

Mimir DéTECTeur d'Alzheimer



Nadejda DOROSENCO, Louis ROQUE et Jean-Marc ZHOU

Mimir DéTECTEUR d'Alzheimer	1
I. Description du projet	4
II. Première approche du projet	4
A. Proposition fonctionnelle :	4
B. Proposition technique	6
C. Design de l'Infrastructure sur GCP	7
III. Veille technologique sur l'existant	8
IV. Proposition de l'équipe	9
A. Étudiants	9
B. Motivation professionnelle	9
C. Référent fonctionnel	9
V. Alzheimer, la maladie	10
1. Pour qui?	13
2. Cas d'usage	13
VI. Analyse de nos données	14
VII. Test et résultat sur différents modèles	15
A. RandomForest, Xgboost,VGG16/19	16
B. SVM	16
1. Explications des Métriques et du résultat	16
1. Analyse des Résultats	18
2. Problème avec le SVM	18
C. CNN	18
1. Analyse des résultats partie 1	18
2. Analyse des Résultats partie 2	21
3. Problème avec le CNN	22
D. Resnet50	23
1. Principe du Transfer Learning	23
2. Pourquoi ResNet50 est efficace pour la classification d'IRM ?	23
3. Modélisation	24
VIII. Modèle finaux	27
Construction du Modèle	27
Ajout de Dropout	27
Flattening	27
Batch Normalization	28
Première Couche Dense	28
Seconde Batch Normalization et Activation	28
Couche de Sortie	28
Compilation	28
IX. Modèle global	29
Entraînement	29
Validation	30
Prétraitement des Nouvelles Données de Test	30
Évaluation et Prédictions	31

Création des Cartes de Chaleur Grad-CAM	31
Sauvegarde et Affichage des Cartes de Chaleur	31
Sélection et Visualisation des Images	31
Conclusion	32
X. Déploiement sur le cloud	33
XI. Présentation de l'application MIMIR	34
Structure des Fichiers	34
B. Visuelle du site et utilisation	35
C. Coût du projet	37
XII. Gestion du projet	38
1. Planning	38
2. Répartition de la charge de travail	39
3. Première approche du projet	39
Design de l'Infrastructure sur GCP (1ère approche)	40
4. Difficultés rencontrées	40
5. Ce qu'on pourrait améliorer	40

I. Description du projet

Un site web nommé Mimir qui permet d'identifier le stade d'Alzheimer d'un patient à l'aide d'une IRM de son cerveau. Il sera une aide à la décision pour les médecins qui vont analyser les images pour délivrer un diagnostic rapide sur les différents stades, ce qui peut être déterminant pour les patients atteints de légers problèmes cognitifs qui peuvent s'aggraver en quelques années jusqu'à devenir de la démence.

II. Première approche du projet

A. Proposition fonctionnelle :

L'utilisateur pourra envoyer un fichier ou une image d'une IRM et le site web lui dira s'il a l'alzheimer ou non et son stade.

L'application affichera après le téléchargement de l'image une interface avec 3 modèles et un récapitulatif de ses performances afin de l'aiguiller au mieux dans son évaluation du patient. Ensuite après prédiction et analyse de l'image par notre application, nous afficherons un pourcentage (la probabilité de prédiction de la classe).

Cette courte analyse de performances permettra au médecin de déterminer par lui-même, grâce à son expertise métier, la méthode de prédiction et les risques associés au modèle utilisé. Nous essayerons de maximiser la valeur de certaines métriques pour les modèles (Recall, F1-score, Précision) pour que le médecin puisse avoir le choix entre les modèles qui ont une meilleure précision sur les erreurs de type I (Faux positif) ou les erreurs de type II (Vrai négatif).

Les erreurs possibles peuvent être graves car des ordonnances peuvent affecter plus ou moins lourdement la vie quotidienne d'un patient selon les différents stades de la maladie, et dans le domaine médical c'est une erreur non négligeable.

Les différents stades étudiés dans notre database :

- Démence légère : légers oublis tels que des mots ou des emplacements d'objets.

- Démence très légère : Les individus ont des oubliés qu' un individu non atteint pourrait avoir tels que le fait d'oubliés si on a fermé la porte.
- Démence modérée : les symptômes de la démence sont graves pour interférer avec les activités quotidiennes, oublie de personne, de lieu ...
- Pas de démence : des individus qui ne présentent pas de signes de démence.



Nous avons aussi mis en place une description de comment utiliser notre application avec des statistiques et une aide à la compréhension des métriques utilisées.

Modèle Global

Comment lire ?

Précision (99%) : Indique la proportion des prédictions correctes parmi toutes les prédictions faites par le modèle. Une précision de 99% signifie que 99% des prédictions du modèle sont correctes.

Rappel (100%) : Mesure la capacité du modèle à identifier toutes les instances positives. Un rappel de 100% signifie que le modèle a correctement identifié toutes les instances positives dans les données.

F1-score (99%) : La moyenne harmonique de la précision et du rappel. Un F1-score de 99% indique un bon équilibre entre précision et rappel, signifiant que le modèle est à la fois précis et complet dans ses prédictions.

Matrice de Confusion

La matrice de confusion présente les résultats de classification sous forme de tableau, où :

Les lignes représentent les classes réelles (vérités terrain).

Les colonnes représentent les classes prédites par le modèle.

Précision : 99%

Rappel : 100%

F1-score : 99%

	Démence légère	0	0	0
Démence modérée	0	1	0	0
Pas de démence	0.0015625	0	0.992188	0.00625
Démence très légère	0.00223214	0	0.00892857	0.988839
	Démence légère	Démence modérée	Pas de démence	Démence très légère

B. Proposition technique

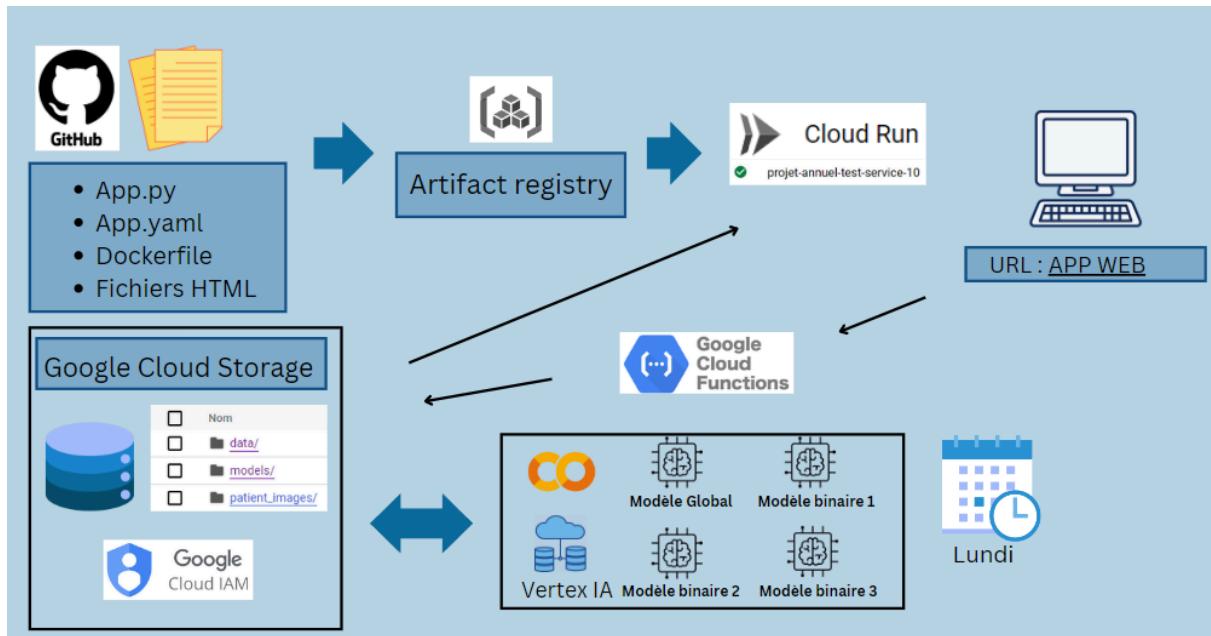
Pour le stockage des données nous utilisons Google Cloud Storage pour stocker les images d'IRM dans un conteneur. Les données seront mises à jour automatiquement avec les autres images entrées par l'utilisateur à la fin du processus.

Nous ferons ensuite l'intégration continue des données via Cloud Build et Colab Enterprise pour automatiser les tests et le déploiement du code. Pour entraîner les modèles nous utilisons AI Platform Nous aurons besoin d'entraîner 4 modèles qui sont des classifications binaires qui utilisent ResNet50 et le modèle global qui prédit sur les 4 classes. Les prédictions et la labellisation se feront ensuite, la prédiction se fera avec le Cloud Run qui permet de servir les prédictions via une API. Le Cloud Run permet de déployer l'application sur une adresse web.

Pour construire l'application web nous utiliserons flask ainsi que Cloud Run ou App Engine pour fournir une interface utilisateur pour que l'utilisateur puisse télécharger les images IRM et afficher les résultats selon le modèle choisi.

C. Design de l'Infrastructure sur GCP

1. **Stockage des données** : Utilisation de Google Cloud Storage et d'un Bucket pour stocker les images d'IRM. (protégé par une clé KMS gérée par le client)
2. **Intégration continue des données** : Mise en place d'enregistrement d'images avec Cloud Run pour placer les images dans notre Bucket.
3. **Entraînement du modèle** : Utilisation de Colab Enterprise pour entraîner les modèles ResNet50 tous les lundis.
4. **Prédiction/Labellisation** : Utilisation des modèles entraînés et sauvegardés dans le bucket par Colab Enterprise et analyse de l'image via Cloud Run .
5. **Monitoring** : Analyses pour surveiller la performance de l'application et l'utilisation directement sur GCP.
6. **Déploiement de l'application** : Mise en place de l'application web avec Cloud Run pour fournir une interface utilisateur pour télécharger les images IRM et afficher les résultats de prédiction.



Nos données :

Un dataset d'images IRM correspondant aux 4 stades d'alzheimer :

- Pas de démence
- Démence très légère
- Démence légère
- Démence modérée

Source : Kaggle, base original OASIS-3

<https://www.kaggle.com/datasets/ninadaithal/imagesoasis/data>

<https://www.kaggle.com/datasets/tourist55/alzheimers-dataset-4-class-of-images>

III. Veille technologique sur l'existant

Overcome : application mobile qui collecte des données sur les habitudes de vie, symptômes, comportements des utilisateurs. Avec un questionnaire et des jeux cognitifs. Ces données sont analysées par un algorithme d'intelligence artificielle pour détecter les signes précurseurs de la maladie d'Alzheimer.

App'zheimer : via des questionnaires

Lili Smart : outils qui aident les personnes atteintes d'Alzheimer à prendre leur médicament grâce à une montre connectée qui envoie des signaux. De plus, elle informe les proches de la situation de vie de la personne atteinte d'Alzheimer si elle mange bien ou non.

Un article de 2022 décrit que des chercheurs britanniques développent une intelligence artificielle pour détecter avec précision la maladie :

<https://www.bonjournsenior.fr/actualites/diagnostiquer-alzheimer-grace-a-une-irm-cest-desormais-possible>

IV. Proposition de l'équipe

A. Étudiants

- Louis ROQUE
- Jean-Marc ZHOU
- Nadejda DOROSENCO

B. Motivation professionnelle

De nombreuses entreprises dans le domaine de la santé et du médical utilisent de l'IA et ce projet pourrait les intéresser.

Nous avons également un intérêt pour le traitement d'images.

Aidez les personnes atteintes de cette maladie.

C. Référent fonctionnel

- **Gilyana Borlikova**

Data Scientist senior, Ancienne chercheuse dans le Biomolecular & Biomedical

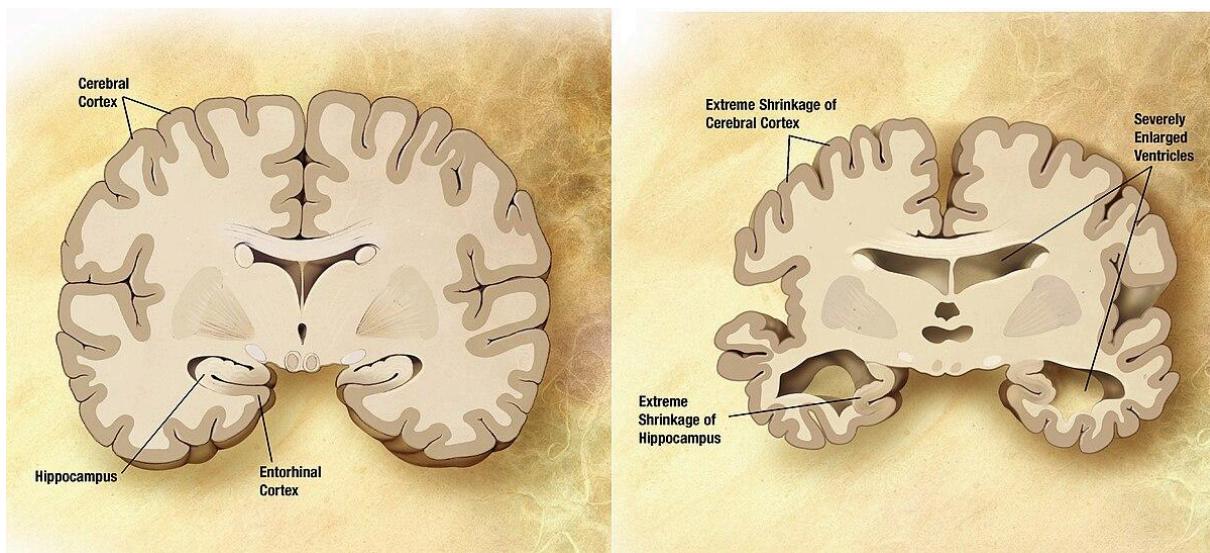
V. Alzheimer, la maladie

C'est une maladie qui touche plusieurs parties du cerveau, affectant certaines fonctionnalités essentielles suite à une détérioration des connexions neuronales :

- Mémoire
- Jugement
- Réflexion
- Langage
- Résolution de problème
- Personnalité
- Mouvement

Pour identifier la maladie par la présence de protéines, nous regardons s'il y a présence de plaques entre les neurones provoquées par une accumulation de la **protéine bêta-amyloïde** et de d'enchevêtrement neurofibrillaires provoquées par la **protéine Tau**.

La présence de ces protéines entraîne progressivement la dégradation des connexions neuronales sur les connexions neuronales voisines.



Ici nous avons la comparaison d'un cerveau d'un patient âgé à gauche et le cerveau d'un patient atteint d'une maladie d'Alzheimer .

Source : https://fr.wikipedia.org/wiki/Maladie_d%27Alzheimer

Les différents stades qu'on peut repérer dans l'évolution de la maladie :

- Le stade asymptomatique : nous notons la présence physiologique des protéines concernées mais le patient n'est pas touché par des symptômes, il n'y a pas de traitement et on ne déclare pas la maladie
- Le stade prodromal : présence de difficultés de mémoire, généralement isolées, ou parfois associées à une désorientation dans l'espace ou à un manque du mot. Les biomarqueurs bêta-amyloïde et Tau permettent de rattacher formellement les troubles
- Le stade de démence : lorsque les troubles cognitifs et comportementaux ont un impact sur la vie quotidienne. Ils ne sont plus autonomes. Dans ce stade, on distingue trois étapes.
 - démence légère : troubles de mémoire récurrents avec oubli d'événements récents, tendance à poser plusieurs fois la même question ou à se répéter. Troubles orientation temporelle et spatiale, attention, raisonnement, difficulté à trouver les mots. -> impacte l'utilisation de l'ordinateur ou d'un téléphone portable
 - démence modérée : les fonctions cognitives sont plus impactées, provoquant des troubles du langage et de la compréhension. Un accompagnement devient nécessaire
 - démence sévère : le patient perd toute autonomie dans la vie quotidienne, ne peut plus s'occuper de sa toilette ni des repas. L'accompagnement devient une aide omniprésente pour assister le patient.

(Source : <https://alzheimer-recherche.org/>)

Dans le dataset Kaggle nous traitons le stade asymptomatique, prodromal, démence légère et démence modérée.

En France

La maladie d'Alzheimer est la plus fréquente des maladies neurodégénératives avec 1 million de malades. Chaque année **225 000 nouveaux cas sont recensés**. En comptant les proches aidants, **3 millions de personnes sont directement concernées** par la maladie d'Alzheimer (malades et proches aidants). Mais si la maladie frappe le plus souvent des personnes âgées (près de **15% des plus de 80 ans**), elle peut aussi survenir beaucoup plus tôt. On estime aujourd'hui en France à

30 000 le nombre de patients de moins de 60 ans atteints de la maladie d'Alzheimer.

En Europe

9,7 millions de personnes souffrent actuellement de démence en Europe (source Alzheimer Europe). Pour rappel, la démence est un syndrome provoqué par des troubles cognitifs majeurs conduisant à la perte d'autonomie. La maladie d'Alzheimer représente environ 70% de ces cas soit **6,8 millions de personnes**. 2 malades sur 3 sont des femmes. Le nombre de malades devrait doubler d'ici 2050.





Dans le Monde

Plus de **35,6 millions de personnes** sont touchées par la maladie d'Alzheimer. Chaque année, on dénombre **7,7 millions de nouveaux cas**. Selon les prévisions de l'Organisation Mondiale de la Santé (OMS), le nombre de malades devrait presque doubler tous les 20 ans.

1. Pour qui?

Notre logiciel est destiné aux personnels médicaux tels que les médecins, infirmiers et autres. Il gagnera du temps sur l'identification de la maladie d'Alzheimer en plaçant une photo ou fichier(comportant plusieurs photo) qui indiquera le stade d'Alzheimer et les zones qui indiquent que le patient a l'Alzheimer.

2. Cas d'usage

Nous avons un patient qui effectue une IRM pour l'Alzheimer.

Le neurologue déposera l'IRM sur l'application qui détectera si le patient est atteint d'Alzheimer et de son stade, les résultats pour le patient seront données en temps réel. Le neurologue aura alors un résultat de l'irm avec les différents stade et les zones qui permettent d'affirmer que le patient est atteint d'Alzheimer.

VI. Analyse de nos données

Nous avons des données provenant de la base OASIS, une base très connue pour l'accès à des données médicales. Nous avons fait la traductions de dataset car la base fournis étais en anglais

- NonDemented = Pas de démence
- VeryMildDemented = Démence très légère
- MildDemented = Démence légère
- ModerateDemented = Démence modérée

Nous avons récupéré un échantillon sur Kaggle. Nos images sont des vues de face (voir exemple ci-dessous).



Leurs tailles sont de 176 x 208 pixels. Elles sont stockées dans des dossiers "train" et "test", avec des sous-dossiers correspondant aux différents stades de la maladie d'Alzheimer. Nous changeons la taille des images en 176 * 176 pixels pour un résultat plus optimal.

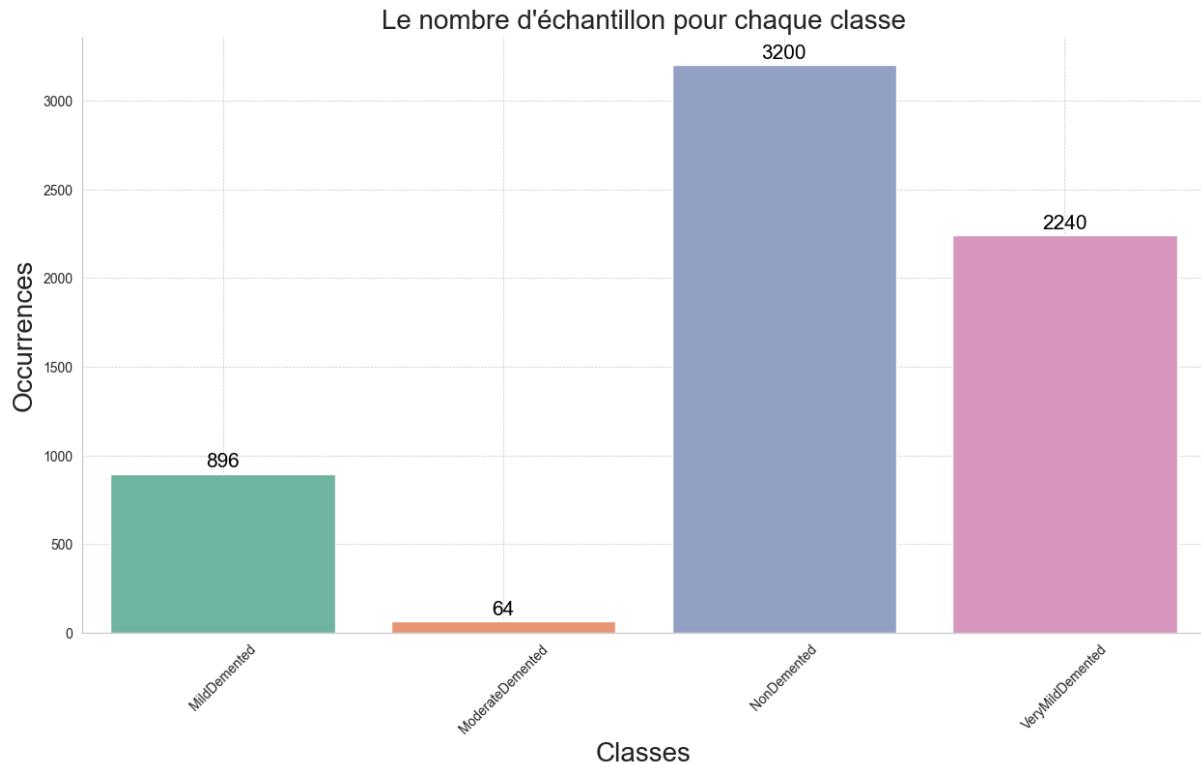
pour la partie train :

- Pas de démence(2560 images)
- Démence très légère(1792 images)
- Démence légère(717 images)
- Démence modérée ((52 images)

pour la partie test:

- Pas de démence(640 images)
- Démence très légère(448 images)
- Démence légère(179 images)
- Démence modérée (12 images)

1. Graphique du nombre d'échantillon test et train



Nous avons remarqué un déséquilibre dans les classes. Par conséquent, nous avons décidé de faire de la data augmentation pour rééquilibrer les classes et améliorer les performances de nos modèles. Nous allons suréchantillonner nos classes jusqu'à ce qu'elle atteigne le volume de la classe dominante.

De plus, lors de l'entraînement nous faisons en sorte que l'image soit toujours de la même taille au cas où il y aurait des images plus grandes ou plus petites qui entrent dans notre jeu de données.

VII. Test et résultat sur différents modèles

Pour obtenir les meilleures performances pour notre outil de détection de la maladie d'Alzheimer, nous avons décidé d'expérimenter plusieurs modèles. Dans un premier temps, nous avons recherché les modèles qui pourraient offrir des prédictions et une précision assez bonnes. Voici la liste des modèles que nous avons testés :

- ResNet50

- CNN
- SVM
- VGG16/19
- RandomForest
- Xgboost

A. RandomForest, Xgboost,VGG16/19

Nous avons testé Random Forest, XGBoost et VGG16/19 pour évaluer l'accuracy que nos modèles pouvaient atteindre. Les résultats varient entre 50% et 60% d'accuracy et entre 40% et 55% de précision pour des modèles globaux. Nous avons rapidement décidé d'abandonner ces modèles car nous ne les trouvions pas utiles pour la suite. Nous nous sommes donc concentrés sur les modèles CNN, SVM et ResNet50.

B. SVM

le svm était notre meilleur modèles au début ,on avait des résultats plutôt bonne voir le tableau ci-dessou:

	precision	recall	f1-score	support
NonDemented	0.75	0.78	0.76	640
VeryMildDemented	0.60	0.72	0.65	448
MildDemented	0.92	0.37	0.53	179
ModerateDemented	1.00	0.33	0.50	12
accuracy			0.70	1279
macro avg	0.82	0.55	0.61	1279
weighted avg	0.72	0.70	0.69	1279

1. Explications des Métriques et du résultat

- **Précision (Précision)** : La précision est le ratio des prédictions positives correctes par rapport au nombre total de prédictions positives. Une précision élevée indique que peu de faux positifs ont été prédits.

- Par exemple, pour la classe "NonDemented", la précision est de 0.75, ce qui signifie que 75% des prédictions de cette classe sont correctes.
- **Rappel (Recall)** : Le rappel est le ratio des prédictions positives correctes par rapport au nombre total de vrais positifs dans les données. Un rappel élevé signifie que peu de vrais positifs ont été manqués.
 - Pour la classe "Démence très légère", le rappel est de 0.72, ce qui signifie que 72% des instances de cette classe ont été correctement identifiées par le modèle.
- **F1-Score** : Le F1-score est la moyenne harmonique de la précision et du rappel. Il fournit un équilibre entre les deux métriques, surtout lorsqu'il y a un déséquilibre entre les classes.
 - Pour la classe "Démence légère", le F1-score est de 0.53, reflétant le compromis entre précision (0.92) et rappel (0.37).
- **Support** : Le support est le nombre d'instances réelles de chaque classe dans les données test.
 - Par exemple, le support pour la classe "NonDemented" est de 640.
- **Accuracy (Exactitude)** : L'accuracy est la proportion de toutes les prédictions correctes parmi le nombre total d'instances testées. Dans notre cas, notre modèle atteint une accuracy de 0.70 (70%).
- **Macro Average (Moyenne Macro)** : La moyenne macro est la moyenne des métriques (précision, rappel, F1-score) calculées indépendamment pour chaque classe. Elle traite toutes les classes de manière égale, sans tenir compte du support.
 - La macro moyenne de précision est de 0.82, indiquant une performance équilibrée à travers toutes les classes.
- **Weighted Average (Moyenne Pondérée)** : La moyenne pondérée est la moyenne des métriques calculées en tenant compte du support de chaque classe. Elle donne plus de poids aux classes avec plus d'instances.
 - La moyenne pondérée de précision est de 0.72, reflétant la performance globale du modèle en tenant compte du déséquilibre des classes.

Nous avons décidé d'utiliser toutes ces métriques dans notre analyse, suivant les conseils de notre référente technique.

1. Analyse des Résultats

Ces résultats montrent que notre modèle SVM atteint une précision globale de 70%, avec des performances variables selon les différentes classes de la maladie. Les classes "Démence légère" et "Démence modérée" sont moins bien représentées, ce qui indique un besoin potentiel d'amélioration de l'équilibrage des classes et des techniques de modélisation. De ce fait nous avons pensé à faire de la data augmentation qui nous a permis d'améliorer la précision et le rappel de ces classes minoritaires. Nous avons également fait un grid search pour trouver les meilleurs paramètres pour le modèles.

2. Problème avec le SVM

Le problème du SVM est qu'il prenait beaucoup de temps à s'entraîner, ce qui fait que nous ne l'avons pas retenu car le temps était trop conséquent. Nous nous sommes donc orientés vers les modèles CNN et ResNet50.

C. CNN

Le CNN était plus rapide que le SVM donc nous nous sommes intéressés à ce modèle. Nous avons donc fait plein de tests sur le CNN en modifiant les couches, les variables pour essayer d'avoir le meilleur résultat possible.

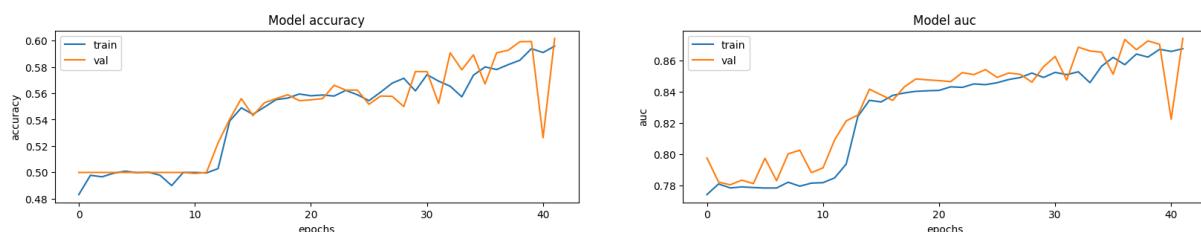
1. Analyse des résultats partie 1

	precision	recall	f1-score	support
MildDemented	0.18	0.13	0.15	179
ModerateDemented	0.00	0.00	0.00	12
NonDemented	0.50	0.35	0.41	640
VeryMildDemented	0.38	0.58	0.46	448
accuracy			0.40	1279
macro avg	0.26	0.27	0.26	1279
weighted avg	0.41	0.40	0.39	1279

Dans un premier temps le CNN avait des résultats plutôt médiocre.

Pour "Démence légère", la précision est de 0.18, le rappel de 0.13, et le F1-score de 0.15, avec un support de 179. Pour "Démence modérée", toutes les métriques sont à 0.00, avec un support de 12, indiquant probablement une mauvaise performance ou un déséquilibre des classes. Pour "Pas de démence", la précision est de 0.50, le rappel de 0.35 et le F1-score de 0.41, avec un support de 640. Enfin, pour "Démence très légère", la précision est de 0.38, le rappel de 0.58 et le F1-score de 0.46, avec un support de 448.

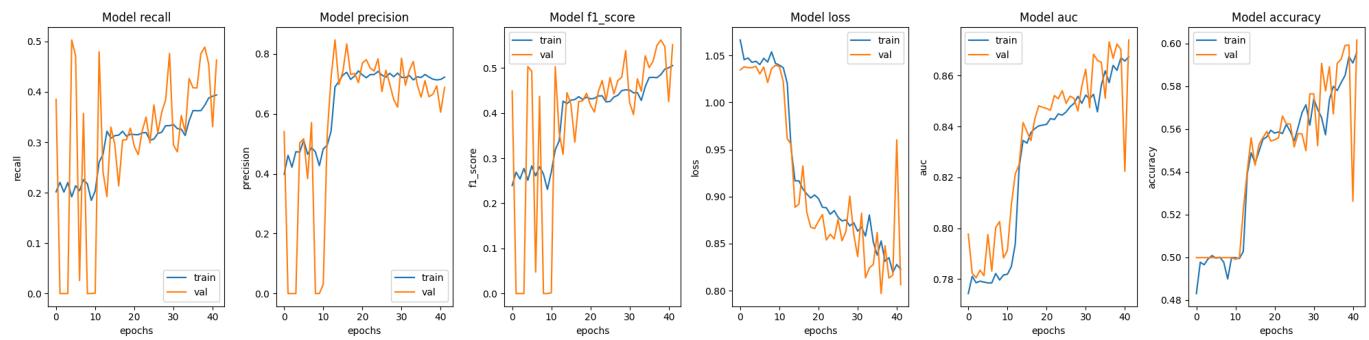
Le tableau inclut aussi des moyennes globales : une précision macro moyenne de 0.26, un rappel de 0.27 et un F1-score de 0.26. La moyenne pondérée de ces métriques est respectivement de 0.41, 0.40 et 0.39. L'exactitude globale du modèle est de 0.40 sur un total de 1279 échantillons. Ces résultats suggèrent que le modèle a une performance globale limitée, avec des difficultés particulières à bien classer les catégories "Démence légère" et "Démence modérée".



Dans le graphique on remarque que la précision de l'ensemble d'entraînement commence très bas comparé au SVM , on est autour de 0.48, et augmente progressivement pour atteindre environ 0.58 à la fin des 40 époques.

La précision de l'ensemble de validation suit une tendance similaire, bien que la courbe validation soit moins alignée avec la courbe train, indiquant des fluctuations et des ajustements qui ont lieu pendant l'entraînement.

Avec l'ensemble d'entraînement, l'AUC commence autour de 0.78 et augmente jusqu'à environ 0.86. L'AUC montre une amélioration continue, avec quelques fluctuations, atteignant finalement environ 0.86.



Nous avons analysé les autres variables tels que le recall, precision, F1-score et la lost.

Le rappel (recall) de l'ensemble d'entraînement augmente progressivement avec quelques fluctuations au niveau de la courbe, atteignant environ 0.45 à la fin des 40 époques. Le rappel de l'ensemble de validation montre de fortes fluctuations mais une tendance à la hausse, indiquant une amélioration mais avec une variabilité importante.

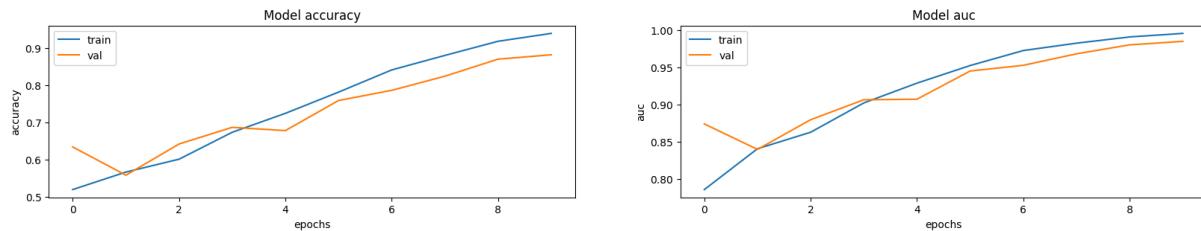
La précision de l'ensemble d'entraînement augmente légèrement mais reste stable autour de 0.45-0.5. La précision de l'ensemble de validation suit une tendance légèrement ascendante, atteignant environ 0.5 à la fin.

Le score F1 (f1_score) pour l'ensemble d'entraînement montre une augmentation qui augmente, atteignant environ 0.5 à la fin. Le score F1 de l'ensemble de validation fluctue fortement mais suit une tendance ascendante, indiquant une amélioration générale de la balance entre précision et rappel.

La perte (loss) de l'ensemble d'entraînement diminue de manière régulière, atteignant environ 0.8 à la fin des 40 époques, ce qui est un bon signe de convergence.

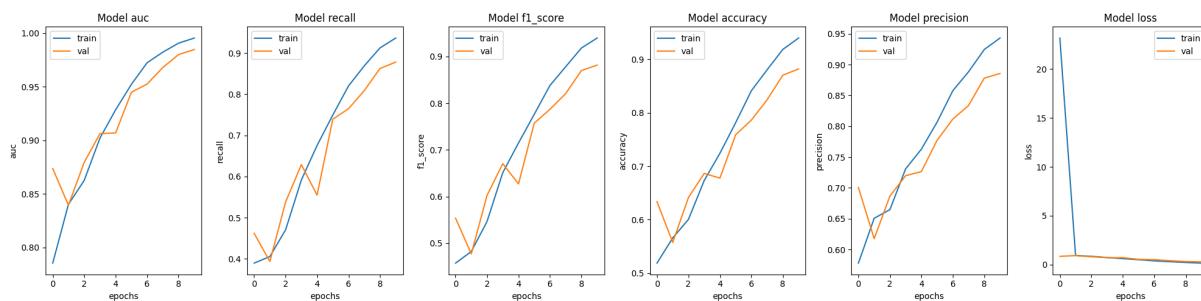
2. Analyse des Résultats partie 2

Nous avons donc modifié notre CNN pour avoir de meilleurs résultats. Pour cela nous avons fait de la data augmentation pour équilibrer les classes, rajouter des couches et une structure plus simple pour le modèle CNN.



Pour l'Accuracy, la précision de l'ensemble d'entraînement augmente progressivement et atteint environ 0.9 à la fin de l'entraînement. La précision de l'ensemble de validation suit une tendance similaire, bien qu'elle soit légèrement inférieure à celle de l'entraînement, atteignant environ 0.8 à la fin.

Pour L'AUC nous avons pour l'ensemble d'entraînement qui augmente de manière régulière pour atteindre presque 1.0. L'AUC pour l'ensemble de validation est très proche de celle de l'entraînement, indiquant une bonne performance du modèle.



Le rappel augmente progressivement pour atteindre environ 0.9 à la fin des 10 époques. Pour la validation nous remarquons que le rappel suit une tendance similaire mais avec des petites fluctuations , atteignant environ 0.85 à la fin. Cela montre que le modèle capture bien les vrais positifs sur l'ensemble de validation.

Le score F1 augmente de manière régulière, atteignant environ 0.95. Le score F1 validation suit également une tendance ascendante, atteignant environ 0.85, avec quelques fluctuations. Cela indique une bonne balance entre précision et rappel..

La précision augmente régulièrement pour atteindre presque 1.0. La précision suit une tendance similaire mais reste légèrement inférieure, atteignant environ 0.8. La courbe montre une augmentation stable avec des fluctuations mineures.

La perte diminue drastiquement dès la première époque, puis continue de diminuer lentement pour atteindre une valeur très basse. La perte diminue également mais de manière légère. Cela indique que le modèle est efficace pour réduire l'erreur mais que la perte sur le jeu de validation se stabilise à un certain point.

Pour résumer, les métriques montrent une amélioration qui augmente pour l'entraînement et la validation dans notre modèle CNN. Les courbes de validation sont proches des courbes d'entraînement, indiquant une bonne capacité sans surajustement significatif. Bien que les courbes de validation ont de légères fluctuations, la tendance est une amélioration des performances, ce qui est un signe positif. La rapide diminution de la perte dans les premières époques indique un apprentissage rapide du modèle, mais la stabilisation de la perte de validation suggère qu'il pourrait y avoir une limite.

3. Problème avec le CNN

Après une analyse plus approfondie des performances de notre modèle de réseau de neurones convolutifs (CNN) avec des tests et d'autres analyses , il nous semble évident que le modèle présente des signes d'overfitting. Plusieurs indicateurs nous donnent ce problème.

Nous avons remarqué que les métriques de performance telles que la précision (accuracy) et l'aire sous la courbe (AUC) montrent une amélioration continue sur l'ensemble d'entraînement, ces mêmes métriques sur l'ensemble de validation stagnent ou fluctuent de manière significative après un certain nombre d'époques. Cela suggère que le modèle s'adapte trop aux données d'entraînement.

De plus, la perte (loss) de validation ne diminue pas aussi rapidement que celle de l'entraînement et tend à se stabiliser voire à augmenter légèrement après plusieurs époques, tandis que la perte d'entraînement continue de diminuer. Ce comportement est un indicateur classique d'overfitting, où le modèle devient très performant sur l'ensemble d'entraînement au détriment de sa performance sur des données non vues.

En conclusion, bien que notre modèle CNN à une capacité à s'ajuster aux données d'entraînement, Cependant il échoue sur l'ensemble de validation et de test. Nous avons donc décidé de passer au Resnet qui est notre meilleur modèle sur la rapidité et l'efficacité.

D. Resnet50

ResNet50, ou réseau résiduel avec 50 couches, est un modèle avancé de CNN. Créé par He et ses collègues en 2015, ce modèle est connu pour pouvoir entraîner des réseaux très profonds sans souffrir du problème de gradient vanishing ou exploding. ResNet50 utilise des "skip connections" (connexions de saut) qui permettent de sauter des couches dans le réseau, ce qui aide à maintenir la propagation des gradients et rend l'apprentissage plus efficace.

1. Principe du Transfer Learning

Le transfert d'apprentissage consiste à prendre un modèle pré-entraîné sur une grande base de données (comme ImageNet) et à le réutiliser pour une nouvelle tâche. Pour la classification d'IRM, on utilise un ResNet50 déjà entraîné et on réentraîne les dernières couches sur notre jeu de données spécifique d'IRM. Cela permet de profiter des caractéristiques déjà apprises par ResNet50, ce qui est particulièrement utile si nos données sont limitées.

2. Pourquoi ResNet50 est efficace pour la classification d'IRM ?

ResNet50 est particulièrement efficace pour la classification d'IRM en raison de sa profondeur et de sa complexité. Avec ses 50 couches, il peut apprendre des caractéristiques très détaillées et complexes des images. C'est crucial pour l'analyse d'IRM, où les subtilités et les petits détails peuvent être déterminants pour la classification. Les CNN traditionnels, qui sont souvent moins profonds, ne capturent pas ces mêmes niveaux de détail et de complexité, ce qui peut limiter leur performance dans des tâches aussi exigeantes.

Une des innovations clés de ResNet50 est l'utilisation de skip connections. Ces connexions permettent de sauter une ou plusieurs couches dans le réseau, facilitant ainsi la propagation des gradients et évitant les problèmes de dégradation qui peuvent survenir dans les réseaux

très profonds. Grâce à ces skip connections, ResNet50 maintient une performance élevée même avec un grand nombre de couches, contrairement aux architectures sans ces connexions qui peuvent souffrir de la perte de gradient, rendant l'apprentissage moins efficace.

Le transfer learning joue également un rôle crucial dans l'efficacité de ResNet50 pour la classification d'IRM. En utilisant un modèle ResNet50 pré-entraîné sur une vaste base de données comme ImageNet, nous pouvons tirer parti des caractéristiques déjà apprises par le modèle. Cela est particulièrement avantageux lorsque les données spécifiques à la tâche, comme les images d'IRM, sont limitées. Le modèle pré-entraîné a déjà une compréhension des caractéristiques générales des images, et il ne reste qu'à ajuster les dernières couches pour qu'elles s'adaptent à la nouvelle tâche. Cette approche permet d'améliorer considérablement la performance par rapport à un entraînement à partir de zéro.

En comparaison avec d'autres modèles comme le SVM (Support Vector Machine), ResNet50 montre des avantages clairs. Les SVM sont efficaces pour des tâches de classification avec des espaces de caractéristiques bien définis et de taille modérée. Cependant, ils ne sont pas aussi puissants pour traiter des données brutes comme les images et leurs performances sont moins efficaces que les modèles avec des réseaux de neurones. ResNet50, en revanche, peut automatiquement extraire les caractéristiques pertinentes des images sans nécessiter cette étape manuelle, ce qui le rend beaucoup plus efficace pour des tâches complexes comme la classification d'IRM.

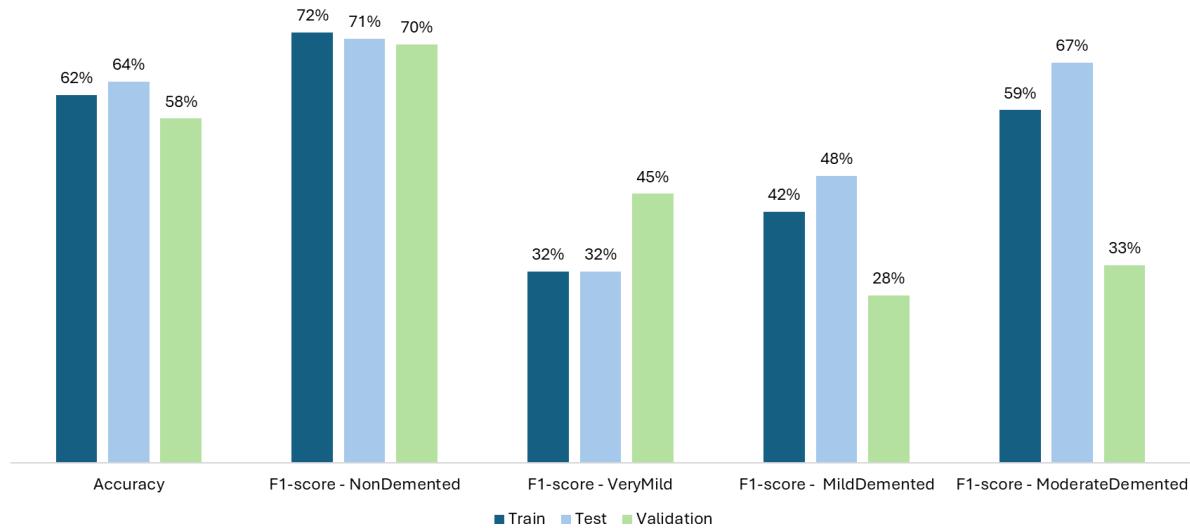
En résumé, la combinaison de la profondeur et de la complexité du modèle, l'utilisation innovante des skip connections, et les avantages du transfer learning font de ResNet50 un choix particulièrement puissant pour la classification d'IRM. Ces caractéristiques lui permettent de surpasser les modèles traditionnels comme les CNN moins profonds et les SVM, offrant une meilleure capacité d'apprentissage et de généralisation sur des ensembles de données complexes et limités, améliorant ainsi la précision et la fiabilité des diagnostics médicaux assistés par IA.

3. Modélisation

Nous avons commencé par geler les couches du Resnet50 pour obtenir une baseline sur nos données. Nous ajoutons une couche Dense et une couche dense avec la fonction d'activation softmax pour la classification.

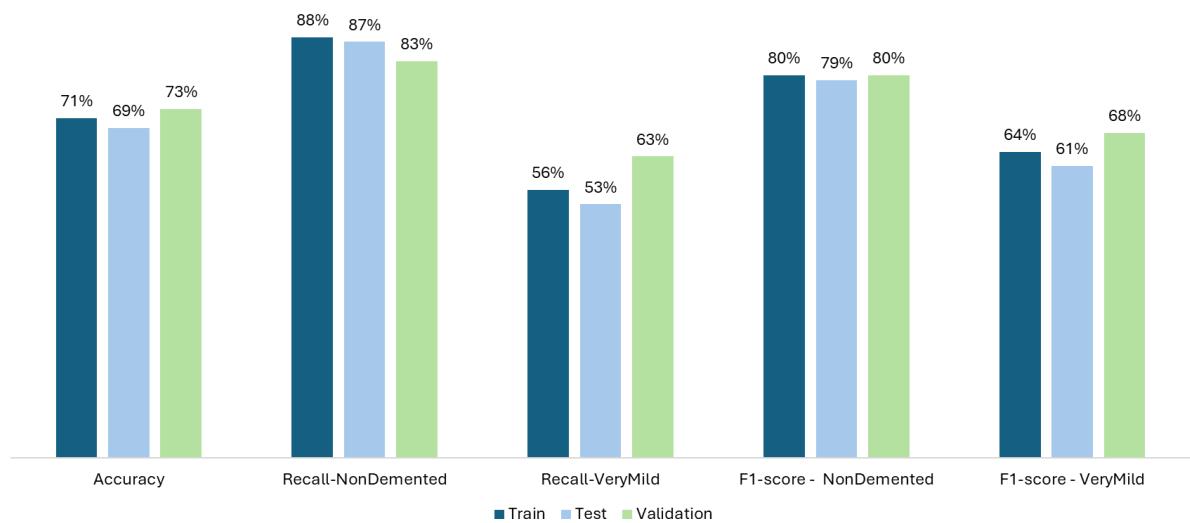
Étant donné le déséquilibre des classes, nous faisons de la data augmentation en générant des images des données d'entraînement avec des angles différents par exemple, pour aider le modèle à mieux généraliser sa prédiction.

Resnet50 Modèle Global



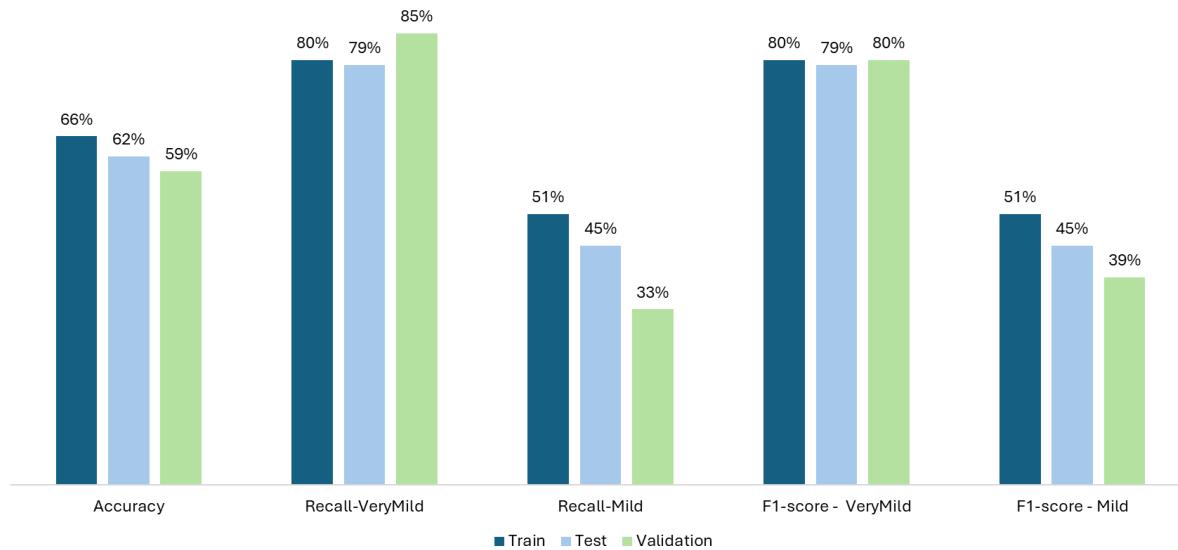
Nous avons une accuracy à 62%, ce qui n'est pas très satisfaisant dans le domaine médical. Pour offrir une plus grande précision, nous décidons de développer 3 modèles de classification binaire pour avoir une deuxième prédiction mais dans un cas plus précis, afin de mieux accompagner les utilisateurs de notre application.

Métriques Resnet50 Modèle 1



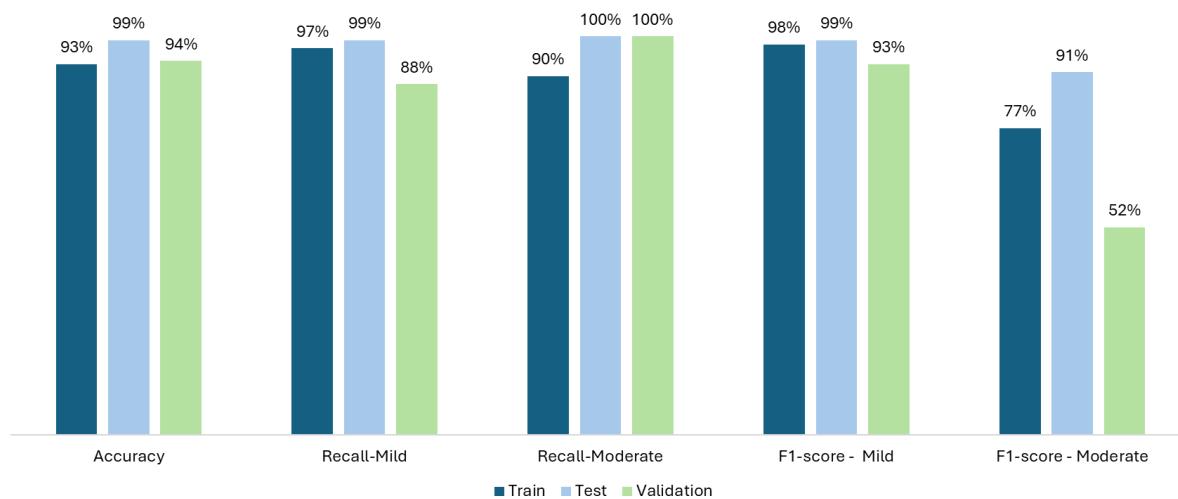
Le modèle 1 est une classification entre les individus “Sain” et ceux atteints de “Très légère démence” (very mild demented en anglais). Avec une accuracy à 69%, le modèle est plus satisfaisant que le modèle global à ce stade.

Métriques Resnet50 Modèle 2



Le modèle 2 est une classification entre les individus atteints de “Très légère démence” et “Légère démence”, c'est un modèle très intéressant car à l'aide des questionnaires et autres tests, il n'est pas possible de déceler des symptômes liés à la maladie d'Alzheimer pour les “Très légère démence”. On a une accuracy à 61%, le modèle n'est pas forcément bon, voire il effectue du surapprentissage.

Métriques Resnet50 Modèle 3



Le modèle 3 fait la classification entre les “Légère démence” et “Démence modérée”, des classes où la maladie prend de plus en plus de place dans la vie des malades, le début de la perte d'autonomie des patients. Notre modèle a une accuracy de 99%, il établit une bonne frontière de décision et se révèle très pertinent par rapport au modèle global.

Conclusion des modèles baseline

Les modèles proposés ne sont pas satisfaisants car trop peu précis pour un intérêt médical aussi important. Alors nous décidons d'ajouter des couches pour que le modèle de transfer learning puisse aussi apprendre sur nos données d'entraînement équilibrées et augmentées.

VIII. Modèle finaux

Pour adapter ResNet50 à notre tâche spécifique de classification d'IRM, nous avons ajouté plusieurs couches au modèle pré-entraîné. Voici une explication détaillée de chaque ajout et de son rôle dans l'amélioration des performances du modèle.

Construction du Modèle

Nous avons commencé par créer un modèle séquentiel avec Keras et ajouté `base_model`, qui correspond au modèle ResNet50 pré-entraîné, sans sa couche de classification finale. Cela permet d'utiliser ResNet50 comme extracteur de caractéristiques, capitalisant sur ses capacités d'apprentissage profond.

Ajout de Dropout

Ensuite, nous avons ajouté une couche Dropout avec un taux de 0.5. La couche Dropout aide à régulariser le modèle et à prévenir l'overfitting en désactivant de manière aléatoire 50% des neurones pendant l'entraînement. Cela oblige le modèle à ne pas trop dépendre de certains neurones spécifiques, ce qui conduit à des représentations plus robustes et généralisables.

Flattening

Nous avons ensuite intégré une couche Flatten qui convertit les sorties multi-dimensionnelles de ResNet50 en un vecteur unidimensionnel. Cette transformation est nécessaire pour passer les données à des couches entièrement connectées (Dense), qui sont couramment utilisées dans les couches de sortie des réseaux de neurones.

Batch Normalization

Pour améliorer la stabilité et la vitesse d'entraînement, une couche de Batch Normalization a été ajoutée. Cette couche normalise les activations des neurones pour chaque mini-lot, réduisant ainsi la covariance shift et permettant un apprentissage plus efficace.

Première Couche Dense

Nous avons ajouté une couche Dense avec 32 neurones et une initialisation des poids selon la méthode he_uniform. Cette initialisation est particulièrement adaptée pour les couches utilisant la fonction d'activation ReLU, car elle aide à maintenir les gradients dans des plages appropriées, facilitant ainsi l'apprentissage.

Seconde Batch Normalization et Activation

Une seconde couche de Batch Normalization suit la couche Dense pour normaliser les activations. Cela est suivi par une couche d'activation ReLU (Rectified Linear Unit), qui introduit de la non-linéarité dans le modèle, permettant de mieux capturer les relations complexes dans les données.

Couche de Sortie

La couche finale du modèle est une couche Dense avec un nombre de neurones égal au nombre de classes dans notre problème de classification (représenté par class_num). La fonction d'activation softmax est utilisée ici pour transformer les sorties en probabilités, facilitant ainsi l'interprétation des résultats comme des scores de probabilité pour chaque classe.

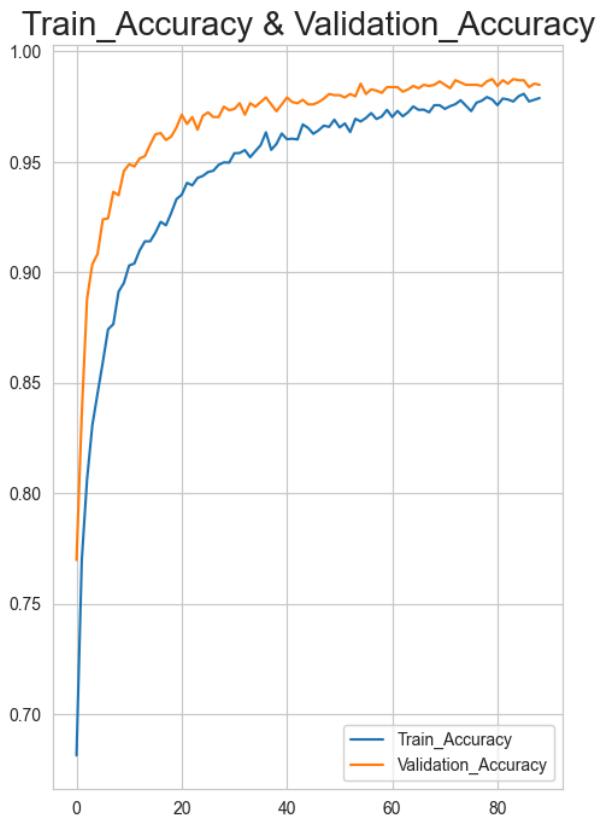
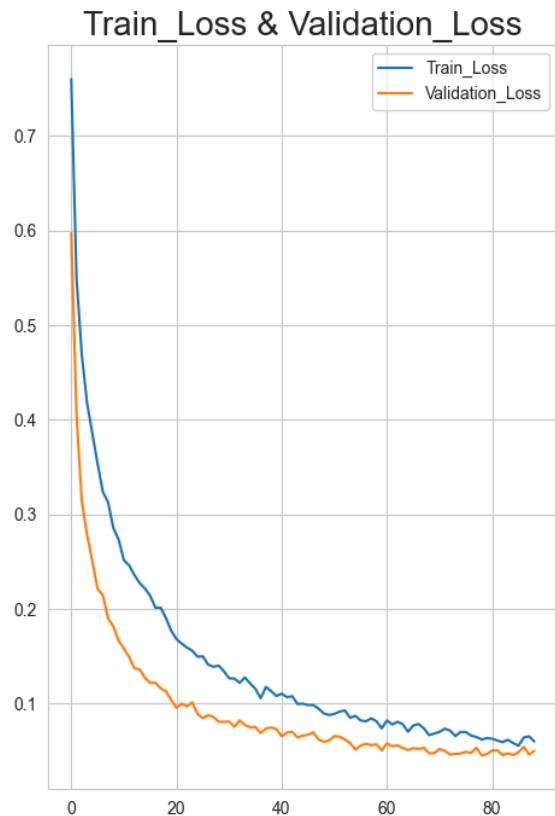
Compilation

Enfin, la méthode build spécifie la forme des entrées du modèle.

Normalement les images en entrées pour Resnet50 sont de taille (224, 224) mais nos images étant plus petites, nous avons décidé de les redimensionner en (176, 176). Cette réduction de la taille a eu un impact positif sur l'entraînement du modèle.

IX. Modèle global

Entraînement



Voici ce que représente les classes :

- 0 : Légère démence

- 1 : Démence modérée
- 2 : Sain
- 3 : Très légère démence

Nous avons une accuracy de 98% pour l'entraînement, ce qui est très bon signe avec les graphiques sur la loss et l'accuracy puisque les données de test montrent de meilleures performances.

Validation

	0	1	2	3
0	1	0	0	0
1	0	1	0	0
2	0.0015625	0	0.992188	0.00625
3	0.00446429	0	0.00446429	0.991071

Avec 99% d'accuracy sur les données de validation, nous confirmons que notre modèle global a très bien compris les différences entre chaque classe.

Nous développons les modèles de classification binaire comme précédemment, ils montrent tous des performances entre 98 et 99%.

Nous avons intégré une fonctionnalité de visualisation Grad-CAM (Gradient-weighted Class Activation Mapping) à notre application pour mieux comprendre quelles parties des images d'IRM sont les plus importantes pour les prédictions du modèle. Cette approche aide à mettre en évidence les pixels qui ont le plus contribué aux décisions du modèle, en analysant les gradients de la dernière couche dense où se trouve notre fonction d'activation "softmax".

Prétraitement des Nouvelles Données de Test

Pour commencer, nous avons prétraité les nouvelles données de test. Nous avons lu les images, les avons redimensionnées à la taille utilisée lors de l'entraînement, et les avons normalisées pour garantir la cohérence des données. Les étiquettes des images ont

également été encodées et converties en format one-hot, compatible avec notre modèle de classification.

Évaluation et Prédictions

Une fois les données de test préparées, nous avons évalué le modèle ResNet50 sur ces nouvelles données pour vérifier sa performance. Nous avons calculé la précision et la perte, et généré un rapport de classification détaillé pour évaluer les performances du modèle sur chaque classe. Nous avons également construit une matrice de confusion pour visualiser les performances de classification.

Création des Cartes de Chaleur Grad-CAM

Pour générer les visualisations Grad-CAM, nous avons utilisé la technique de Grad-CAM qui permet de créer des cartes de chaleur. Cette méthode consiste à analyser les gradients de la dernière couche convulsive du modèle ResNet50 par rapport à la sortie de la couche de classification. Plus précisément, nous avons extrait les gradients des prédictions les plus probables par rapport aux activations de la dernière couche convulsive. En moyenne, ces gradients pondérés ont été multipliés par les activations de la couche convulsive pour obtenir une carte de chaleur qui indique les zones les plus importantes de l'image.

La carte de chaleur est ensuite redimensionnée pour correspondre à la taille de l'image d'origine et superposée sur cette dernière en utilisant une échelle de couleur, ce qui permet de visualiser clairement les zones d'intérêt.

Sauvegarde et Affichage des Cartes de Chaleur

Nous avons mis en place une fonction pour sauvegarder et afficher ces cartes de chaleur superposées aux images originales. Cette fonction utilise les cartes de chaleur générées pour chaque image et les combine avec les images d'origine en ajustant la transparence afin de visualiser les zones d'intérêt sans masquer complètement l'image sous-jacente.

Sélection et Visualisation des Images

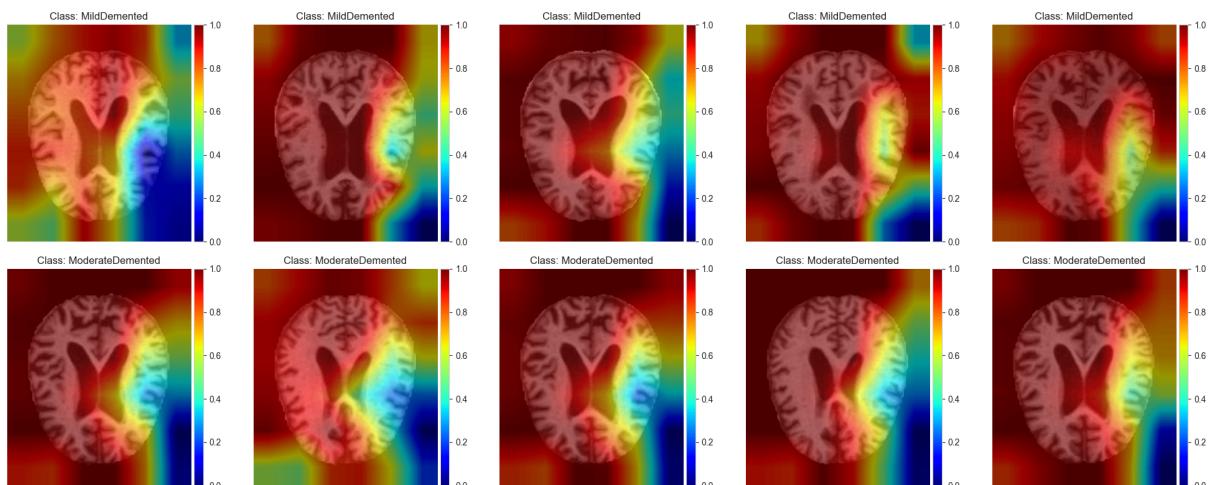
Pour rendre cette analyse plus ergonomique, nous avons sélectionné un échantillon d'images représentatives de chaque classe pour générer et afficher les visualisations Grad-CAM. En sélectionnant cinq images par classe, nous avons pu illustrer les différentes manières dont le modèle identifie les caractéristiques pertinentes pour chaque type de démence (Légère démence, Démence modérée, Sain, Très légère démence).

Ces visualisations ont été organisées dans une grille pour faciliter la comparaison. Chaque image est accompagnée de sa carte de chaleur correspondante, permettant de voir les zones que le modèle considère importantes pour sa prédiction. Une échelle de couleurs a

également été ajoutée à côté de chaque carte de chaleur pour interpréter les intensités de la carte.

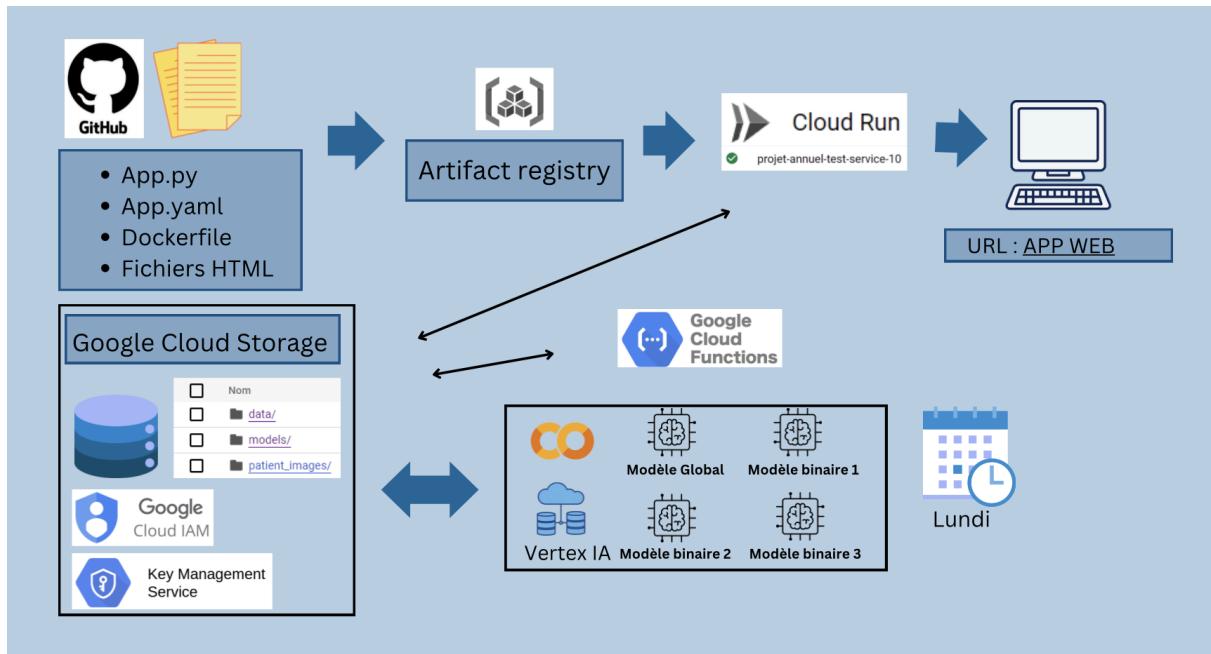
Conclusion

L'intégration de la visualisation Grad-CAM dans notre application a considérablement amélioré notre capacité à interpréter les décisions du modèle. En visualisant les zones des images qui contribuent le plus aux prédictions, nous avons pu obtenir des informations précieuses sur le fonctionnement interne de notre modèle ResNet50. Cela nous permet non seulement de mieux comprendre pourquoi le modèle prend certaines décisions, mais aussi d'identifier des moyens potentiels pour améliorer encore sa performance et sa fiabilité. Ces visualisations sont particulièrement utiles pour les applications médicales où l'interprétabilité et la transparence des modèles de machine learning sont essentielles pour la confiance et l'acceptation par les professionnels de santé.



Les zones rouges indiquent des parties de l'image qui ont été importantes pour la prédiction, tandis que les zones bleues indiquent un impact nul.

X. Déploiement sur le cloud



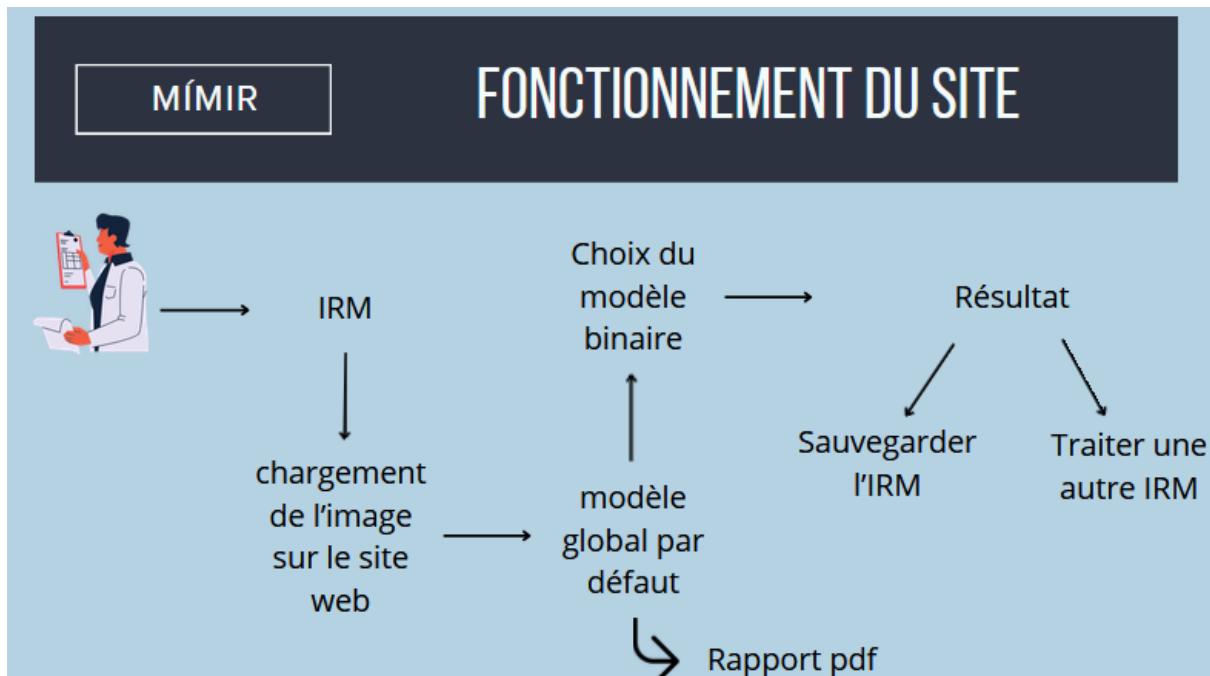
Pour notre projet, nous avons décidé de déployer l'application sur Google Cloud Storage, en nous inspirant de notre structure initiale (voir image de l'infrastructure du projet).

Pour le stockage des données et des modèles de notre projet, nous avons choisi d'utiliser Google Cloud Storage. Ce service offre des fonctionnalités de sécurité avancées telles que le chiffrement des données au repos et en transit, ainsi que la gestion des identités et des accès (IAM), que nous avons activé pour protéger nos données. En utilisant Google Cloud Storage, nous bénéficions d'une solution fiable, sécurisée et évolutive qui répond parfaitement aux exigences de notre projet. Nous avons utilisé un bucket situé à Paris, ce qui permet de réduire les émissions de CO₂. Le coût de ce service est de 0,025 centime par mois, ce qui est très abordable.

En outre, nous utilisons Cloud Run pour lancer l'application. Pour ce faire, nous avons créé un Dockerfile pour construire une image qui a été ensuite poussée vers le container registry, ajoutant ainsi une étape supplémentaire à notre déploiement sur le cloud. Nous avons mis en place des accès IAM pour sécuriser le site web. Nous avons également créé un autre container pour stocker l'image Docker, avec les mêmes configurations que le container destiné à nos données. Les fonctions Cloud permettent d'alimenter notre base de données avec de nouvelles images enregistrées par les utilisateurs. Les modèles sont entraînés sur Vertex AI via Google Colab en utilisant les données téléchargées et les notebooks Jupyter.

Vertex AI gère le processus d'entraînement et fournit des modèles prêts à être déployés, qui seront stockés dans notre bucket. Les modèles seront entraînés tous les dimanches.

XI. Présentation de l'application MIMIR



Nous avons développé un site web pour permettre la prédition de la maladie d'Alzheimer en utilisant différents modèles de machine learning.

lien du site : [MIMIR](#)

Structure des Fichiers

Notre site est construit avec 4 fichiers principaux :

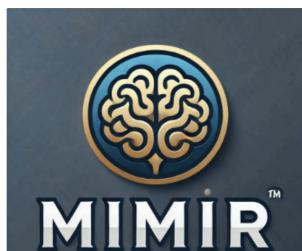
1. **index.html** : La page d'accueil permet aux utilisateurs de charger une image à partir de leur appareil pour effectuer une prédition.
2. **result_global.html** : Page de prédition par défaut, qui applique la prédition avec le modèle global et une carte de chaleur générée par Grad-CAM avec un top 3 des classes prédites et une matrice de confusion. Il est possible d'utiliser un modèle de classification binaire pour affiner l'analyse depuis cette page.

3. **result_binaire.html** : Même type d'affichage que result_global.html mais avec une prédiction binaire.
4. **style.css** : La feuille de style définit l'apparence générale du site, incluant la mise en page, les couleurs, les polices, et les boutons.
5. **app.ipynb** : Script Python pour la gestion des routes et de la logique de l'application.

Nous avons opté pour une architecture simple mais efficace pour permettre aux utilisateurs de charger des images, de choisir des modèles de prédiction et de visualiser les résultats de manière intuitive. Les différentes pages sont conçues de manière minimalistre pour offrir une expérience utilisateur agréable, tandis que le backend en Python gère la logique de traitement des images et des prédictions.

B. Visuelle du site et utilisation

Dans la première page nous aurons :



Sur la première page de notre site en haut on voit le logo présentant un cerveau. Ce design rappelle qu'on travaille sur l'alzheimer. De plus, on affiche le nom du site en bas du logo.

Charger une image

Aucun fichier sélectionné.

En dessous nous avons la partie pour parcourir les dossiers pour charger une IRM pour le traitement de tailles 176 x 208 pixels en vue du dessus

Vers le bas du site web, sous le chargement d'image, nous avons ajouté des descriptions sur les différents stades de l'Alzheimer ainsi qu'une explication des différentes métriques et des

modèles, ainsi qu'un exemple d'image que le site accepte. Cela permet à l'utilisateur d'être bien conscient de ce qu'il va voir.

Nous passons à la 2ème page:

Résultat de la prédiction globale



Classe globale prédictive : Démence légère

Temps de prédiction : 2.22 secondes

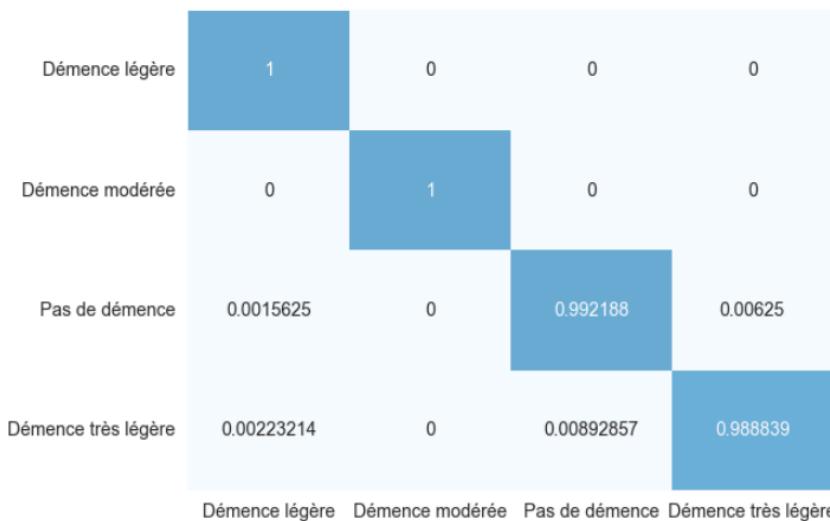
Légende de la carte de chaleur: Rouge (forte importance) à Bleu (faible importance)

Une fois l'image chargée, nous aurons les résultats d'une prédiction globale concernant un cas de démence légère dans notre cas. Nous aurons en informations la classe globale prédictive ("Démence légère"), le temps de prédiction ("2.22 secondes") et une légende de la carte de chaleur ("Rouge (forte importance) à Bleu (faible importance)").

Top 3 des prédictions globales

Classe	Confiance
Démence légère	100.0%
Pas de démence	0.0%
Démence très légère	0.0%

Matrice de Confusion



Nous avons ensuite le "Top 3 des prédictions globales" qui montre la classe prédite avec le plus haut niveau de confiance. Ainsi que la "Matrice de Confusion" qui visualise les performances du modèle en montrant la répartition des prédictions correctes et incorrectes pour chaque classe de démence de notre modèle global.

Sauvegarder l'image dans :

Nom de l'image :

Sauvegarder

Sélectionner un modèle binaire

Pas de démence vs Très légère démence **Démence très légère vs Démence légère** **Démence légère vs Démence modérée**

Télécharger le rapport PDF **Retour à l'accueil** **Charger une autre image**

Ensuite, on pourra sélectionner différents modèles binaires, télécharger un rapport PDF, sauvegarder l'image avec différents choix de statut dans le cas où le modèle se serait trompé, ou bien charger d'autres images.

En choisissant les modèles binaires, nous aurons la même chose visuellement que pour le modèle global. On retrouve la même structure pour la troisième page.

C. Coût du projet

Nous avons déployé notre projet sur gcp et nous avons pu bénéficier des 300\$ de crédits offerts à la création d'un compte. Nous n'avons en réalité rien eu à dépenser et les montants dont nous allons parler ci-dessous ont été déduits de nos crédits.

Le projet nous a coûté 60,21 euros au total.

Nous avons testé différents services gcp. Celui qui nous a le plus coûté a été vertex ai avec les 53 euros dépensés. Les autres coûts étaient principalement des tests que nous avons effectués mais ne seront pas des dépenses courantes après mise en production.

Vertex AI : Nous utilisons des modèles d'exécution de type N2-highcpu32 avec 100 Go de stockage. Ce qui revient à 20,10 € / mois pour le modèle d'exécution avec 4 heures de charge de travail 1 jour par semaine (tous les lundis). Le stockage associé nous revient à 1,87€ / mois environ.

Cloud Storage : Nous avons choisi comme emplacement la région europe-west9 (Paris). Le coût standard associé est de 0,023\$ par mois.

Networking : Nous avions défini le bucket comme étant à Paris, cependant les machines vertex AI que nous avons choisi d'utiliser par la suite n'étaient pas disponibles à Paris et nous avons alors choisi la Belgique. Ceci a engendré de légers frais de transferts de données (inférieur à 1 euro dans notre cas). Il aurait peut-être fallu redéfinir un bucket en Belgique pour ne pas avoir ce souci.

Cloud Function : Lorsque l'on enregistre une image sur le site mimir, elle arrive dans le dossier patient-images/ et non le dossier data/train/. Nous avons créé une cloud function qui lorsqu'une image arrive dans le dossier patient-images/, renomme l'image en lui ajoutant le suffixe '_patient-images' pour reconnaître le fait que l'image ne venait pas du dataset de base puis la déplace l'image dans data/train (et dans le bon dossier correspondant au stade d'alzheimer associé comme c'était dans le dossier patient-images/).

Nous avons défini que cette fonction s'exécuterait dès qu'il y aurait 2 images enregistrées pour pouvoir la tester rapidement. On voulait qu'après ceci les modèles soient directement réentraînés avec les nouvelles données dans le dossier data/train mais finalement mettre un time Scheduler sur les notebooks de vertex AI a été plus simple. Le coût de la cloud function correspond au coût des opérations de classe sur le cloud storage.

Nous utilisons des opération de classe A (storage.objects.list et storage.objects.copy) qui reviennent à 0,005\$ pour 1000 opérations et des opérations de classe B (storage.objects.get et storage.objects.delete) qui reviennent à 0,0004\$ pour 1000 opérations.

Artifact Registry : Pour déployer notre site nous avons eu à utiliser Artifact Registry. Nous avions choisi de déployer notre projet dans la région europe-west9 (Paris). La tarification de ce service est de 0,10\$ par Go par mois (car notre app dépasse les 0,5 Go).

Cloud run : Nous avons déployé l'app et la cloud function dans cloud run. Nous avions choisi d'utiliser 3Gi pour la mémoire et 8 CPU pour l'app et 16Gi de mémoire et 8 CPU pour la cloud function. Le processeur est uniquement alloué lors de la requête. Les 257 142 premiers Gio-seconde sont gratuits par mois puis c'est 0,0000035\$ par Gio pour la mémoire. Pour le CPU, les 128 571 premières secondes de vCPU-secondes sont gratuites par mois puis c'est 0,0000336\$ par vCPU-seconde.

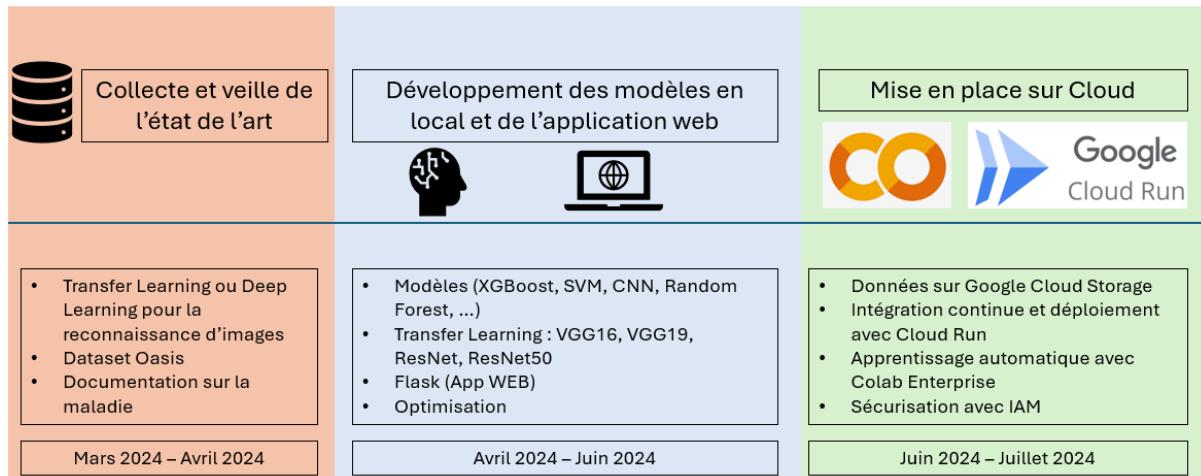
XII. Gestion du projet

1. Planning

Nous avons commencé par rechercher des données disponibles pour notre projet sur l'Alzheimer. Il s'agissait de la partie 1 du projet.

Par la suite, nous nous sommes concentrés ensemble sur le développement des modèles en explorant différents algorithmes pour trouver le modèle optimal, soit la partie 2 : le déploiement local. C'est en trouvant de très bons résultats avec Resnet50 que nous nous sommes concentrés pleinement sur l'application web. Une fois que le déploiement local a été un succès, nous avons pu passer à la partie 3 du projet : le cloud.

Il fallait adapter notre structure locale à un environnement cloud. Nous avions initialement prévu d'utiliser des services différents tels que AI Platform sur GCP (Google Cloud Platform), mais AI Platform a été intégré à Vertex AI.



2. Répartition de la charge de travail

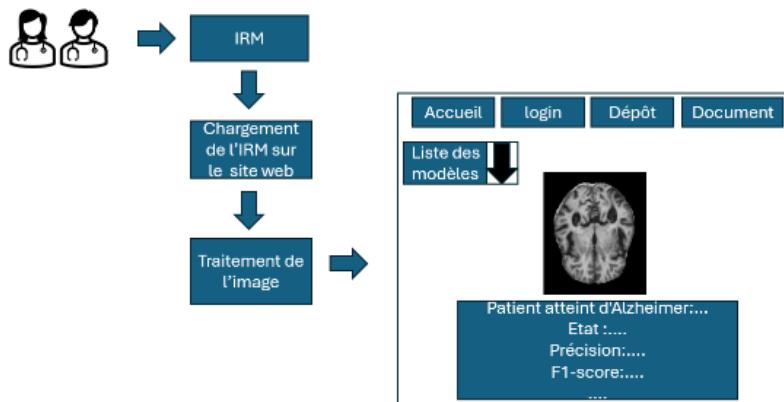
Nous avons tous participé activement à chaque phase du projet, avec certaines disparités :

- Jean-Marc : développement des modèles, de l'application web et de l'automatisation cloud de l'entraînement
- Louis : développement des modèles, de l'application web et automatisation du Cloud Run
- Nadejda : développement des modèles, de l'application web et automatisation du Cloud Run

En sachant que l'ensemble de l'équipe a effectué la veille technologique, la recherche des données et la documentation sur la maladie.

3. Première approche du projet

Nous avons abordé le projet avec la proposition fonctionnelle suivante et technique :



Design de l'Infrastructure sur GCP (1ère approche)

1. **Stockage des données** : Utilisation de Google Cloud Storage pour stocker les images d'IRM.
2. **Intégration continue des données** : Mise en place de Cloud Build pour automatiser les tests et le déploiement du code.
3. **Entraînement du modèle** : Utilisation de AI Platform pour entraîner les modèles XG Boost ,CNN et SVM.
4. **Prédiction** : Déploiement du modèle entraîné sur AI Platform Predict pour servir les prédictions via une API.
5. **Labellisation**: insertions des images dans la classe prédite.
6. **Monitoring** : Utilisation de Cloud Monitoring pour surveiller la performance et la santé de l'application.
7. **Déploiement de l'application** : Mise en place de l'application web avec Cloud Run ou App Engine pour fournir une interface utilisateur pour télécharger les images IRM et afficher les résultats de prédiction.

4. Difficultés rencontrées

Nous n'avons pas développé la fonctionnalité de compte à travers notre application par manque de temps, mais nous avons développé une fonctionnalité permettant de télécharger un rapport pdf.

Les fonctionnalités de GCP ont changé durant le projet, AI Platform est devenu Vertex AI par exemple. Alors nous avons dû nous adapter et utiliser de nouvelles fonctionnalités.

Notre plus grande difficulté restera la collecte de données, dans le domaine médical il ne s'agit pas d'une tâche facile car les données de santé sont considérées comme des données

très sensibles. Nous n'avons pas pu récupérer les données dans un format d'image 2D, ce qui rendait le projet trop complexe.

5. Ce qu'on pourrait améliorer

Pour améliorer notre application, nous pourrions utiliser des données de meilleure qualité : image IRM complète en 3D avec les mesures chimiques pour que le diagnostic soit bien plus fiable. De plus, développer la fonctionnalité de comptes utilisateurs afin de consulter son historique ou des dossiers de patients. Nous pourrions aussi sur la partie cloud mettre de la sécurité via FireBase et avoir un domaine pour l'application ainsi que du monitoring via Cloud Monitoring.