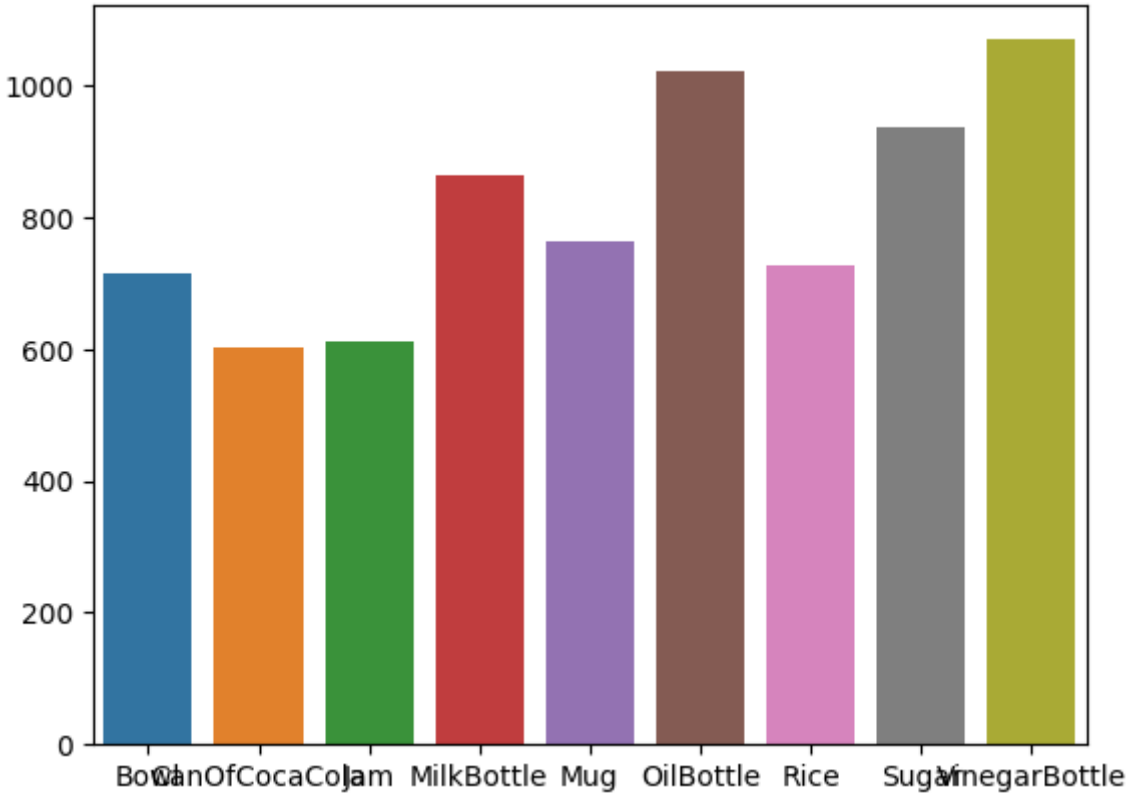
## Analyse du jeu de données




*Répartition du jeu de données d'entraînement - saliency maps*




*Répartition du jeu de données d'entraînement - bounding boxes*



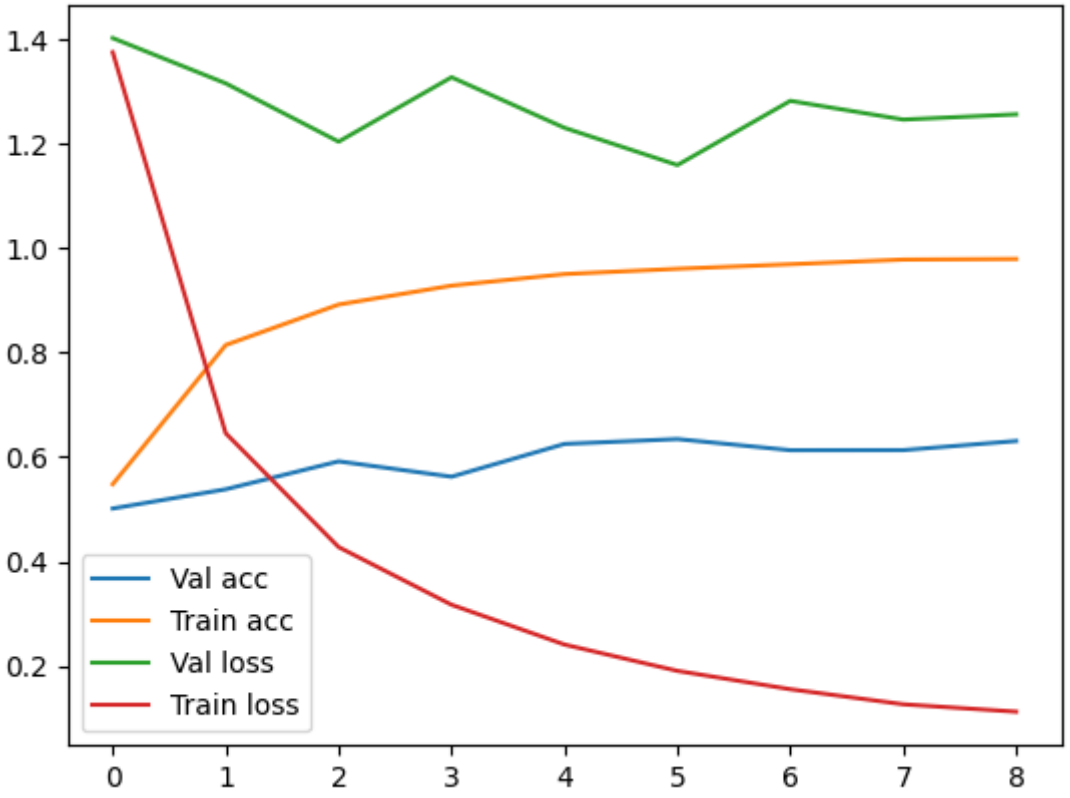

*Répartition du jeu de données de test*

On remarque un déséquilibre entre les variables, spécifiquement Coca-Cola et Jam sont en sous effectifs et également la proportion de Rice et de VinegarBottle est déséquilibrée entre le jeu de données de test et le jeu de données d'entraînement.

Pour les bounding boxes également, les données sont en sous effectifs pour les variables Jam et VinegarBottle

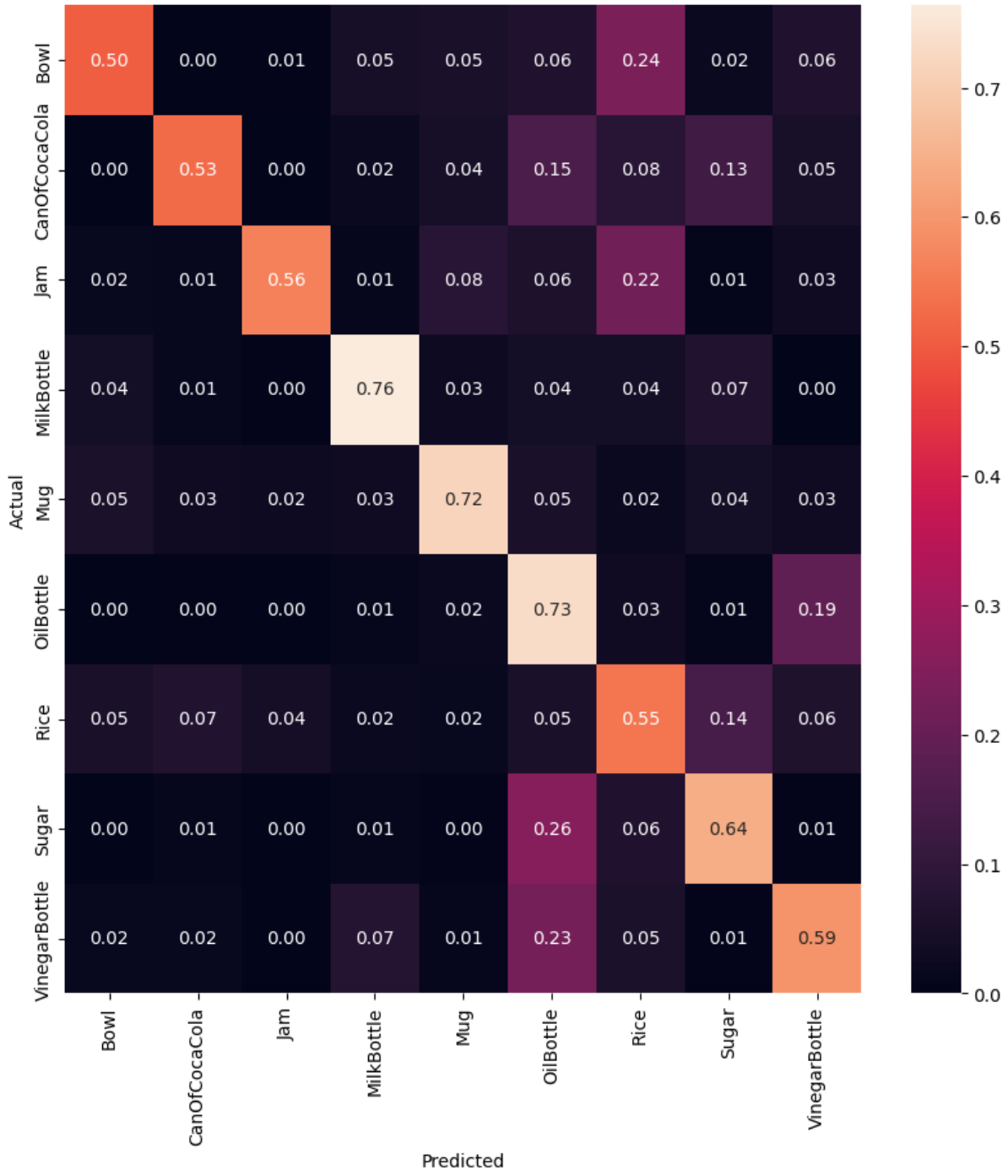
## Modèle de base

Nous avons décidé de partir sur du transfer learning en nous basant sur le modèle EfficientNet V2 et plus précisément la variante B3 (le papier est accessible ici: https://arxiv.org/abs/2104.00298). Nous avons choisi ce modèle car il a un bon ratio efficacité / nombre de paramètres. Nous avons freeze les poids de ce modèle puis ajouté une simple couche de classification de 9 neurones avec une fonction d'activation softmax. Deux premiers modèles ont été entraînés, l'un avec les saliency maps, l'autre avec les bounding boxes.

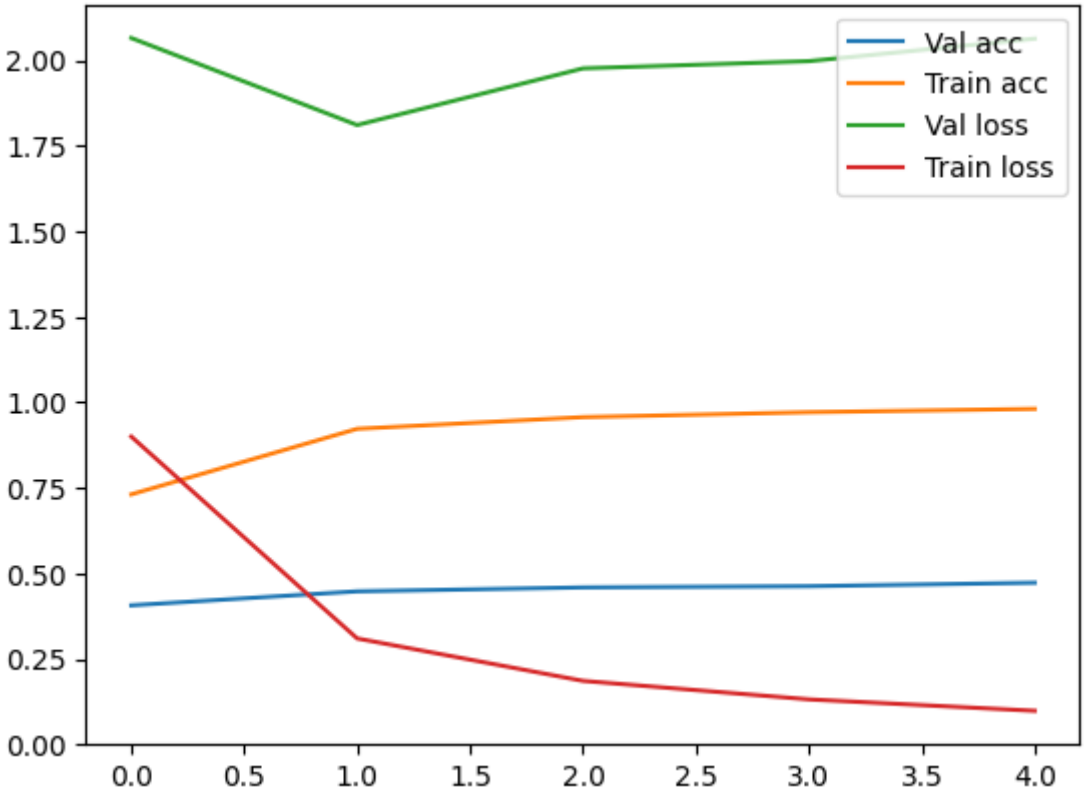



*Courbe d'entraînement - saliency maps*

final Val accuracy: 0.63

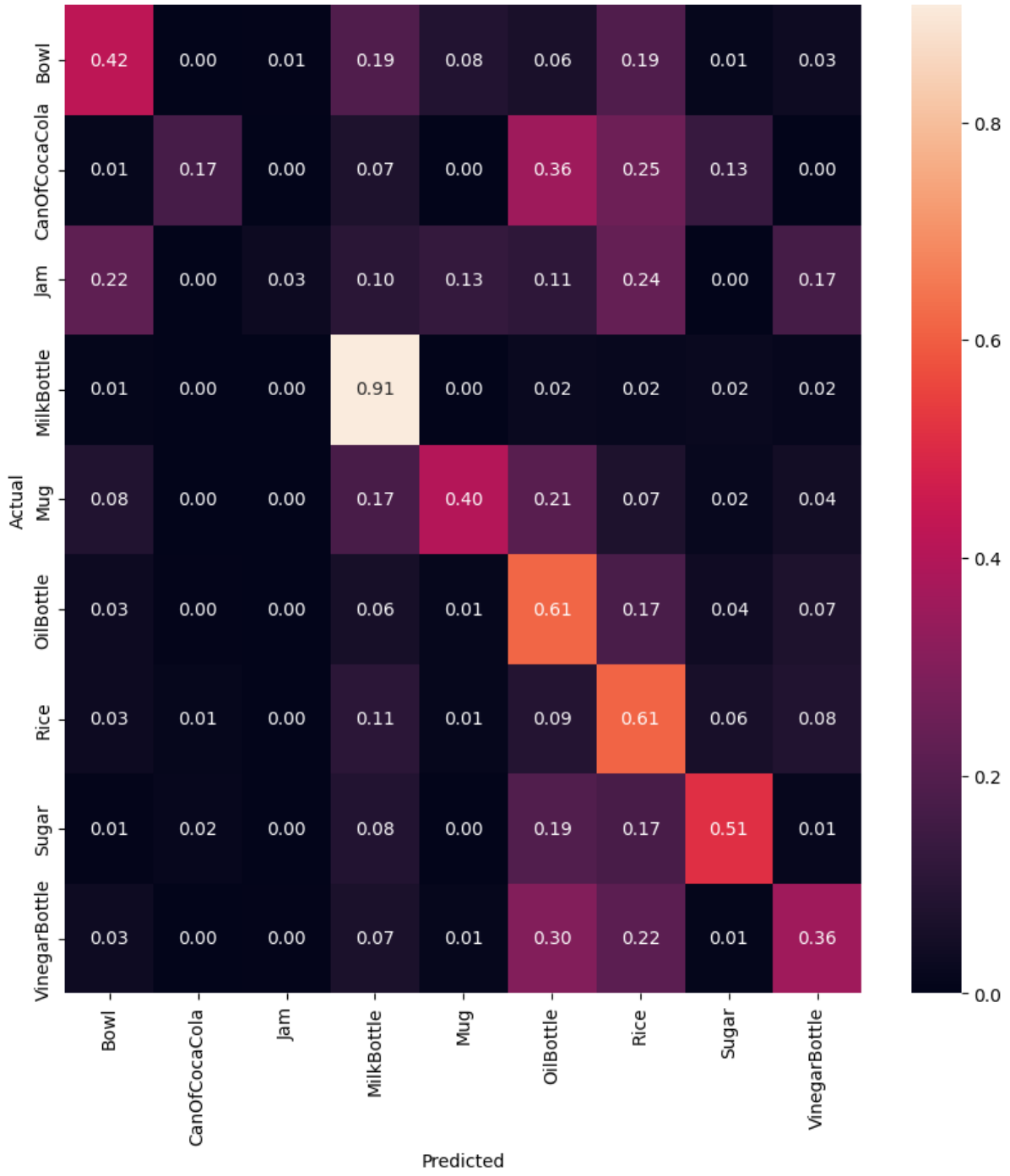



*Matrice de confusion - saliency maps*




*Courbe d'entraînement - bounding boxes*

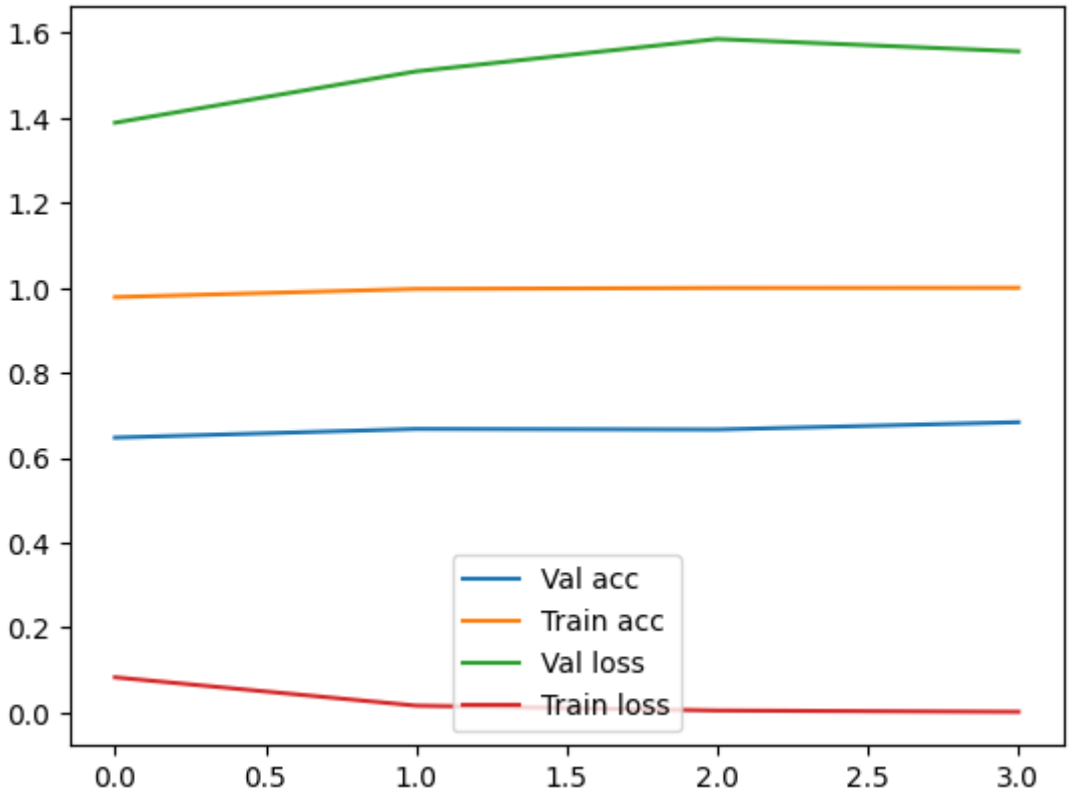
final Val accuracy: 0.47



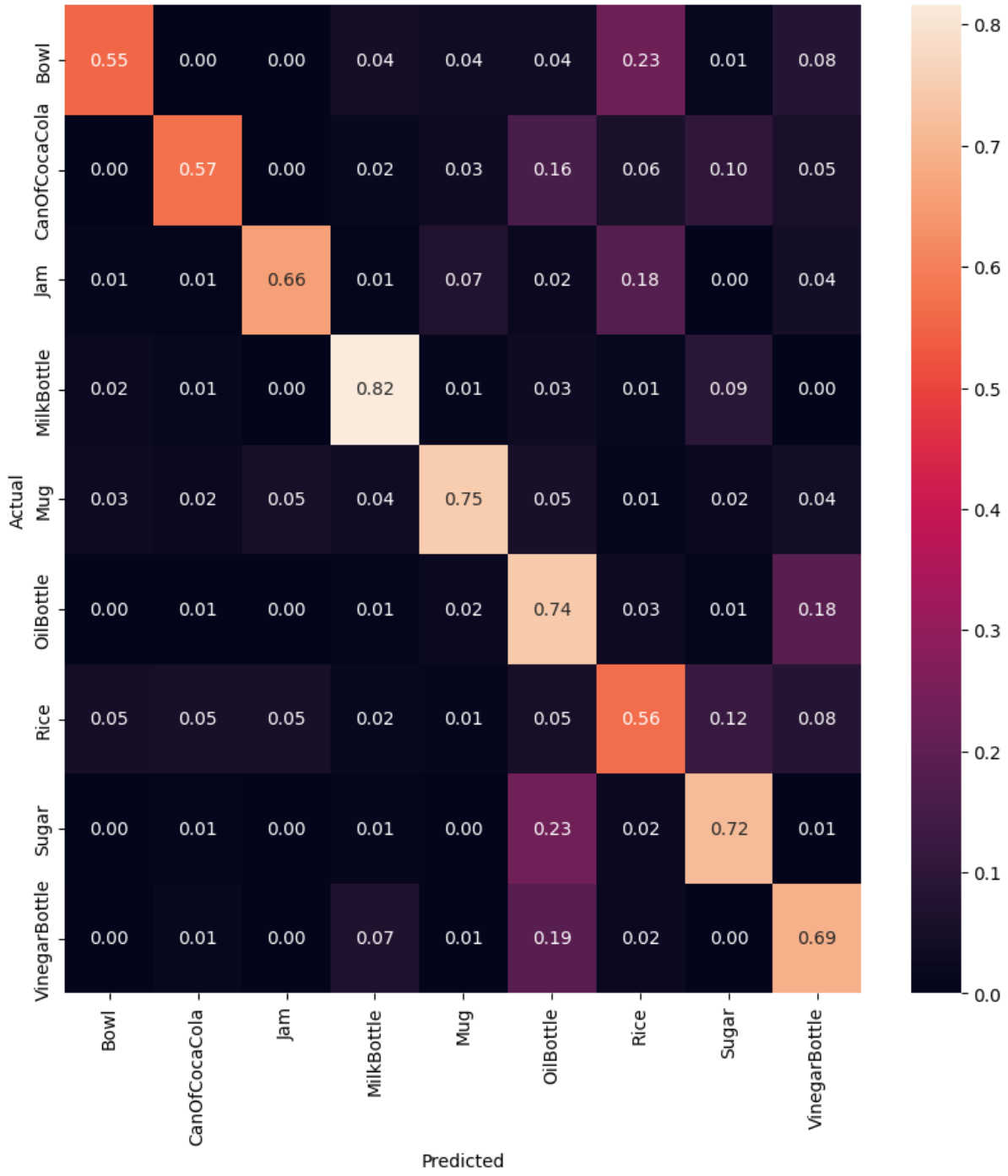

*Matrice de confusion - bounding boxes*

On remarque que le modèle avec les bounding boxes ne prédit quasiment jamais “Jam”, et cette classe était largement sous représentée dans le jeu de données associés. De même, pour le modèle avec les saliency maps, on remarque plus de difficultés sur 3 classes sous représentées: “Bowl”, “CanOfCocaCola” et “Jam”. Également, pour la classe “Rice”, qui est sur représenté dans les données d’entraînement par rapport à celle de test, et à l’inverse “VinegarBottle” qui est sur représenté dans les données de testing par rapport à celles d’entraînement.

Bien que les bounding boxes offrent des données d'entraînement plus qualitatives, le testing est toujours réalisé sur des données en saliency maps. Nous supposons que la différence dans la qualité de ces données résulte dans la perte d'accuracy constatée ici. Nous avons continué notre modèle de base en réalisant du fine tuning sur le modèle en saliency maps, qui nous permet d'avoir une hausse d'accuracy assez conséquente.




*Courbe d'entraînement - saliency maps - fine tuning*
final Val accuracy: 0.68

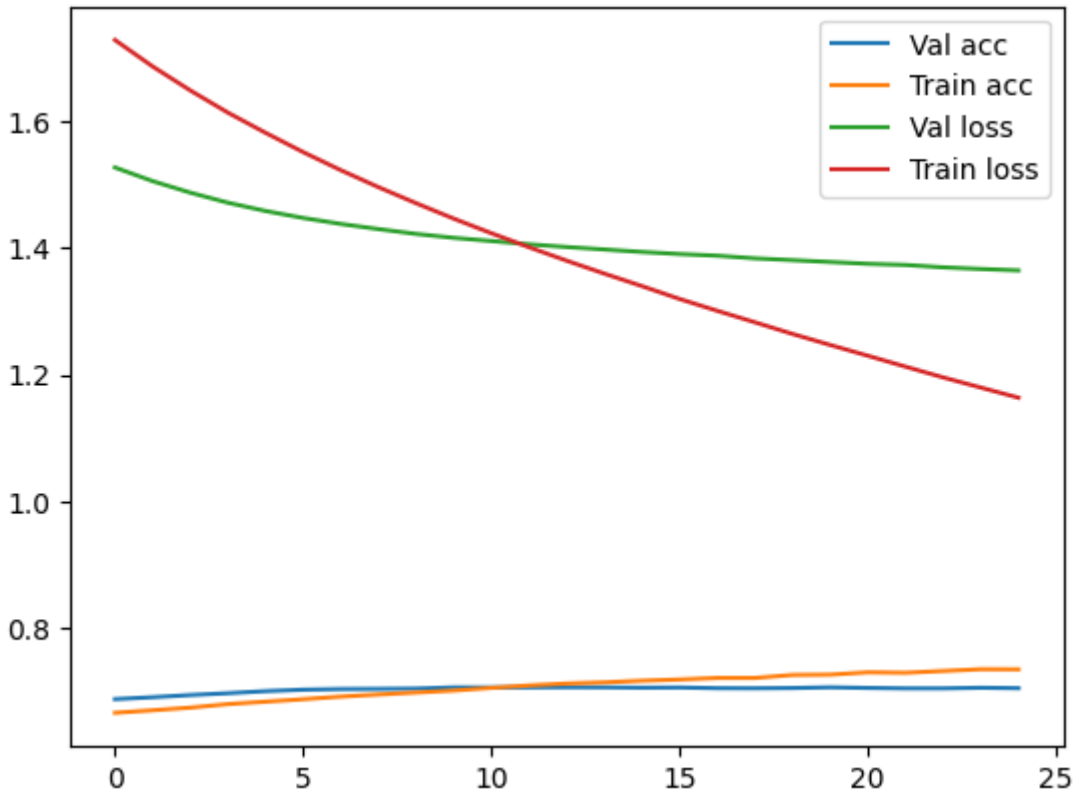



*Matrice de confusion - saliency maps - fine tuning*

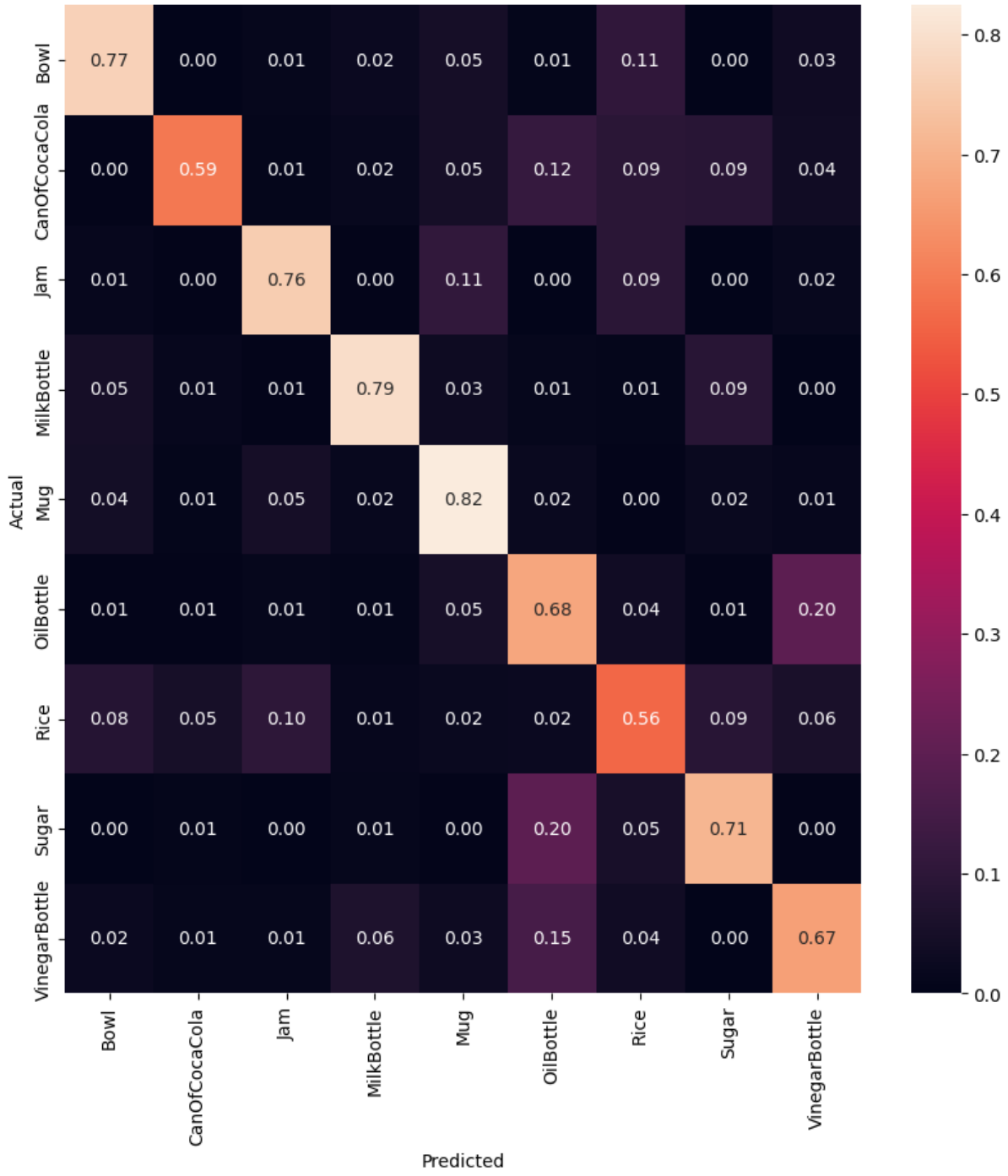
## Incremental learning

Nous explorons deux approches, un fine tuning et un move-to-data. Pour le fine-tuning, nous essayons en entraînant uniquement la couche supérieure (trainable = false) ou en entraînant toutes les couches.

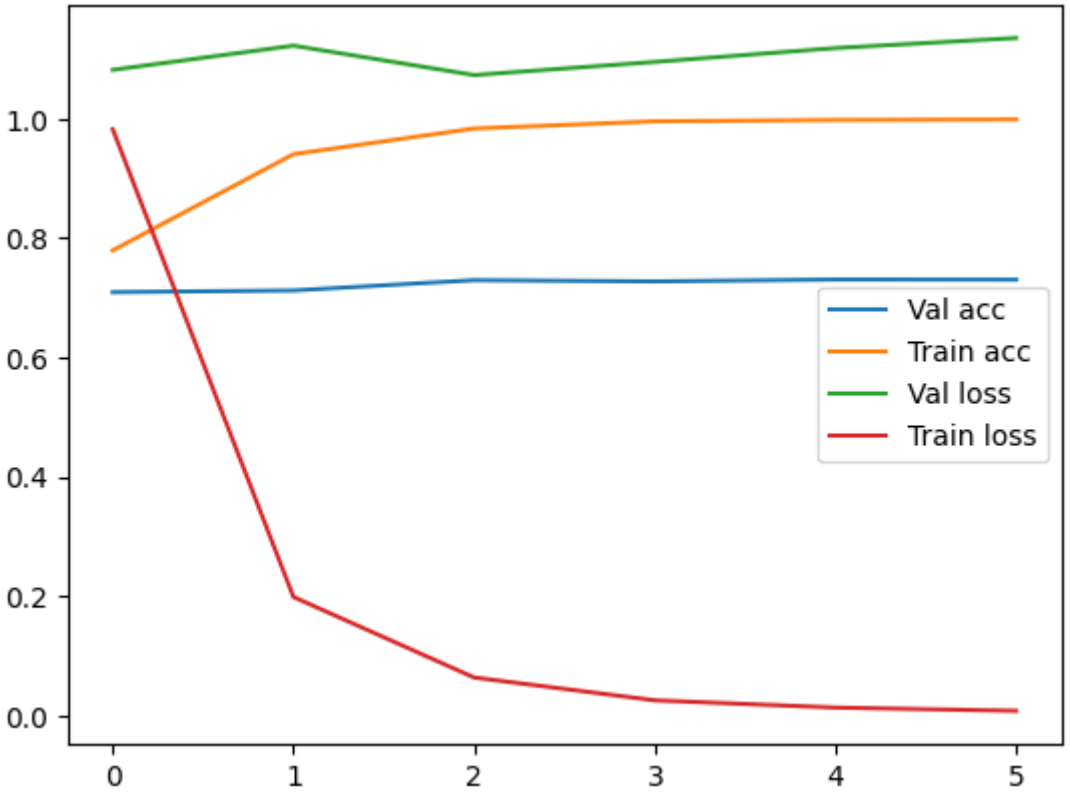
Approche classique: fine-tuning




*Courbe d'entraînement - saliency maps - fine tuning - incremental false*
final Val accuracy: 0.706

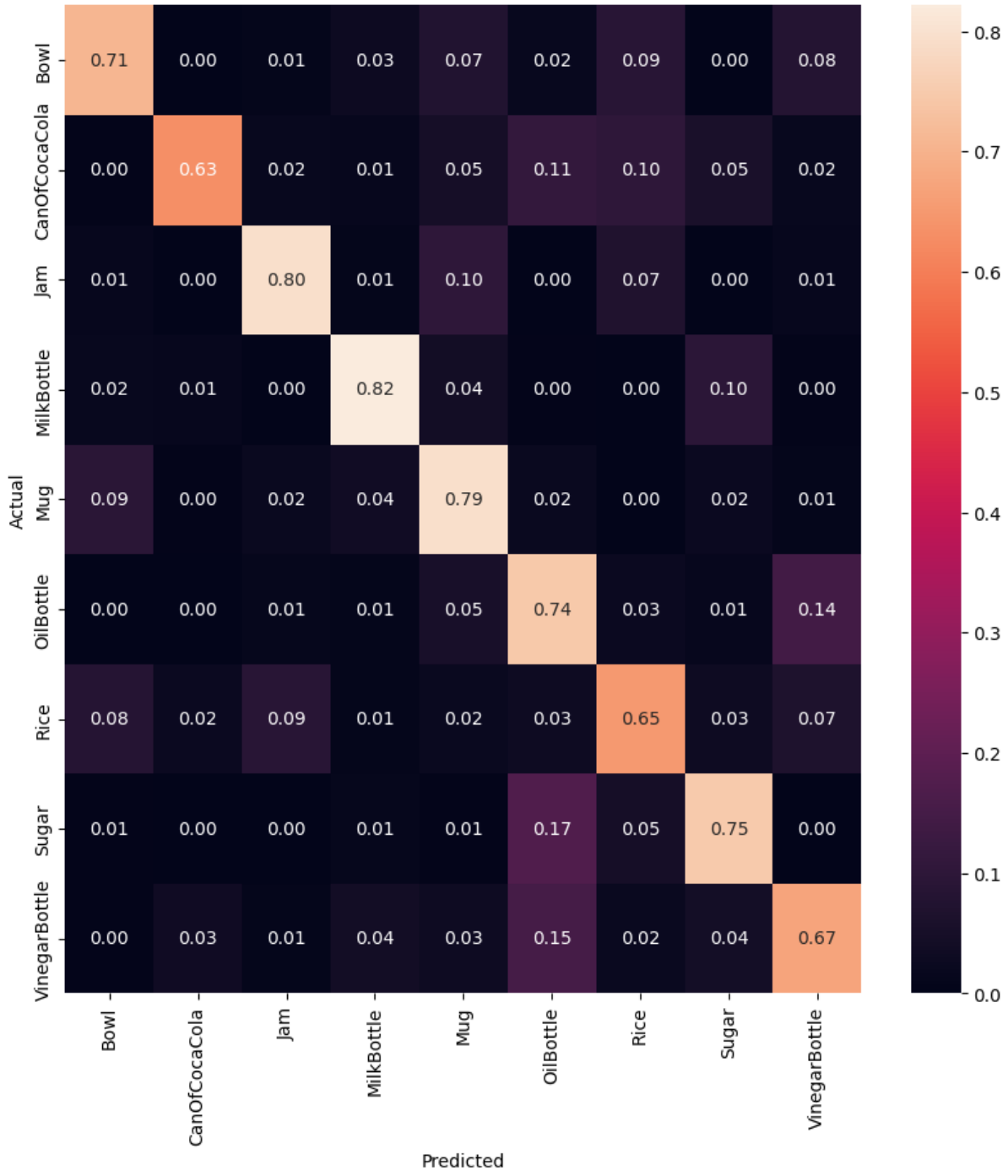



*Matrice de confusion - saliency maps - fine tuning - incremental false*




*Courbe d'entraînement - saliency maps - fine tuning - incremental true*

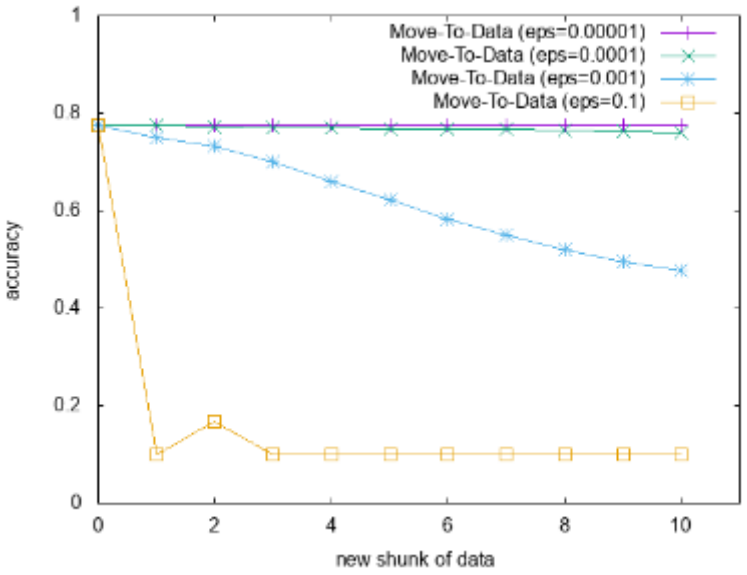
final Val accuracy: 0.73




*Matrice de confusion - saliency maps - fine tuning - incremental true*

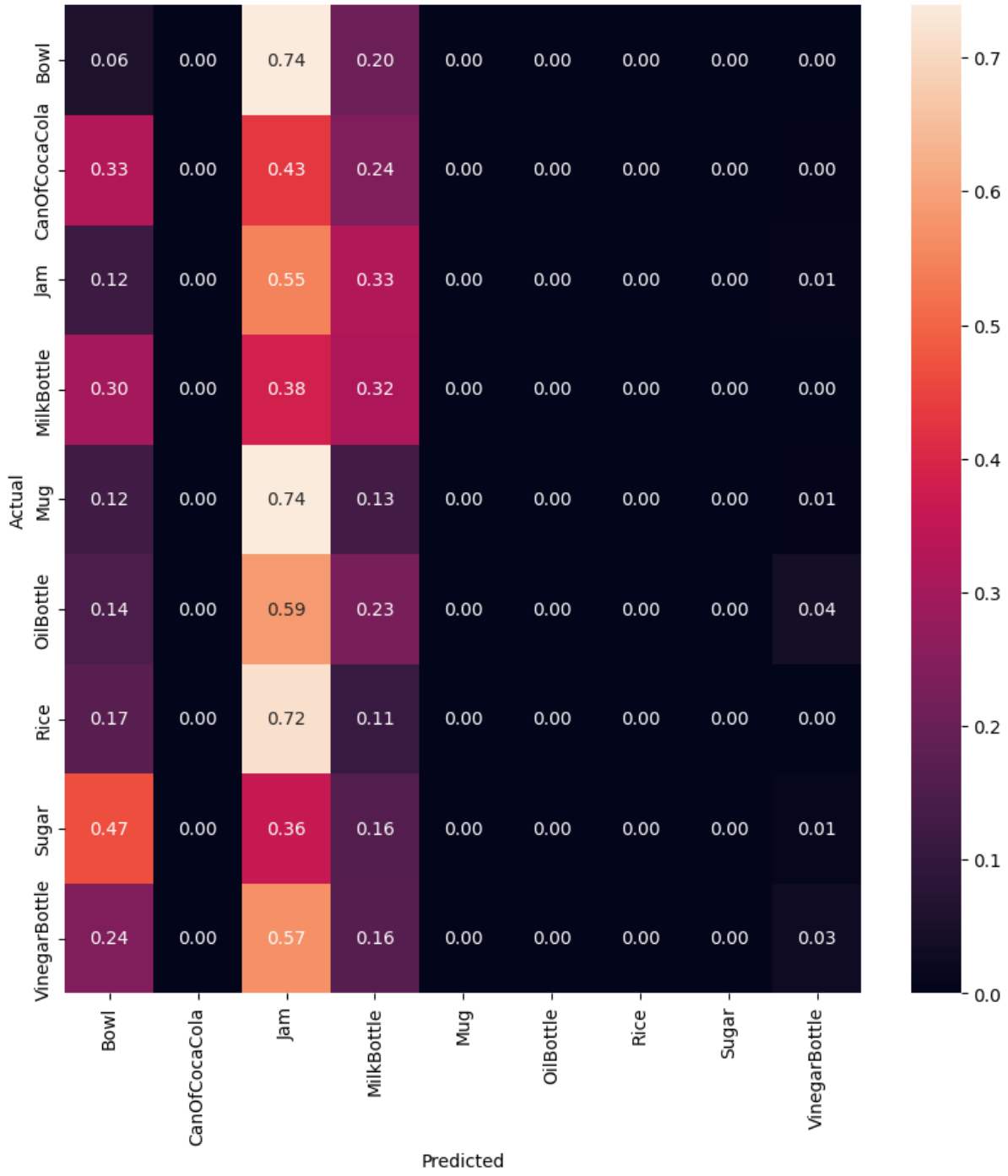
Nous observons de meilleurs résultats en n'ayant pas les poids du modèle EfficientNet freezé. Cela n'est pas plus coûteux car la convergence est plus rapide et le EarlyStopping implémenté permet d'éviter d'over fitté et de faire trop d'epoch.

## Nouvelle approche: Move-To-Data



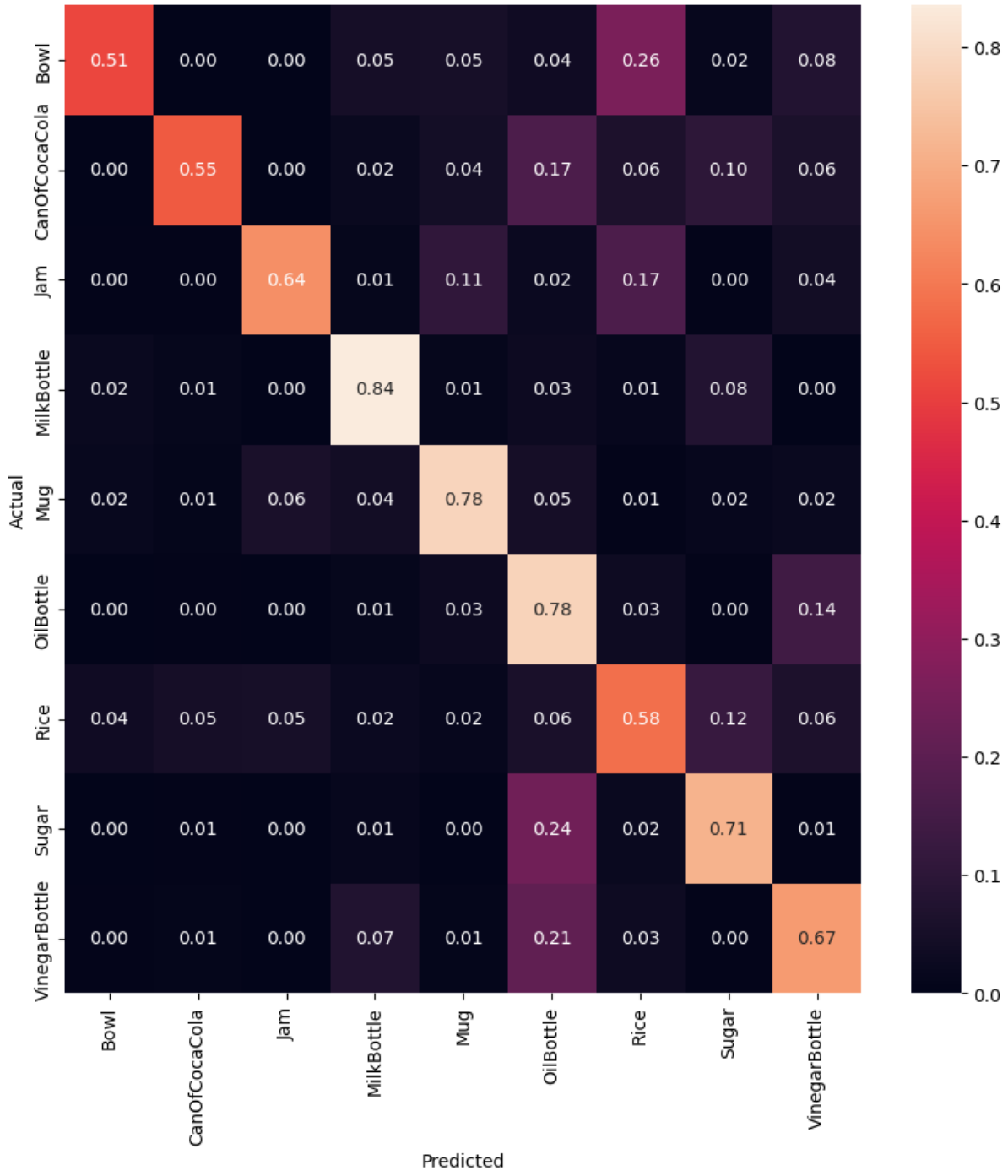

*Graphique issu de votre papier*

Nous avons lancé un entraînement pour quatres valeurs d'epsilon, en cherchant à reproduire les résultats présentés ci-dessus.




*Matrice de confusion - Move-To-Data epsilon = 10e-2*

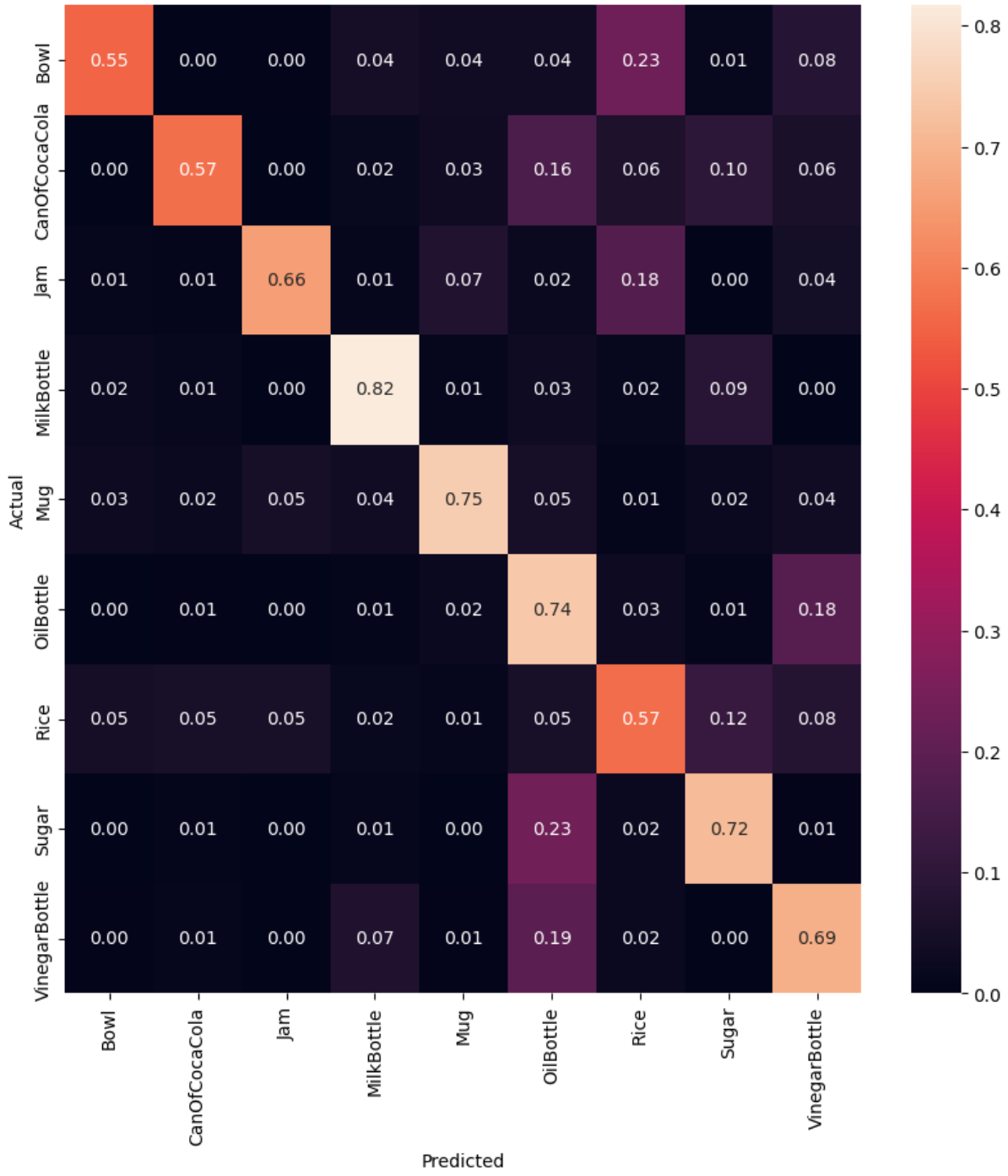
final Val accuracy = 0.092




*Matrice de confusion - Move-To-Data epsilon = 10e-5*

final Val accuracy = 0.685

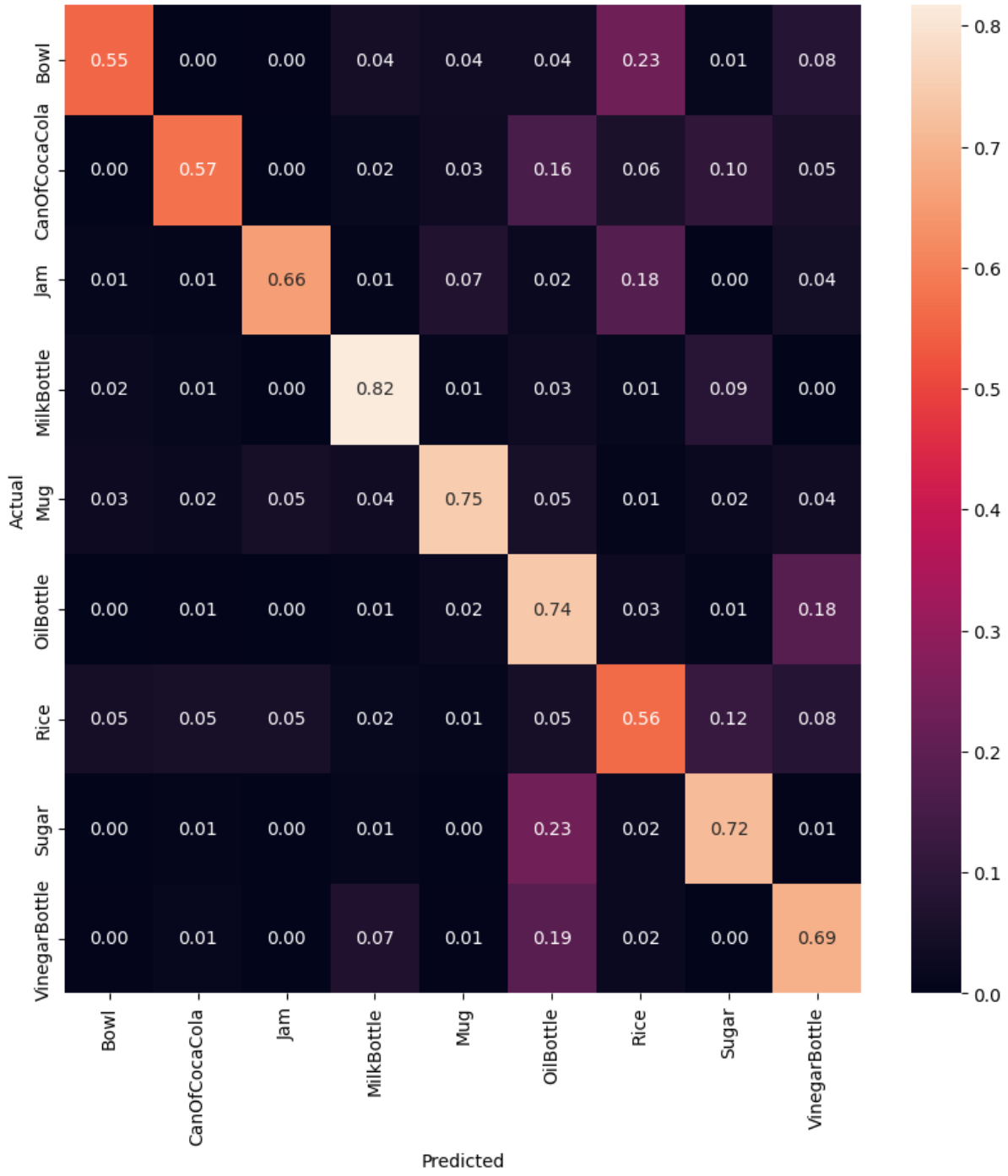
→ amélioration sur les classes les mieux représentées dans le dataset, mais détérioration sur les autres




*Matrice de confusion - Move-To-Data epsilon = 10e-6*

final Val accuracy = 0.684

→ très très légère amélioration (0.57 au lieu de 0.56 pour la classe “Rice” bien prédite)




*Matrice de confusion - Move-To-Data epsilon = 10e-7*

final Val accuracy = 0.683

→ aucune amélioration

Nous n'avons pas réussi à obtenir des résultats concluants pour le move-to-data, mais nous restons dans l'idée du papier: une valeur d'epsilon trop grande cause du "catastrophic forgetting".

## Suivi des entraînements

Le suivi des entraînements a été réalisé grâce à CodeCarbon, qui a regroupé le temps d'entraînement ainsi que les émissions carbone de chaque entraînement réalisé dans le fichier "emissions.csv".

En comparant l'incremental learning par fine-tuning et par move-to-data, nous remarquons que la fine-tuning (avec poids non gelé) est plus rapide (606s) que le move-to-data (epsilon = 0.0001) (841s) mais plus coûteux en émissions (0.0015 contre 0.00084). Il faut aussi faire remarquer que nous entraînons le fine-tuning sur un batch de 16 alors que le move-to-data est réalisé image par image. Également, il tourne sur GPU au lieu de CPU. Si l'on compare au fine-tuning avec les poids gelés, la différence est encore plus notable, car l'entraînement a duré bien plus longtemps que celui avec les poids non gelés (1421s et 0.0033). La comparaison est plus raisonnable car on entraîne uniquement la dernière couche, comme dans le move-to-data.

## Conclusion

Ce projet nous a permis de nous sensibiliser à diverses choses. Tout d'abord, une utilisation concrète de l'analyse de vidéo, ici dans le cadre médical, pour aider les personnes amputées à retrouver les meilleures fonctions motrices possibles. Également d'un point de vue écologique, avec l'importance dans le monde actuel de trouver les méthodes les plus efficaces et moins coûteuses pour la planète.