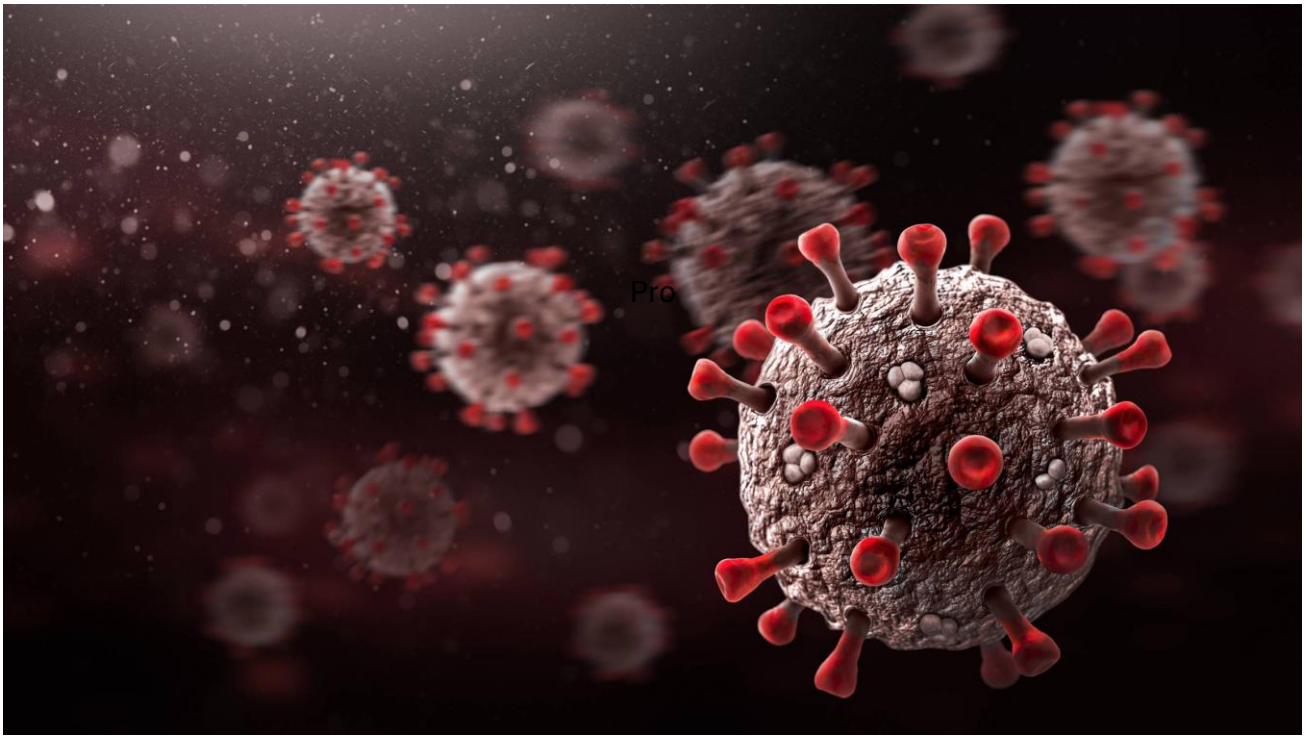
Rapport final : Développement d’une aide au diagnostic médical.

Utilisation d’un modèle de Deep Learning pour le diagnostic de pathologie pulmonaires



Projet réalisé par : Ambroise Dehaye, Romain Chichoux, Morvan Fortier et Grégoire Balluais

Introduction du projet :

Contexte :

Un constat évident dans le domaine médical est la haute spécialisation et la qualité des expertises du personnel soignant. Chaque branche de la médecine repose sur des compétences spécifiques qui permettent d'établir des diagnostics précis et fiables. Cependant, cette spécialisation implique un nombre limité de spécialistes, ce qui peut entraîner une saturation rapide du système de santé dans les contextes de forte demande.

Cette surcharge impacte directement la qualité des soins et peut conduire à une diminution significative de la précision des diagnostics.

Un exemple récent et marquant est la pandémie de SARS-CoV-2, qui a mis à rude épreuve le système de santé en provoquant une saturation des services spécialisés en maladies pulmonaires. De nombreux patients ont en effet contracté des formes graves de l'infection, ce qui a fortement sollicité les structures hospitalières.

Pour prévenir l'apparition de cas graves et protéger les personnes dites « à risque », différentes méthodes de détection ont été mises en place afin d'identifier les infections à un stade précoce.

Le test de détection le plus couramment utilisé repose sur un prélèvement nasopharyngé, suivi d'une analyse par RT-PCR (Reverse Transcriptase - Polymerase Chain Reaction). Cette méthode consiste à rechercher la présence de séquences spécifiques d'ARN du virus, absentes chez les autres organismes, ce qui permet de confirmer l'infection à SARS-CoV-2 dans les voies respiratoires supérieures du patient[7].

Cependant, cette méthode présente certaines limites, notamment en termes de sensibilité, qui varie en fonction du type et de la localisation du prélèvement. Une étude menée sur 205 patients a mis en évidence des taux de positivité différents selon les échantillons analysés : 93 % pour les lavages broncho-alvéolaires, 72 % pour les expectorations, 63 % pour les prélèvements nasaux, et seulement 32 % pour les prélèvements de gorge[4].

Par ailleurs, le taux de faux négatifs – c'est-à-dire les cas où un patient infecté n'est pas détecté – dépend du stade de l'infection. Par exemple, la probabilité d'un faux négatif est de 100 % le premier jour suivant l'infection, diminue à 38 % au cinquième jour (coïncidant généralement avec l'apparition des symptômes), atteint un minimum de 20 % au huitième jour, puis remonte à 66 % au vingt-et-unième jour[2].

En plus de ces limitations cliniques, la RT-PCR reste une méthode coûteuse, avec un prix moyen d'environ 54 € à la charge de la sécurité sociale par patient et par test[11].

D'autres types de tests sont également disponibles, comme les tests sérologiques, qui permettent de détecter la présence d'anticorps spécifiques au SARS-CoV-2. Toutefois, ces tests ont aussi leurs limites, notamment dans les phases précoces de l'infection, car les anticorps ne sont généralement détectables qu'à partir du septième jour après l'exposition au virus.

Face à ces limites diagnostiques, notamment le coût, le taux de faux négatifs et la variabilité des résultats selon la méthode de prélèvement, l'imagerie médicale s'est imposée comme une alternative ou un complément précieux, notamment dans la détection des atteintes pulmonaires liées au SARS-CoV-2.

L'imagerie par rayons X (radiographies thoraciques) et la tomodensitométrie (scanner thoracique) permettent de visualiser les signes caractéristiques d'infections pulmonaires, tels

que les opacités en verre dépoli ou les condensations alvéolaires[8]. Ces outils, bien qu'efficaces, nécessitent une expertise médicale approfondie pour une interprétation fiable, ce qui peut engendrer des délais en cas de surcharge des services radiologiques.

Objectifs :

L'objectif est donc le développement d'un outil basé sur des modèles de Deep Learning. Cet outil d'aide à la décision permet de désengorger la charge de travail des médecins spécialisés, en diagnostiquant les pathologies possibles présentes sur une radiographie pulmonaire. Le jeu de données utiliser pour l'entrainement de ces modèles est disponible en open source sur Kaggle et va être présenté dans la partie suivante.

Compréhension et manipulation des données :

Cadre :

Le jeu de données qui est utilisé ici provient d'une initiative de chercheurs de l'université du Qatar et de Dhaka, avec la collaboration de docteurs malaisiens et pakistanais, à regrouper et annoter un gros volume de données images de radiologie à rayons X pulmonaires[1,3] de patients qui sont regroupés en 4 catégories distinctes :

- **Lung Opacity**, correspondant aux anomalies présentes sur les poumons de type occlusions.
- **COVID**, qui correspond aux patients infectés par le Sars-CoV-2 et ayant contracté des lésions visibles localisées dans la région pulmonaire.
- **Viral Pneumonia**, correspondant aux cas d'infection virale visible dans la région pulmonaire.
- **Normal**, qui sont les patients sans anomalie visible sur les radiologies pulmonaires et qui sont considérés comme sains.

Ce jeu de données a été mis à disposition librement aux utilisateurs sur la plateforme Kaggle, nous n'avons donc pas eu de problèmes à exploiter ces données. À noter que, dans le cadre de l'obtention d'imageries médicales, certaines limites entrent en compte dans le cadre du secret médical ainsi que des bases de données non structurées en interne des CHU, ce qui a limité notre base de données à celle produite par ces chercheurs.

Ce jeu de données d'un volume de 806.84 MB (.zip) contient donc les quatre classes d'images mais aussi un filtre associé pour chaque image. Ces filtres permettent, avec une compilation

de l'image associée, de masquer les éléments visuels qui ne correspondent pas aux poumons du patient.

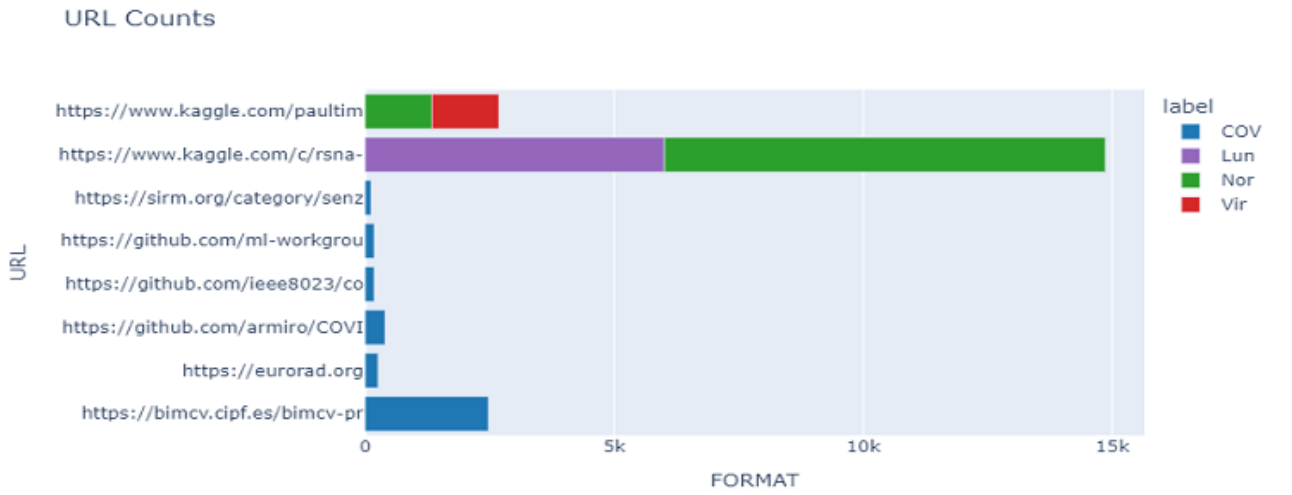


Figure 1 : Sources des images du jeu de données utilisé.

Les sources des données sont multiples avec une disparité quantitative et qualitative. En effet, (figure 1) six sources sont à l'origine des images pour la classe COVID, contre seulement deux pour les trois autres classes. Une limite possible mise en avant ici est potentiellement la diversité des images concernant les trois classes Lung Opacity, Normal et Viral Pneumonia, mais cela ne s'est pas avéré être un risque après une analyse visuelle et la qualité des données est restée acceptable.

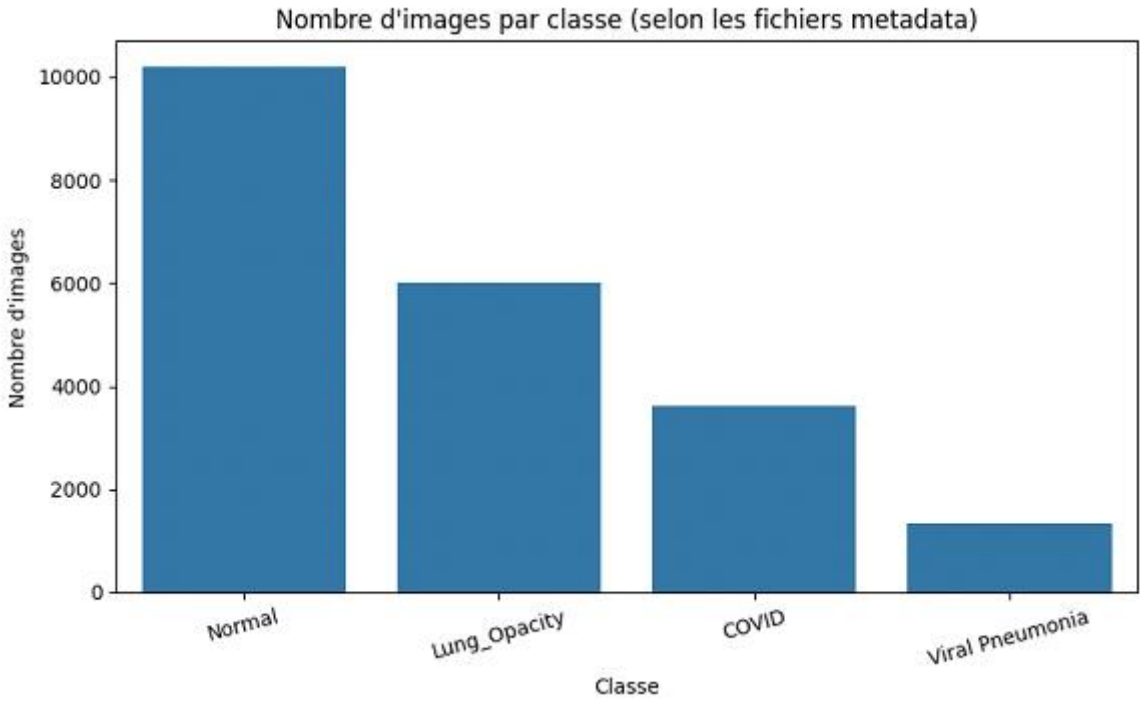


Figure 2 : Déséquilibre des classes dans le jeu de données utilisé.

Le premier problème mis en avant en explorant ce jeu de données est le déséquilibre entre les classes (figure 2). En effet, le nombre d'occurrences d'images de la classe Normal est six fois plus important que la plus petite classe, c'est-à-dire Viral Pneumonia. Ce déséquilibre de classes peut avoir un impact important sur la phase d'entraînement des modèles sélectionnés, notamment avec un surapprentissage sur les classes majoritaires. Des techniques de sur et sous-échantillonnage vont être présentées et utilisées dans le cadre de la préparation de notre jeu de données avant la phase d'entraînement du modèle.

Visualisation et Statistiques :

En s'intéressant plus en détail au jeu de données présenté ici, il est possible de sortir certaines variables d'une image permettant l'exploration de leurs caractéristiques. À titre d'exemple, il est intéressant de sortir la luminosité moyenne par image et par classe, le contraste moyen ou bien même l'aire de certains composants visuels (comme les poumons). Cette approche exploratoire permet de faire ressortir les tendances du jeu de données et de dégager les limites de celui-ci qui seront exploitées dans la partie pré-processing.

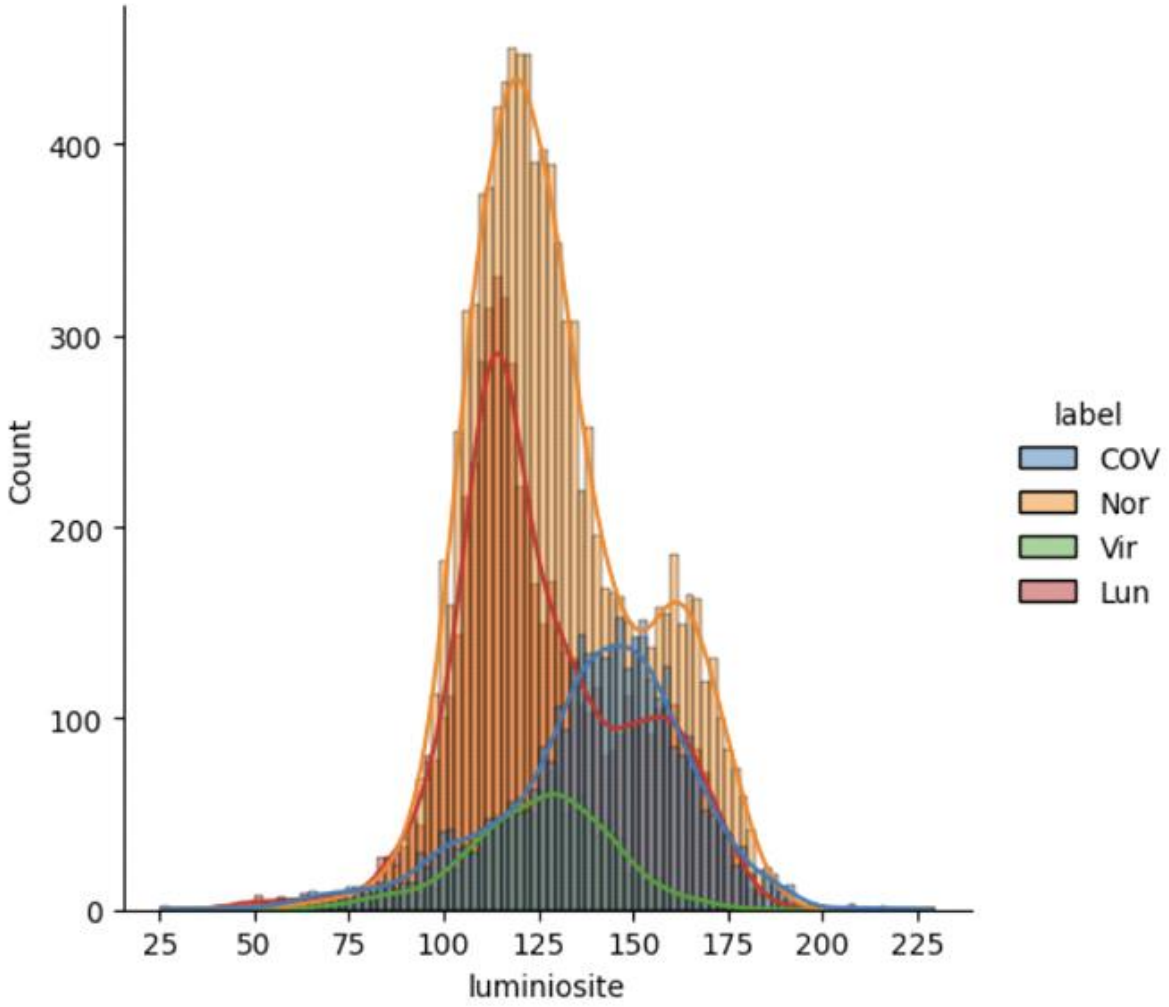


Figure 3 : Répartition de l'intensité lumineuse moyenne par image par classe.

La répartition par classe de l'intensité lumineuse (Figure 3, chaque classe est représentée par une couleur) indique des pics d'abondance relativement différents entre les classes, avec un pic de luminosité à 125 pour *Normal* contre 150 pour *COVID*. Il est donc probable que les techniques de prise de vue ne soient pas standardisées ou bien que les protocoles liés à l'obtention de ces images soient relativement différents entre les centres d'analyse pulmonaire. Cette différence d'intensité lumineuse entre les classes est validée par un test ANOVA démontrant une variance significative entre les classes (alpha à 0,05). De plus, le test post-hoc de Tukey HSD démontre que chaque classe a une variance significativement différente.

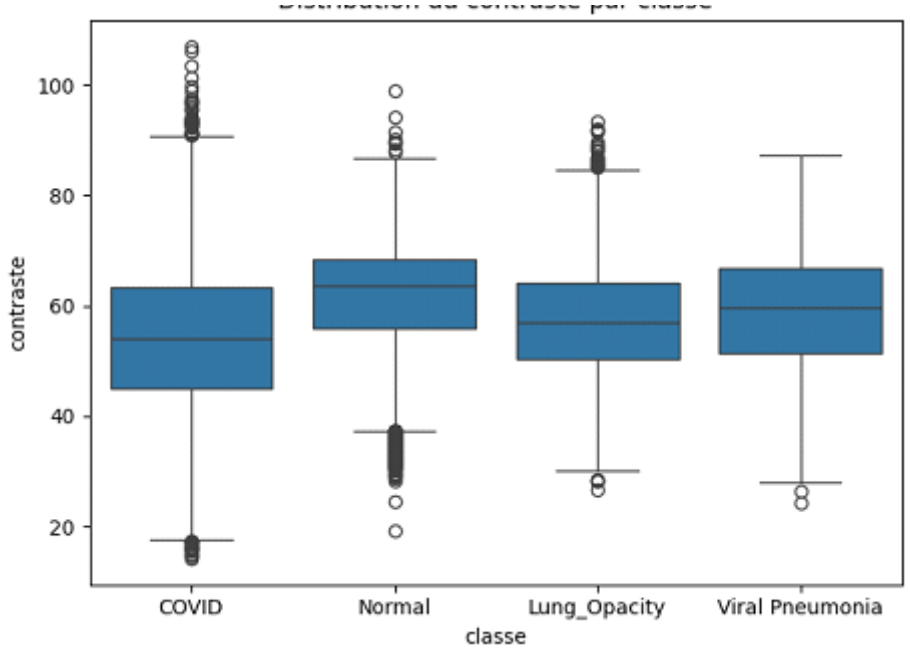


Figure 4 : Distribution du contraste par classe.

À contrario, la distribution du contraste mise en avant dans la Figure 4 ne montre graphiquement pas de différence, malgré une distribution plus large pour la classe *COVID*. On pourrait émettre l'hypothèse que la plus forte distribution de cette classe est due au nombre important de sources qui ont été compilées à sa création (se référer à la Figure 1). La même hypothèse peut être émise pour les classes avec les distributions les plus centrées, c'est-à-dire *Lung Opacity* et *Viral Pneumonia*, qui ne possèdent qu'une seule source. Une différence de variance significative est cependant mise en avant par un test ANOVA entre les classes. Le test post-hoc de Tukey HSD permet de dégager une différence de variance significative entre les classes sauf entre *COVID* et *Normal*, possiblement en raison de leurs grandes distributions.

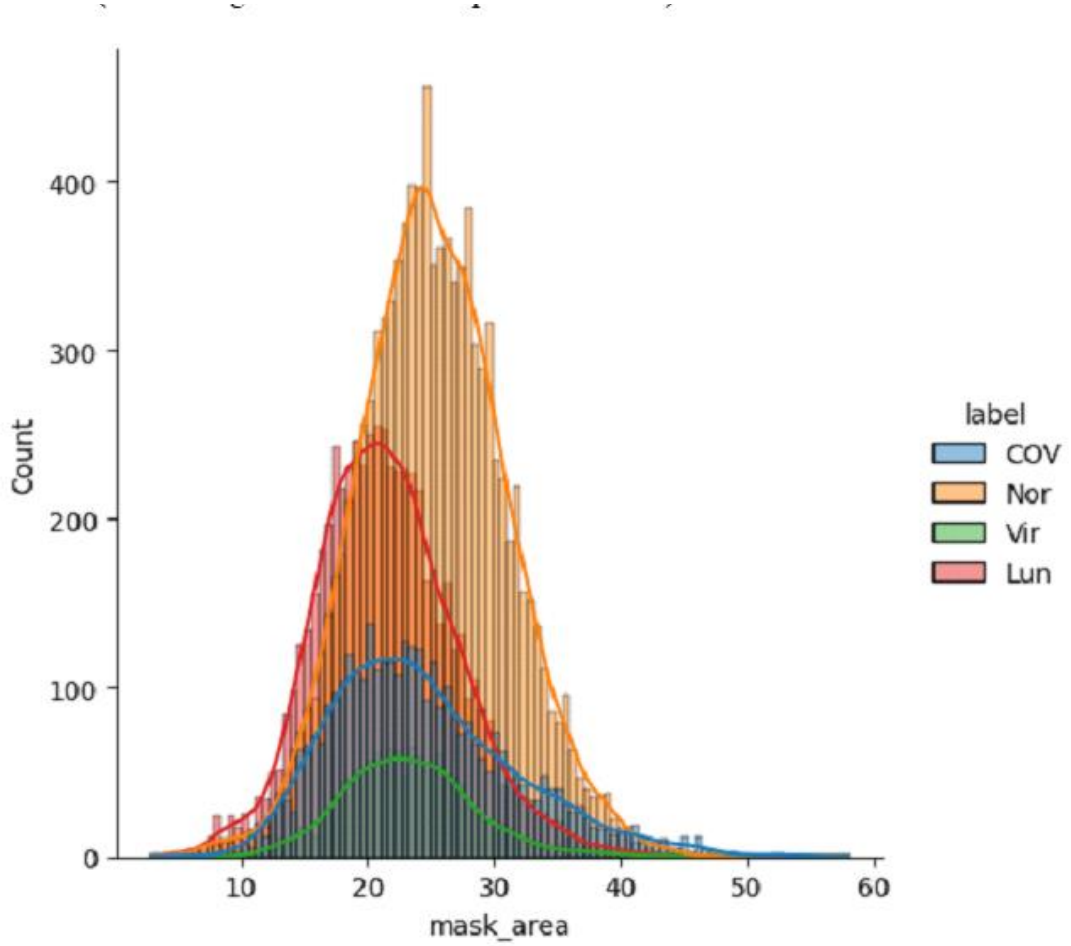


Figure 5 : Distribution de l'air des masques par classe.

Les masques sont une segmentation des images de base permettant, en superposant ce masque à son image d'origine, d'obtenir une représentation exclusive des poumons. Le protocole permettant la réalisation de ces masques n'est pas clairement indiqué dans les métadonnées. Cependant, ces mêmes chercheurs ont développé un modèle de segmentation des poumons sur des images de radiologie de patients infectés par le Sars-Cov-2, nommé *OsegNet*[90]. D'autres modèles de segmentation ont peut-être été utilisés comme *U-Net*[9].

En regardant la distribution des surfaces des masques en fonction des classes auxquelles ils appartiennent (Figure 5), on ne voit aucune tendance qui se dégage ; les surfaces des poumons des patients sont homogènes et identiques entre les classes. Cependant, les variances entre les classes sont significativement différentes (ANOVA), mais la taille différente des classes pourrait être à l'origine de ces résultats.

Pertinence :

En regardant le jeu de données par le prisme de la problématique du projet, celui-ci est de très bonne qualité avec une annotation claire et vérifiée par des spécialistes. Aucune manipulation n'est à prévoir sur cet aspect-là. Cependant, le déséquilibre très marqué entre les différentes classes va devoir être remédié.

Pre-processing et feature engineering :

Après avoir mis en avant les limites et les avantages du jeu de données présenté ci-dessus, une phase de pre-processing a été réalisée en abordant plusieurs approches indépendamment. Ces différentes approches sont en accord avec le processus de réflexion de l'équipe pour préparer les données à l'étape de modélisation, ainsi qu'avec les limites mises en avant pendant l'étape exploratoire. La pertinence de chaque transformation sera discutée et testée au travers de la problématique de ce projet.

Nature des fichiers :

Il est primordial de s'intéresser aux caractéristiques propres du jeu de données et notamment à la répartition des poids entre les classes. En effet, une différence de poids des images pourrait mettre en avant une différence de qualité entre les classes ou encore une variance plus importante d'une classe tendrait le modèle à être plus robuste dans les prédictions pour cette dernière.

#	Dossier	Moyenne	Médiane	Écart-type	Min	Max	Fichiers
0	Covid	35836.43	36186	4552.82	9913	52004	3616
1	Lung_Opacity	35228.88	35269	3565.40	13878	48292	6012
2	Normal	37266.68	37346	3088.64	14178	54043	10192
3	ViralPneumonia	40304.59	38598	6542.73	27005	73131	1345

Table 1 : Statistiques exploratoires des données par classe.

Dans le cas du jeu de données initial (Table 1), la taille moyenne des images en octets est relativement constante entre les classes. Il est à noter que *Viral Pneumonia* ainsi que *Normal* possèdent un poids moyen légèrement supérieur. Cette différence peut être facilement mise en relation avec leur source commune, qui pourrait expliquer cette tendance (voir Figure 1).

Cependant, la variabilité de la classe *Viral Pneumonia* est plus importante en comparaison aux autres.

Cette approche exploratoire de la taille des images met l'accent sur les caractéristiques intrinsèques de la création de ces images. Aucune transformation n'est à prévoir sur cet aspect en raison de l'homogénéité relative du poids des classes mise en avant par cette analyse.

Reduction des dimensions par PCA (Analyse par composantes principales) et auto-encoding :

Plusieurs techniques de réduction de dimension ont été testées lors de ce projet. Cette approche réductrice découle de plusieurs constats et limites techniques :

Réduction du temps d'entraînement et de charge mémoire : en effet, une contrainte majeure du projet est la capacité de calcul limitée mise à notre disposition. L'utilisation de machines de calcul personnelles induit une contrainte de réduction du temps de calcul. Les techniques de réduction de dimension permettent donc de réduire le nombre de pixels par image et donc de limiter les ressources nécessaires (nombre de paramètres, temps de calcul, GPU…).

Réduction du surapprentissage et du bruit : cette approche réduit la redondance entre pixels voisins et permet de conserver les composantes les plus informatives en supprimant une spécificité limitante à l'apprentissage avec données images : le bruit. Dans le cas de notre jeu de données, ce bruit peut être représenté par les artefacts matériels visibles comme des patchs médicaux ou encore la présence de baleines de soutien-gorge visibles sur ce type d'imagerie médicale.

Dans le cadre de la première approche de réduction de dimension, une analyse en composantes principales (PCA) a été appliquée au jeu de données.

Les images ont été chargées dans l'environnement de travail au format 128x128 (soit 16 384 pixels).

Une PCA avec *EigenImages* a été appliquée au jeu de données, permettant la création de vecteurs propres aux données de la matrice d'autocorrélation, adaptée à la réduction des dimensions d'une image. Cette approche réductrice permet de diminuer considérablement le nombre de dimensions en gardant seulement les 50 plus explicatives, tout en conservant une variance expliquée cumulée de 89 %.

Les limites de cette approche résident dans son fonctionnement. En effet, la linéarité des transformations appliquées aux images capture très mal les structures complexes

indispensables dans le cadre de diagnostics médicaux, telles que les opacités diffuses ou les consolidations hétérogènes

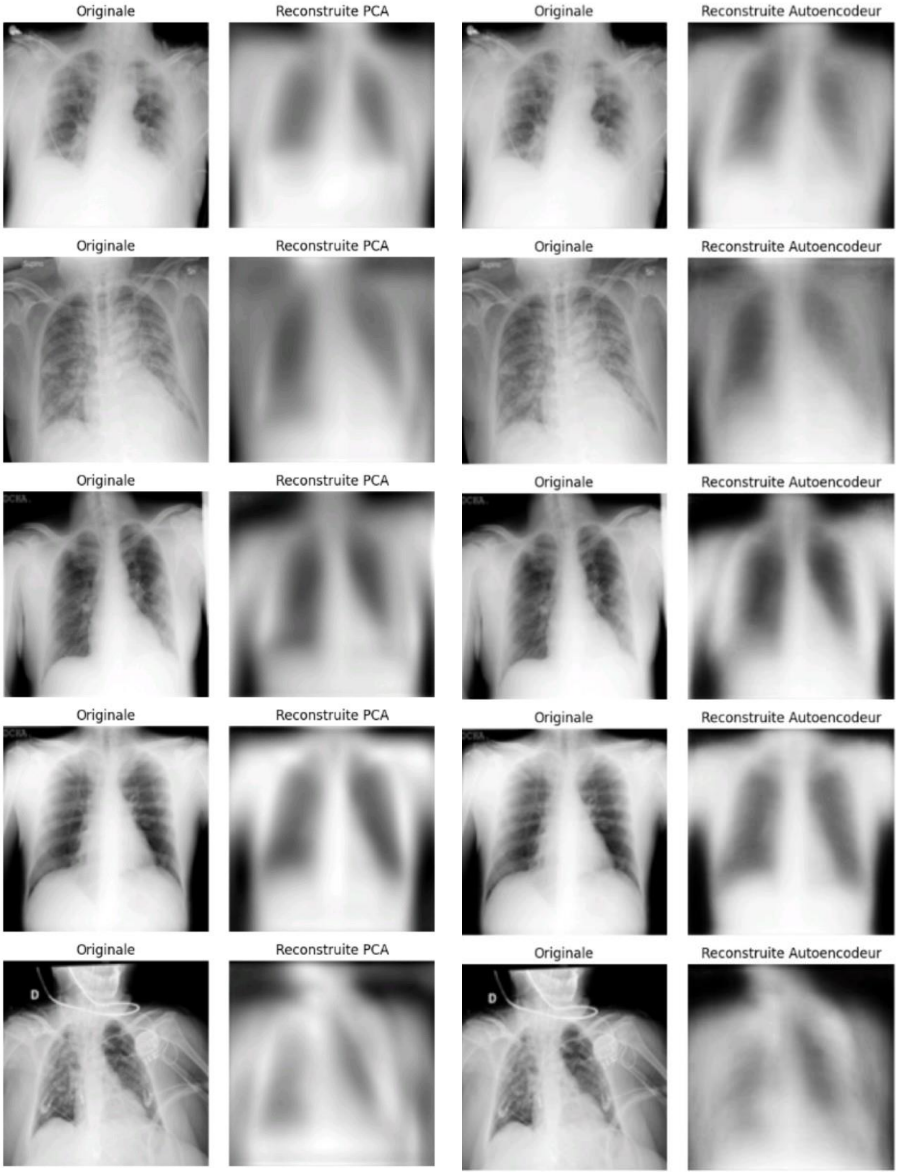


Figure 6 : Redictions des dimensions par PCA (à gauche) et par auto-encoding dense(à droite).

À la suite des résultats obtenus par l'analyse en composantes principales, une autre méthode de réduction du nombre de dimensions des images du jeu de données a été utilisée et comparée à la transformation par PCA.

L'utilisation d'un autoencodeur dense permet, grâce à un réseau de neurones dense simple, de capturer les structures visuelles complexes et non linéaires tout en réduisant les

informations non essentielles. Cette approche est donc très pertinente dans le cadre de radiographies pulmonaires.

L'entraînement du modèle a été réalisé sur toutes les images normalisées avec une taille de batch de 256 et pendant 20 époques.
Après entraînement et visualisation des reconstructions, plusieurs structures complexes sont présentes (contours pulmonaires, détails texturaux, etc.). Dans la perspective du développement de modèles de classification d'images, cela est primordial.

Pour appuyer la pertinence de la réduction de dimension par autoencodage dense, une représentation en 2 dimensions de la variance expliquée (Figure 7) démontre une ségrégation par classe bien marquée.

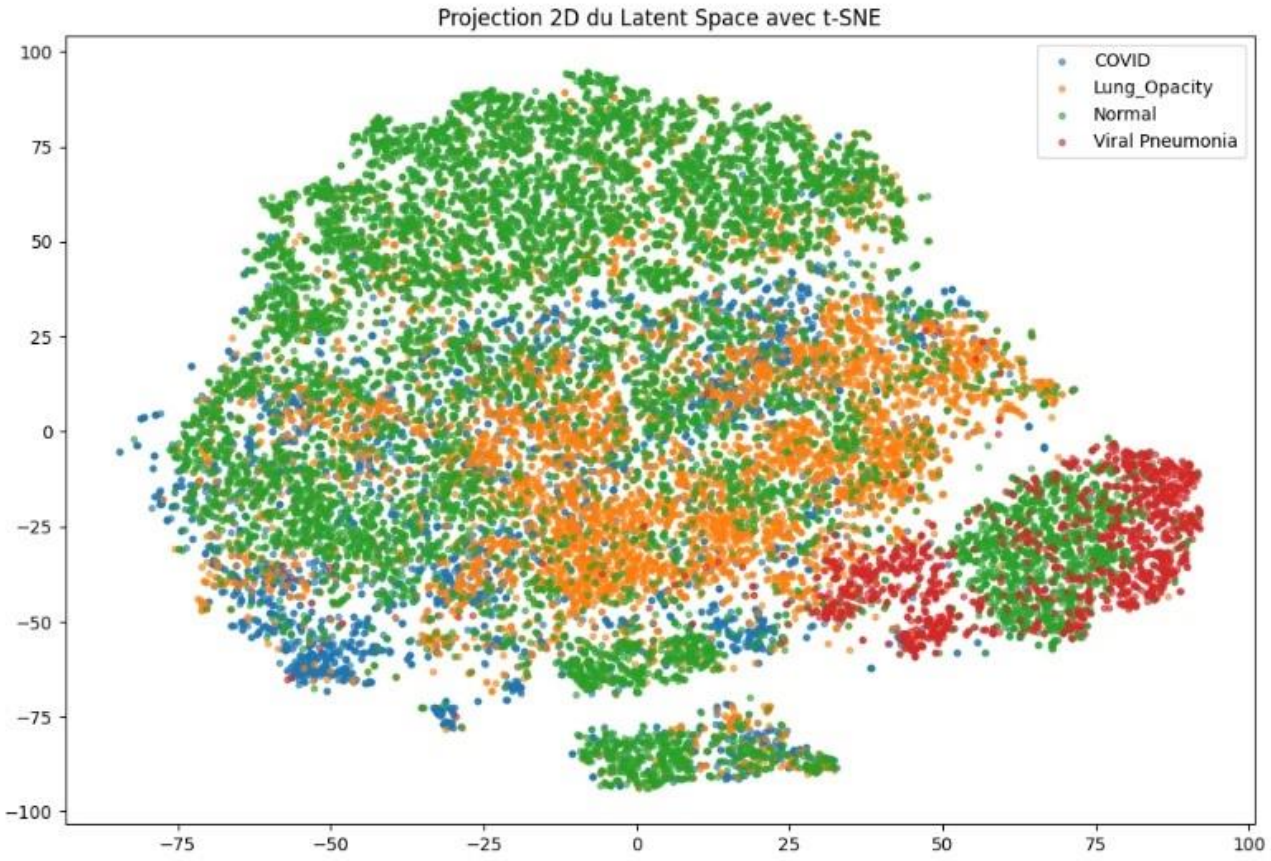


Figure 7 : Projection 2D avec t-SNE.

Cette analyse permet également de mettre en avant des caractéristiques visuelles communes
En effet, la classe *Viral Pneumonia* semble être fortement clusterisée, traduisant une homogénéité radiographique de cette classe.

Les cas de *COVID* (en bleu) apparaissent relativement dispersés, ce qui reflète la variabilité clinique et radiologique de la maladie, connue pour ses manifestations pulmonaires hétérogènes selon les patients et les stades d'évolution.

Les cas *Normal* (en vert) sont logiquement plus étalés, correspondant à la diversité physiologique des poumons sains, sans motif pathologique spécifique à regrouper.

La classe *Lung Opacity* (en orange) présente une distribution diffuse et intermédiaire, ce qui est cohérent avec la nature même de cette catégorie. En effet, les opacités pulmonaires peuvent résulter de diverses pathologies (non-COVID), avec des caractéristiques radiographiques variées (infiltrats, condensations, etc.). Cette hétérogénéité explique l'absence de cluster net et le chevauchement observé avec les classes *COVID* et *Viral Pneumonia*, traduisant une frontière floue entre ces pathologies sur le plan radiologique.

L'autoencodeur dense entraîné sur les images du dataset permet donc une reconstitution pertinente des informations.
Pour déterminer la fidélité de cette reconstitution, il est intéressant de calculer l'indice de similarité structurelle (SSIM) entre l'image d'origine et l'image reconstruite. La valeur moyenne de cet indice est de 65 %, ce qui est cohérent avec le nombre de dimensions choisi (50). À noter que durant cette approche, le nombre d'époques et le nombre de dimensions ont été testés afin d'évaluer leurs effets sur les performances de l'autoencodeur dense, et que les meilleurs résultats (coût de calcul et performance) sont présentés ici.

Une dernière approche a été réalisée dans l'optique de réduction de dimensions : l'autoencodeur convolutionnel. Cette approche permet de pallier les limites évoquées par l'autoencodeur dense, telles que la perte d'informations spatiales et la qualité moyenne des reconstructions. Cette architecture est spécifiquement conçue pour exploiter la structure 2D des images, en capturant efficacement les motifs locaux (contours, textures, anomalies pulmonaires).

L'entraînement de ce modèle a donc été effectué sur la totalité des images, avec un *batch size* de 256, un nombre de 20 *epochs* et une fonction de perte "Binary Crossentropy". Cette fois, le SSIM moyen des images reconstruites est de 0,91 %, ce qui traduit une reconstruction très pertinente (Annexe 10). De même, la projection 3D du *Latent Space* après autoencodage convolutionnel (Annexe 11) démontre des tendances similaires à celles de l'autoencodage dense et donc une représentation des informations dans l'espace qualitatif. **Cette dernière approche a donc été choisie pour notre pipeline de *preprocessing*.**

Analyse de similarité, SSIM et Hash :

Une des approches utilisées dans le pipeline de prétraitement est la recherche d'images en double. En effet, il est pertinent d'appliquer cette analyse pour identifier les images similaires et ainsi éliminer les informations redondantes.
Pour réaliser cette analyse, les indices de similarité structurelle (SSIM) ont été calculés (traitement d'image avec la librairie `skimage` de Scikit-image) entre chaque image d'une même classe, à l'aide de la moyenne et de la variance d'une fenêtre glissante, pixel par pixel (gradient allant de –1 « anti-corrélation » à 1 « corrélation parfaite »).

Une réduction de la taille des images a été effectuée en amont (50×50 pixels) afin de diminuer le temps de calcul.
Un seuil a été fixé (0.8) pour ségréger les images présentant une similarité réelle. Une analyse par *hash* conceptuel a ensuite permis d'identifier les doublons.

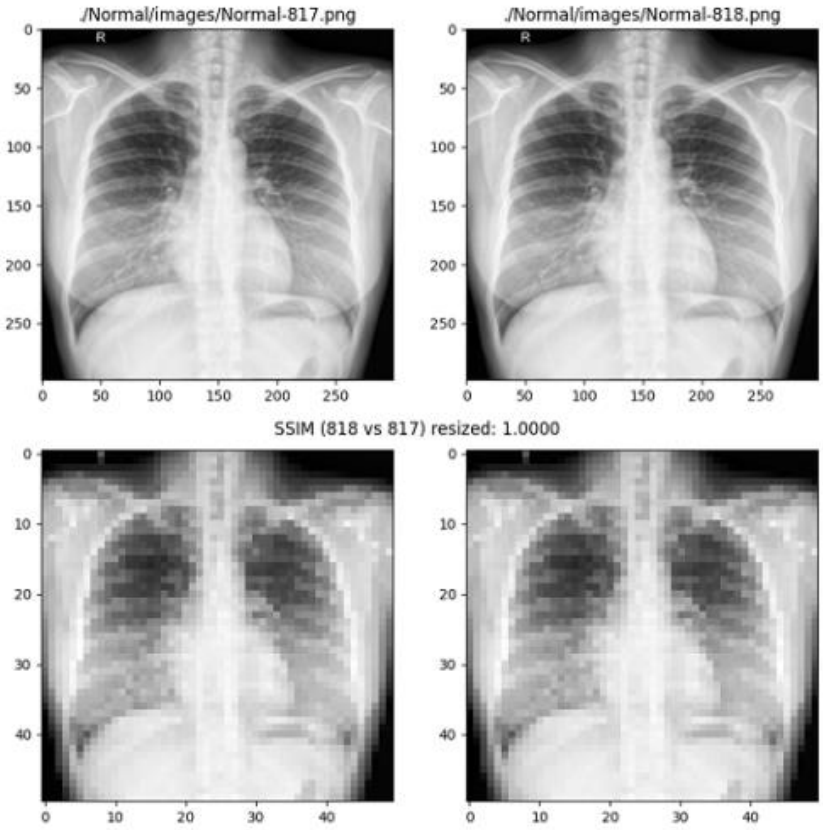


Figure 8 : Analyse de similarité par SSIM et Hash.

La Figure 8 met en évidence un exemple de doublon détecté par analyse de similarité SSIM et *hash* conceptuel. Ces doublons ont donc été supprimés dans le pipeline de prétraitement de ce projet.

Analyse par matrice de Laplacien :

Dans le cadre de notre pipeline de prétraitement, nous avons intégré une étape de filtrage fondée sur l'analyse différentielle des images par l'opérateur de Laplacien. L'objectif est d'éliminer automatiquement les images floues ou de faible qualité visuelle, susceptibles de perturber l'apprentissage du modèle de classification.

L'opérateur de Laplacien permet d'estimer la seconde dérivée de l'intensité lumineuse d'une image, détectant ainsi les zones de forte variation locale (contours, détails structurels). Il s'applique sous forme de convolution entre un noyau standardisé (*Laplacian Kernel*) et l'image en niveaux de gris. Le résultat est une matrice dérivée, dont la variance constitue un indicateur pertinent de la quantité d'information visuelle présente dans l'image.

Cette méthode, bien que simple et rapide, s'est révélée particulièrement robuste dans le contexte de données médicales en imagerie radiographique, pour identifier les images de moindre qualité.

L'opérateur de Laplacien a été appliqué à l'aide de la fonction `cv2.Laplacian(gray, cv2.CV_64F)` de la bibliothèque OpenCV. La profondeur `CV_64F` assure une précision suffisante dans le calcul des dérivées discrètes. Après convolution, nous avons calculé la variance du résultat et appliqué un seuil d'exclusion.

Un premier seuil empirique de 50 (moyenne – écart-type) a été choisi pour éliminer les images floues, permettant de conserver entre 76 % et 90 % des images selon les classes. Cette première approche n'a cependant pas permis un rééquilibrage des classes.

Nous avons ensuite affiné ce seuil à l'aide de tests systématiques. Des modèles MobileNet ont été entraînés sur des échantillons filtrés avec des seuils décroissants (de t = 50 à t = 30). Il est apparu que le seuil de 30 offrait le meilleur compromis entre conservation des images pertinentes, amélioration de la qualité des données, et performance de classification.

Cette approche semble donc permettre de filtrer la qualité des images tout en réduisant marginalement la taille du dataset, entraînant ainsi des pertes de performance très limitées.

Sur- et sous-échantillonnage par augmentation d'images :

Une étape cruciale a consisté à compenser le déséquilibre de classes présent dans le jeu de données initial. En particulier, les catégories *COVID-19* et *Viral Pneumonia* sont nettement sous-représentées par rapport aux classes *Normal* et *Lung Opacity*, ce qui pourrait conduire à un biais d'apprentissage défavorable pour les pathologies rares.
Afin d'atténuer ce déséquilibre, nous avons mis en œuvre une stratégie d'augmentation d'images ciblée, spécifiquement appliquée aux classes minoritaires.

L'augmentation d'images (*image augmentation*) consiste à générer artificiellement de nouvelles images à partir des images existantes en leur appliquant une série de transformations visuelles mineures mais plausibles. Cette approche vise à enrichir le jeu de données de manière contrôlée (en l'absence de nouvelles données réelles), mais aussi à améliorer la robustesse du modèle en l'exposant à des variations naturelles d'images.

Figure 9 : Augmentations d'image appliquées à la classe *COVID*.

Les transformations appliquées aux images peuvent être de plusieurs natures. Une contrainte importante à respecter est toutefois l'intégrité métier des images. En effet, certaines transformations, comme une rotation sur l'axe vertical ou horizontal, dégraderaient la dissymétrie naturelle du corps humain. Les transformations retenues pour cette approche sont les suivantes (Figure 9) :

- Zoom aléatoire
- Rotation
- Variation de contraste
- Variation d'intensité

Nous avons testé plusieurs combinaisons de ces paramètres pour optimiser la qualité des images augmentées tout en conservant leur validité clinique. Quatre axes principaux ont été évalués (Tableau 2) : le nombre final d'occurrences par classe, le nombre de couches de transformation, l'amplitude maximale des transformations, ainsi que la taille des *batches*.

Pour chaque combinaison testée, les images augmentées ont été mises à l'échelle (standardisation à 1./255) et sauvegardées au format 256×256 pixels.

Table 2 : Combinaison des paramètres pour l'optimisation de l'augmentation des images.

ID	Nombres d'occurances	Nombre de couches	Valeur argument	Batch size
1	4000	3	0.1	32
2	5000	3	0.1	32
3	6000	3	0.1	32
4	6000	2	0.1	32
5	6000	3	0.1	32
6	6000	4	0.1	32
7	6000	3	0.1	32
8	6000	3	0.2	32
9	6000	3	0.3	32
10	6000	3	0.1	16
11	6000	3	0.1	32
12	6000	3	0.1	64

Chaque combinaison a été testée à l'aide d'un modèle de référence basé sur MobileNet, en mesurant les performances via l'accuracy et le F1-score, des métriques bien adaptées aux jeux de données déséquilibrés.

Les résultats ont montré une amélioration significative des performances du modèle après augmentation :

Table 3 : Métrique de performance post-augmentation des classes minoritaires.

ID	1	2	3	4	6	8	9	10	12	Masks
accuracy	0.65	0.77	0.82	0.82	0.81	0.81	0.81	0.81	0.80	0.79
F1-score	0.63	0.76	0.82	0.82	0.81	0.81	0.81	0.81	0.80	0.79

La combinaison **ID 3** (Tableau 3) s'est révélée optimale, atteignant 0.82 en *accuracy* et F1-score. Elle a donc été sélectionnée pour la suite du projet. Cette configuration repose sur un compromis équilibré entre enrichissement du dataset et préservation des caractéristiques médicales pertinentes.

À noter qu'un test a également été réalisé en superposant les masques du jeu de données sur les images associées, en utilisant les mêmes paramètres que ceux de l'ID 3. Les performances de classification du modèle se sont avérées très faibles ; l'utilisation de masques a donc été exclue de notre démarche.

Il convient de souligner que ces résultats ont été obtenus à partir des images brutes. Des tests complémentaires, combinant filtres Laplaciens et augmentation d'images, suggèrent que

certaines associations peuvent légèrement réduire les performances. Ce point sera exploré plus finement lors des phases d'optimisation croisée.

L'augmentation d'images constitue une composante essentielle de notre stratégie de prétraitement. Son efficacité a été validée empiriquement par une amélioration des performances du modèle de base. Cette technique sera donc maintenue dans notre pipeline final, avec une marge d'évolution possible selon les interactions avec d'autres méthodes.

Le pipeline de prétraitement est disponible ici :

Etapes de réalisation du projet :

Classification du problème :

La vision par ordinateur semble être l'outil le plus adapté à la problématique posée ici :

Développement d'un outil d'aide au diagnostic médical rapide et robuste

En effet, le but ici est de fournir un modèle permettant de classer des images dans une des quatre catégories présentées ci-dessus. Les modèles utilisés ici sont structurés en couches de neurones denses permettant une analyse et une lecture fine des caractéristiques ségrégant chaque image dans une catégorie. Une des limites premières de l'utilisation de réseaux de couches denses est son coût de calcul avec plusieurs millions de paramètres à optimiser. L'approche utilisée ici est donc le transfert learning, permettant (grâce à des modèles pré-entraînés sur des problématiques similaires) d'optimiser le modèle sur seulement une fraction de paramètres.

Pour l'évaluation des performances de classification ainsi que la robustesse des modèles, le jeu de données a été séparé en trois (Train, val, test), permettant l'apprentissage, la validation et la mesure des métriques de façon totalement indépendante.

Les métriques utilisées ici sont la matrice de confusion, la precision, le recall et le F1-score, permettant une représentation globale des performances et de la robustesse des modèles développés. L'interprétabilité de ces derniers est possible grâce à l'utilisation de Grad-CAM donnant un gradient de pertinence à la prise de décision de chaque pixel d'une image par le modèle. Enfin, une représentation de l'indice de confiance apporté à chaque classe lors de la classification par le modèle a été effectuée, présentée dans les chapitres suivants.

Choix des modèles et optimisation :

Dans le cadre de notre projet de classification d'images radiographiques pulmonaires, nous avons opté pour une approche de transfert learning, les différents modèles et approches sont présentés ici.

Présentation des modèles :

VGG :

Le modèle VGG19 a été développé par le Visual Geometry Group (VGG)[12] de l'Université d'Oxford dans le cadre du défi ImageNet ILSVRC 2014. Il s'agit d'un réseau profond, composé de 19 couches pondérées : 16 couches convolutives suivies de 3 couches entièrement connectées (Figure 10).

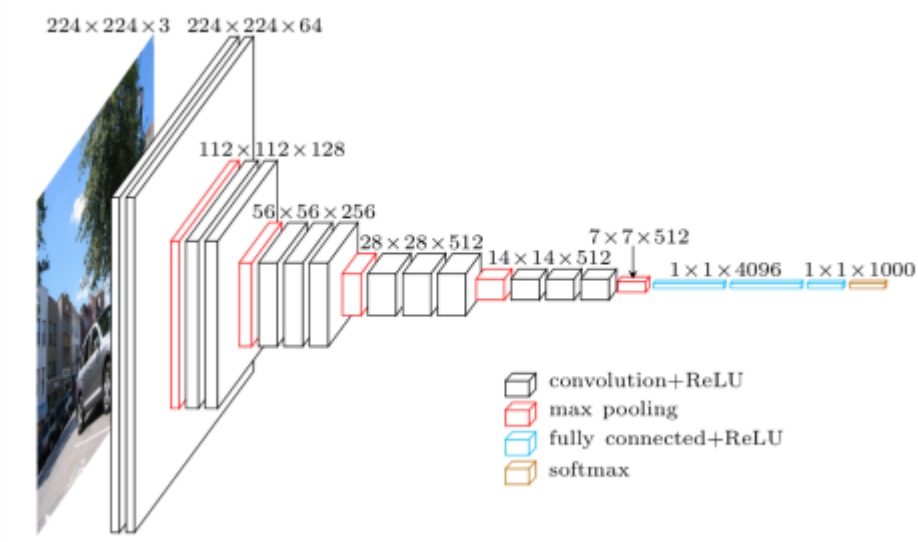


Figure 10 : Architecture de VGG 19.

Les caractéristiques principales de ce modèle sont son architecture simple et uniforme (convolutions 3x3 et max-pooling 2x2), une grande profondeur permettant une extraction hiérarchique de caractéristiques visuelles et un entraînement sur le jeu de données ImageNet.

La phase de fine-tuning a été réalisée avec l'ouverture de la dernière couche du modèle "Block5_conv4", avec en dernière couche du modèle un dropout de 0.5 et une couche dense avec comme fonction d'activation "softmax".

L'optimisation a été réalisée avec Adam en minimisant la categorical cross-entropy.

MobileNet :

Le modèle développé repose sur l'architecture MobileNet, importée via tf.keras.applications.MobileNet avec les poids pré-entraînés sur ImageNet (weights='imagenet') et sans la tête de classification d'origine (include_top=False). L'entrée des images a été redimensionnée au format (224, 224, 3) pour correspondre aux exigences du modèle. Afin de tirer parti des représentations génériques apprises sur ImageNet tout en adaptant les couches supérieures à notre tâche spécifique de classification de radiographies thoraciques, un gel partiel a été appliqué : seules les 30 dernières couches de la base MobileNet sont restées entraînables, les autres étant figées (trainable=False). La tête de classification ajoutée au-dessus de cette base se compose d'une couche GlobalAveragePooling2D() pour vectoriser les cartes de caractéristiques, suivie d'une couche dense (Dense(256, activation='relu')) accompagnée d'un Dropout(0.5) pour limiter le surapprentissage. Enfin, une couche de sortie Dense(4, activation='softmax') permet de prédire les probabilités sur les quatre classes cibles.

Pour la validation et le test, seuls les prétraitements de normalisation propres à ImageNet ont été appliqués. En raison d'un déséquilibre des classes, des pondérations manuelles ont été définies : {0: 3.0, 1: 1.0, 2: 1.0, 3: 1.0} afin d'accentuer la prise en compte de la classe COVID, jugée prioritaire.

Le modèle a été entraîné avec les hyperparamètres suivants : une taille de batch de 16, un nombre d'époques fixé à 10, et l'optimiseur Adam avec un faible taux d'apprentissage (learning_rate = 1e-5). Deux callbacks ont été utilisés : EarlyStopping sur la perte de validation (avec patience de 5 et restauration des meilleurs poids) et ModelCheckpoint pour sauvegarder les poids du modèle à chaque amélioration de la val_loss. Le processus d'apprentissage débute directement avec la phase de fine-tuning, c'est-à-dire que les 30 dernières couches sont déjà dégelées dès le départ. Toutefois, le faible taux d'apprentissage permet un ajustement progressif sans altérer les représentations génériques. Dès la première époque, l'accuracy de validation atteint 86 %, pour se stabiliser autour de 93,9 % à la fin des 10 époques.

Modèle EfficientnetB0 :

Le modèle EfficientNet[5], introduit par Google en 2019, a la particularité d'être ajustable et de mettre à l'échelle les profondeurs / largeurs et résolutions du modèle.

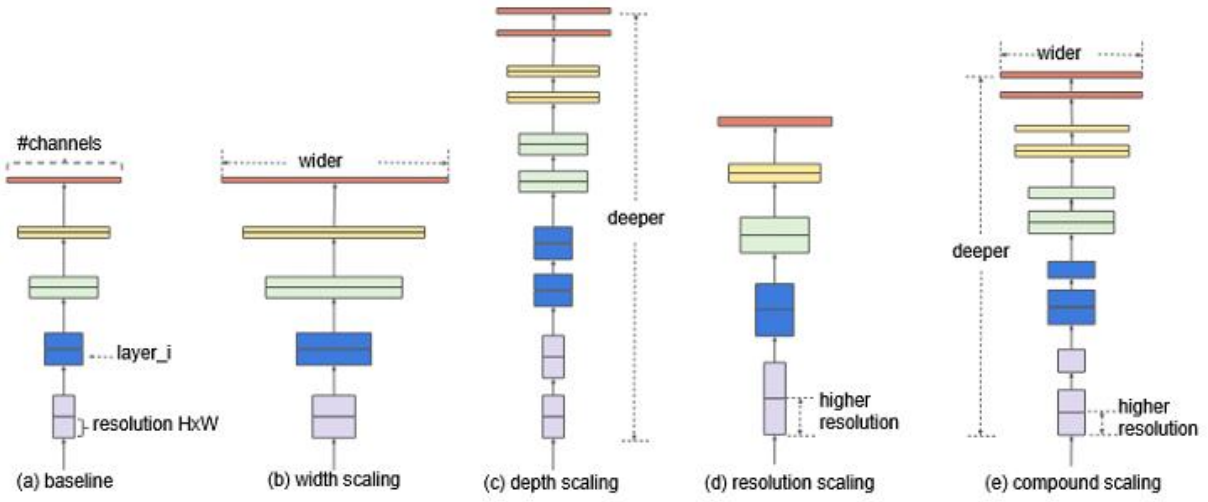


Figure 2. **Model Scaling.** (a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.

Figure 11 : Architecture d'EfficientNetB0.

Le modèle est constitué d'un ensemble de blocs MBConv (Mobile Inverted Bottleneck Convolution) utilisant des fonctions d'activations swish

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224 × 224	32	1
2	MBConv1, k3x3	112 × 112	16	1
3	MBConv6, k3x3	112 × 112	24	2
4	MBConv6, k5x5	56 × 56	40	2
5	MBConv6, k3x3	28 × 28	80	3
6	MBConv6, k5x5	14 × 14	112	3
7	MBConv6, k5x5	14 × 14	192	4
8	MBConv6, k3x3	7 × 7	320	1
9	Conv1x1 & Pooling & FC	7 × 7	1280	1

Figure 12 : Ensemble des blocs MBConv

Pour la partie classification du modèle, les couches suivantes ont été rajoutées :

```
x = layers.GlobalAveragePooling2D()(x)
x = layers.Dropout(0.3)(x)
x = layers.Dense(256, activation='relu')(x)
x = layers.Dropout(0.2)(x)
x = layers.Dense(num_classes, activation='softmax')(x)
model = models.Model(inputs=base_model.input, outputs=x)
```

Figure 13 Architecture des couches finales du modèle :

ResNet :

Avant ResNet, l'entraînement de réseaux de neurones très profonds posait des problèmes majeurs :

Dégradation des performances : Contrairement aux attentes, augmenter la profondeur d'un réseau ne menait pas toujours à de meilleures performances. Au-delà d'une certaine profondeur, l'exactitude sur les ensembles d'entraînement et de test commençait à se dégrader. Ce n'était pas dû au surapprentissage, mais à une difficulté intrinsèque pour le réseau d'apprendre des fonctions d'identité efficaces.

Problèmes de gradients : Le problème des gradients évanescents (vanishing gradients) et des gradients explosifs (exploding gradients) rendait l'entraînement de réseaux très profonds instable, car les gradients devenaient soit trop petits (empêchant l'apprentissage) soit trop grands (provoquant des mises à jour de poids erratiques).

Pour résoudre ces problèmes, les chercheurs Kaiming He, Xiangyu Zhang, Shaoqing Ren, et Jian Sun de Microsoft Research Asia ont introduit le concept de Réseaux Résiduels (Residual Networks - ResNet) dans leur article de 2015, "Deep Residual Learning for Image Recognition". Ce modèle a révolutionné le domaine de la vision par ordinateur, remportant la première place à l'ILSVRC 2015 (ImageNet Large Scale Visual Recognition Challenge) dans la tâche de classification.

L'innovation clé de ResNet réside dans les connexions résiduelles (residual connections), également appelées skip connections. Au lieu de laisser les couches apprendre directement une nouvelle transformation H(x), une connexion résiduelle permet à l'entrée x d'une couche d'être ajoutée à la sortie de ses couches suivantes.

Le bloc apprend donc une fonction résiduelle $F(x)=H(x)-x$. Si le réseau a du mal à apprendre une nouvelle transformation, il peut simplement apprendre $F(x)=0$, permettant à l'identité $H(x)=x$ de passer facilement à travers le réseau. Cela facilite la propagation des gradients à travers de

nombreuses couches, atténuant les problèmes de gradients évanescents et permettant l'entraînement de réseaux beaucoup plus profonds.

ResNet est construit à partir de blocs résiduels empilés. Il existe deux types principaux de blocs :

- Bloc d'identité (Identity Block) : L'entrée et la sortie ont la même dimension.
- Bloc de convolution (Convolutional Block) : Utilisé pour changer les dimensions spatiales ou le nombre de canaux. La connexion résiduelle inclut alors une couche de convolution 1x1 pour que les dimensions correspondent avant l'addition.

Le "50" dans ResNet50 indique que le réseau comporte 50 couches profondes qui sont entraînables. La désignation "_V2" fait référence à une amélioration de l'architecture originale de ResNet.

ResNet V1 applique la normalisation par lot (Batch Normalization) et la fonction d'activation ReLU après la couche de convolution.

ResNet V2 (proposé dans "Identity Mappings in Deep Residual Networks") utilise une stratégie de pré-activation. La normalisation par lot et la ReLU sont appliquées avant la couche de convolution. Cette modification rend le chemin d'identité encore plus propre, facilitant la propagation des informations et des gradients à travers le réseau et améliorant la stabilité de l'entraînement.

ResNet50, comme d'autres modèles ResNet plus profonds (ex: ResNet101, ResNet152), utilise des blocs "bottleneck" pour réduire la complexité computationnelle et le nombre de paramètres. Un bloc bottleneck se compose généralement de trois couches de convolution :

- Une convolution 1x1 qui réduit le nombre de canaux (le "goulot d'étranglement").
- Une convolution 3x3 sur les canaux réduits.
- Une autre convolution 1x1 qui restaure le nombre de canaux d'origine.

Cette séquence permet d'effectuer des convolutions sur un nombre réduit de canaux, puis de ré-étendre, économisant ainsi des ressources.

ResNet50_V2 est un modèle très polyvalent et largement utilisé, principalement pour :

- Classification d'images : C'est sa fonction première, notamment pour les tâches de reconnaissance d'objets dans des images.
- Apprentissage par transfert : Étant pré-entraîné sur de vastes jeux de données comme ImageNet, il est souvent utilisé comme base (backbone) pour des tâches spécifiques en ajustant les dernières couches.

- Détection d'objets : Intégré dans des architectures plus complexes comme Faster R-CNN, YOLO, ou SSD.
- Segmentation sémantique et d'instance : Utilisé comme extracteur de caractéristiques dans des modèles comme U-Net ou Mask R-CNN.
- Imagerie médicale : Pour la classification de maladies, la détection de tumeurs, etc.
- Reconnaissance faciale et autres tâches de vision par ordinateur.

Sur le jeu de données de référence ImageNet, un ResNet50_V2 pré-entraîné peut atteindre des performances impressionnantes :

- Précision Top-1 (Top-1 Accuracy) : Environ 76.0% (le modèle prédit correctement la classe unique la plus probable).
- Précision Top-5 (Top-5 Accuracy) : Environ 93.0% (la bonne classe fait partie des 5 prédictions les plus probables).

Ces performances, combinées à sa capacité à gérer des réseaux profonds sans dégradation, font de ResNet50_V2 un choix populaire et puissant pour de nombreuses applications de vision par ordinateur.

Feedback sur le pre-processing :

Après développement des premiers modèles de classification, une phase de feedback sur la pertinence des étapes de preprocessing a été réalisée. En effet, leur pertinence devait être testée empiriquement au travers des modèles de classification permettant de répondre à notre problématique.

Pour cela, un modèle en particulier va permettre d'évaluer les étapes du pipeline, DenseNet201 présenté ci-dessus.

Table 4 : Mesure de performance de l'entrainement d'un modèle basé sur DenseNet201 pour chaque étape du pre-processing.

Type de base de données	Accuracy mesurée (DenseNet201)
Equilibré, et split	0.89
Equilibré, augmenté, et split	0.92
Equilibré, augmenté, lapiacien et split	0.91
Equilibré, augmenté, lapiacien, autoencoding et split	0.76
Images masqué et split	0.73

Premièrement, comme indiqué dans la table 4, les images masquées donnent de moins bonnes performances que les images initiales, validant une seconde fois le test effectué pendant la phase d'augmentation des images. De plus, un fait important ici est les résultats

significativement moins bons pour les images auto-encodées, avec une perte d'accuracy de l'ordre de 5 % et aucun gain de temps de calcul. Cette approche de réduction des dimensions des images a donc été abandonnée car elle ne permettait ni des gains en temps de calcul ni en suppression du bruit.

Cependant, les étapes de suppression de doublons ainsi que d'augmentation d'images sont très intéressantes dans le cadre de la classification de radiologie pulmonaire, avec des valeurs d'accuracy pertinentes (modèle sans fine-tuning), mais aussi des valeurs de F1-score et de recall similaires à l'accuracy, démontrant le non-surapprentissage du modèle.

Il est à noter que l'approche par matrice de Laplacien ne semble pas pertinente, avec une perte de 1 % d'accuracy. Le jeu de données choisi pour l'apprentissage des modèles est donc celui après sur- et sous-échantillonnage, sans approche par matrice de Laplacien ni autoencodeur (correspondant dans notre nomenclature au jeu de données **V2V2 équilibré**).

Comparaison des modèles :

Pour évaluer les modèles produit différentes métriques sont utiliser leur application mathématique est démontrée ci-dessous :

- **Précision**, mesure la proportion de prédictions positives correctes parmi toutes les prédictions positives.

$$\text{Precision} = \frac{TP}{TP + FP}$$

où :

- TP = Vrais positifs (True Positives)
- FP = Faux positifs (False Positives)

- **Recall,** mesure la proportion de vrais positifs correctement identifiés parmi tous les exemples positifs réels.

$$\text{Recall} = \frac{TP}{TP + FN}$$

où :

- FN = Faux négatifs (False Negatives)

- **F1-score**, est la moyenne harmonique entre la précision et le rappel. Il permet de trouver un équilibre entre ces deux mesures.

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN}$$

- **Accuracy**, mesure la proportion de bonnes prédictions (positives et négatives) parmi l'ensemble des prédictions.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

où :

- TN = Vrais négatifs (True Negatives)

Durant l'optimisation des modèles proposés ci-dessus, et les retours sur la problématique métier discutée entre les collaborateurs de ce projet, il s'est avéré pertinent d'inclure dans l'entraînement des modèles un poids différent. En effet, la liberté à l'utilisateur de choisir le poids accordé à la classe COVID (allant de 1 à 3 fois plus important que les autres classes) durant l'entraînement du modèle a été mise en place. Ce choix découle de l'importance, dans le cadre d'une pandémie, de déterminer le plus possible les cas infectieux pour traiter et désengorger le secteur médical ciblé.

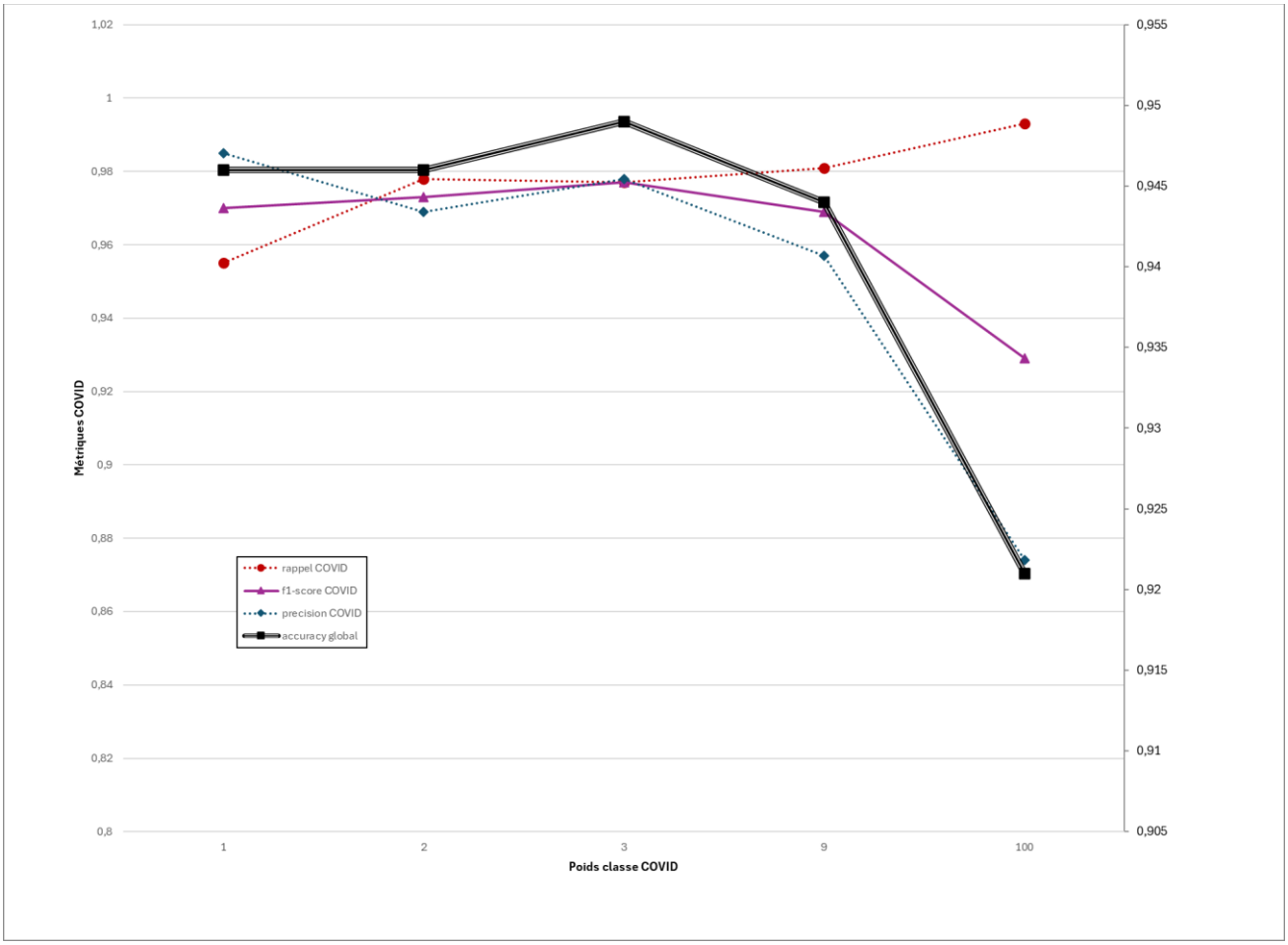


Figure 14 : Métriques mesurées sur la classe COVID en fonction du poid accordée à cette classe (Modèle EfficientNet).

Une approche exploratoire empirique a été lancée permettant de déterminer le poids accordé à la classe COVID afin de répondre à la problématique tout en accordant une importance à la performance et à la robustesse des modèles développés. La figure 14 montre donc le rappel, le F1-score et la précision de la classe COVID, ainsi que le F1-score global. Il se dégage au travers de ce graphique une légère amélioration des métriques pour un poids de 3, avec par la suite une chute brusque des métriques pour des poids supérieurs à 9. Cependant, une diminution du rappel de la classe COVID est visible pour un poids de 3. À noter que le rappel est croissant et ne décroche pas comme les autres métriques, ce qui est en accord avec un poids très important accordé à cette classe. Il est donc intéressant d'utiliser des poids allant de 1 à 3 dans le cadre du développement de modèles accordant une importance à la classe COVID tout en gardant des performances globales intéressantes.

Table 5 : Comparaison des métriques des différents modèles finaux.

Poids COVID	Dataset	Precision	Precision COVID	Recall	Recall COVID	F1 score	F1 score COVID	Paramètres entrainables
MobileNet								
1	V2V2 équilibré	93,9 %	97,6 %	93,6 %	94,4 %	93,7 %	96,0 %	2,659,076
2	V2V2 équilibré	94,1 %	98,7 %	93,7 %	93,3 %	93,8 %	95,9 %	2,659,076
3	V2V2 équilibré	94,1 %	96,6 %	93,8 %	95,8 %	93,8 %	96,2 %	2,659,076
EfficientNet								
1	V2V2 équilibré	94,6 %	99,0 %	94,5 %	94,8 %	94,5 %	96,9 %	741,124
2	V2V2 équilibré	94,9 %	97,7 %	94,9 %	97,3 %	94,9 %	97,5 %	741,124
3	V2V2 équilibré	94,7 %	97,9 %	94,7 %	96,6 %	94,7 %	97,2 %	741,124
ResNet								
1	V2V2 équilibré	94,2 %	97,0 %	94,2 %	96,3 %	94,2 %	96,7 %	8,144,388
2	V2V2 équilibré	93,7 %	98,5 %	93,6 %	93,3 %	93,6 %	95,8 %	8,144,388
3	V2V2 équilibré	93,6 %	98,0 %	93,4 %	92,3 %	93,4 %	95,0 %	8,144,388
VGG19								
1	V2V2 équilibré	94,1 %	96,9 %	94,1 %	96,0 %	94,1 %	96,5 %	2,361,860
2	V2V2 équilibré	93,7 %	96,7 %	93,7 %	96,1 %	93,7 %	96,4 %	2,361,860
3	V2V2 équilibré	93,4 %	96,2 %	93,4 %	95,8 %	93,4 %	96,0 %	2,361,860

Une fois les modèles développés, leurs performances spécifiques ont été déployées au travers d'un Streamlit. Après enregistrement des modèles en `.h5` ou en `.keras`, cette interface graphique permet de tester la portabilité de ces derniers. La figure 15 montre les métriques mesurées sur le jeu de données test après exportation des modèles.

Quatre modèles ont été produits avec des performances et une robustesse supérieure à 90 %, avec trois déclinaisons en fonction du poids accordé à la classe COVID pendant l'entraînement. On constate que dans sa globalité, le modèle EfficientNet est le plus performant, avec des valeurs de métriques spécifiques à la classe COVID très élevées (précision, rappel et F1-score), de même pour les métriques globales. Cependant, aucune relation croissante n'est observable entre la valeur du poids et les métriques spécifiques de la classe. À contrario, le modèle basé sur MobileNet, avec des performances et une robustesse sensiblement moins grande qu'EfficientNet, possède une relation croissante et démontre donc la pertinence de l'application de ce poids. MobileNet avec un poids de 2 pour l'entraînement de la classe COVID donne une précision sur cette classe de 98,68 %, et est donc à privilégier si le but est de la prédire avec rigueur.

Les modèle VGG19 et ResNet possède des métriques similaires à MobileNet sans relation croissante entre la valeur de poids et les métriques spécifiques à la classe COVID. Ce sont donc de bons modèles qui répondent eux aussi parfaitement à la problématique, mais les spécificités d'actions d'EfficientNet et MobileNet tend à les sélectionner en priorité.

Au regard de cette comparaison détaillée, deux combinaisons de modèles sont performantes :

- **EfficientNet Poids 2** : Modèle à utiliser pour une prédiction globale robuste.
- **MobileNet Poids 2** : Modèle avec des prédictions précises à 98,68 % pour la classe d'intérêt (COVID).

Les différents modèles développés permettent donc une plasticité d'utilisation en fonction des attentes des médecins et des patients. Dans leur globalité, ils permettent rapidement de donner un diagnostic de qualité.

Interprétation des résultats :

Plusieurs méthodes ont été utilisées pour permettre de comparer les résultats des modèles développés.

Courbe d'apprentissages

Dans un premier temps, les courbes d'apprentissage ont été étudiées. On s'intéresse en particulier à surveiller les symptômes de sur-apprentissage :

- Écart croissant entre la précision sur entraînement et celle sur validation
- Divergence entre les courbes de loss d'entraînement et de validation.

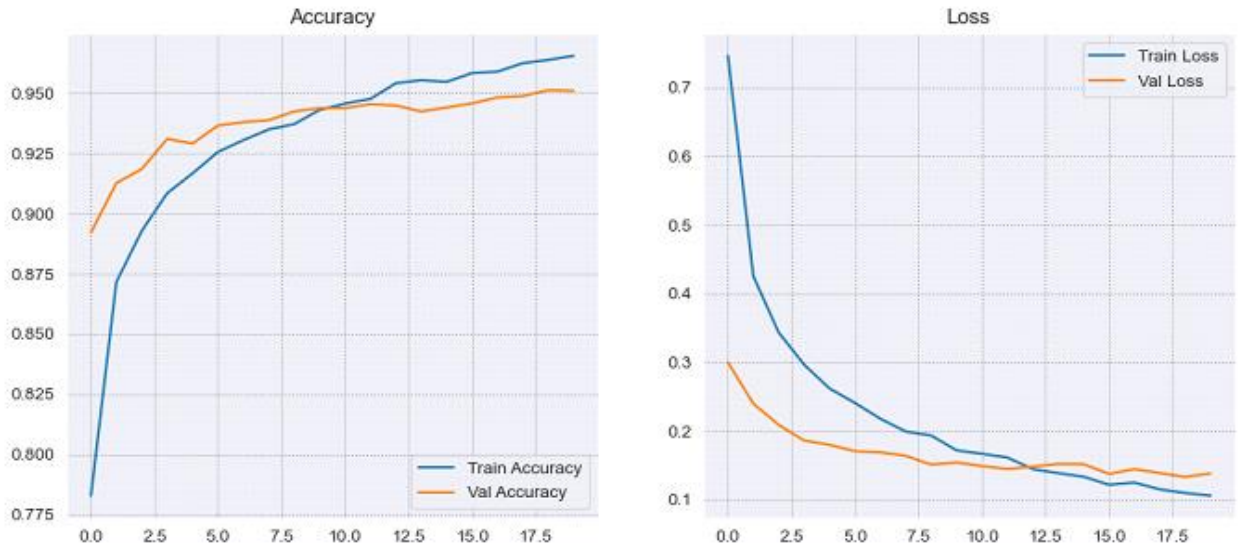


Figure 15 : Exemple de courbes d'apprentissage sur le modèle Efficient Net

Sur l'exemple montré dans la Figure 15, l'époque qui présente les meilleurs résultats tout en limitant le surapprentissage se situe entre l'époque 9 (où la précision d'entrainement dépasse celle de la validation) et l'époque 12 (où les fonctions de pertes se croisent)

GradCam

Un second outil utilisé pour mesurer les performances du modèle est la visualisation Grad-Cam.

Les premières couches d'un CNN détectent des motifs bas niveau : bords, textures, coins. Les couches intermédiaires : motifs un peu plus complexes. La dernière couche de convolution code des patterns complexes et discriminants pour la classification finale. C'est aussi la dernière couche du modèle à conserver la spatialité de l'image d'origine.

La méthode GradCam consiste à regarder le gradient de la dernière couche de convolution par rapport à la classe prédite. Elle met ainsi en évidence les zones ayant servi à la prédiction (Rouge) et les zones ayant peu/pas d'impact sur la prédiction.

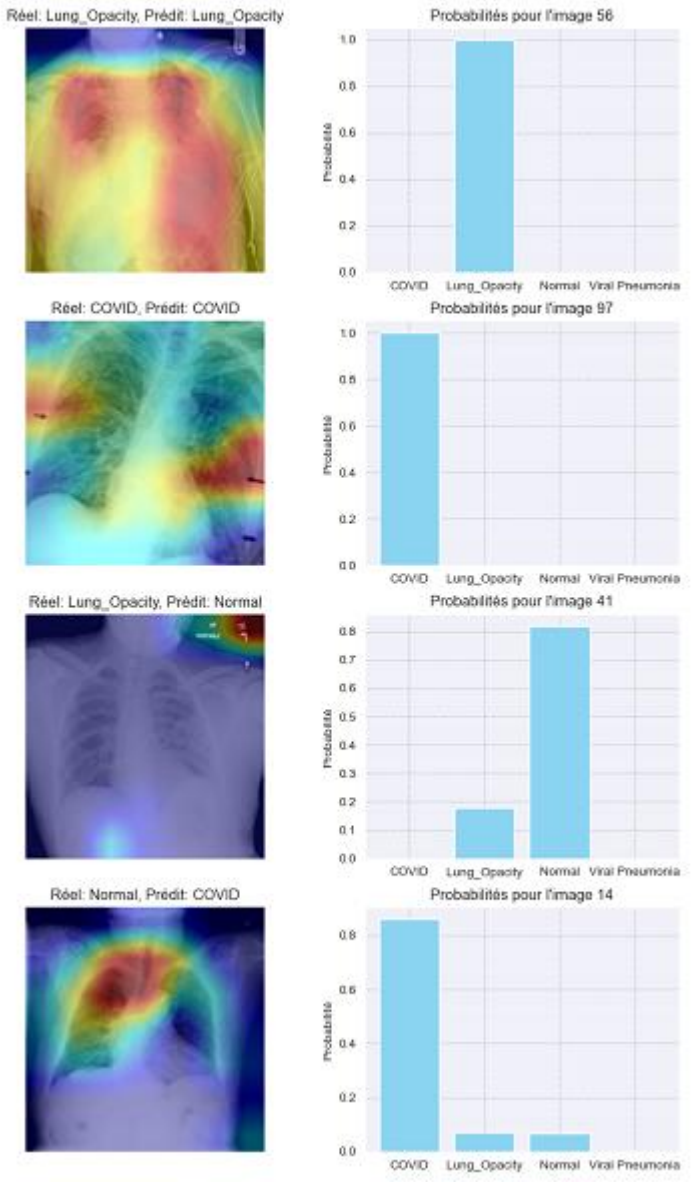


Figure 16 : Exemple de Visualisation GradCam

Sur l'exemple de la Figure 16 constitué de deux prédictions correctes et deux prédictions incorrectes, on constate que le modèle s'est globalement bien concentré sur la zone des poumons, sauf sur la troisième image où une annotation de radio a dirigé le modèle sur une prédiction Normal.

Ces visualisations sont précieuses pour s'assurer que le modèle prédit en s'intéressant aux endroits pertinents de l'image.

Indice de confiances

Enfin pour pousser l'analyse des résultats au-delà du simple rapport de classification, les probabilités softmax en sortie de la dernière couche dense ont été visualisées. En effet, nous cherchons aussi à évaluer avec quel niveau de confiance le modèle effectue ses prédictions. Le résultat souhaité est qu'il soit juste avec la confiance la plus élevée, et incorrecte avec la probabilité la plus élevée. Pour cela nous avons tracé des histogrammes de répartition des probabilités :

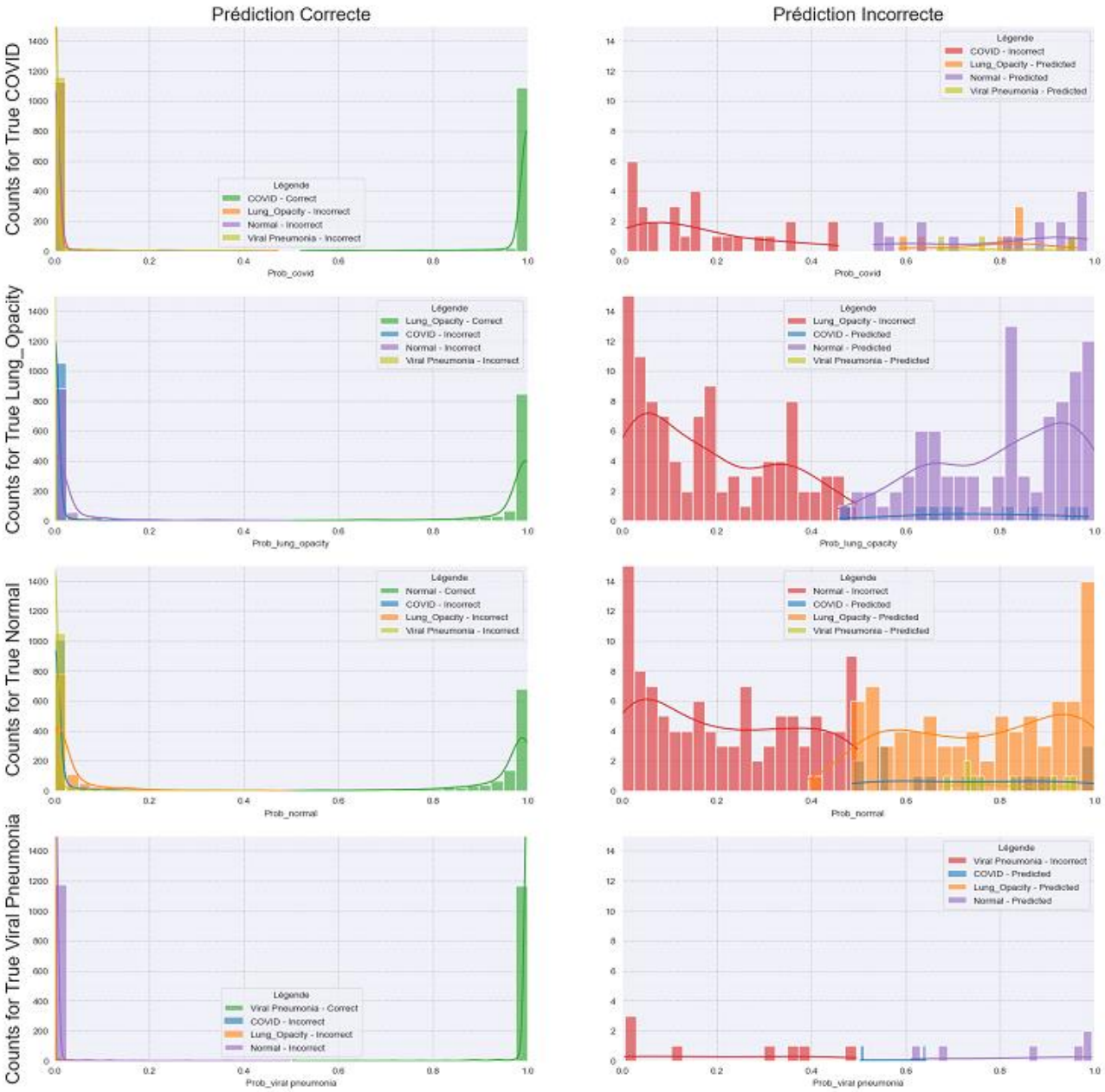


Figure 17: Prédictions faites par le modèle EfficientNet sur les quatre classes par images du jeu de données test.

Cela montre que lorsque les prédictions sont correctes, elles le sont avec un très fort indice de confiance (largement supérieure à 0.9)

En revanche, quand les prédictions sont incorrectes, les résultats sont plus confus :

- Pour la classe COVID, les classes Normal et Lung Opacity semblent équitablement être prédites, parfois avec un indice de confiance très élevé (> 0.8) , les probabilités pour la classe COVID sont relativement faibles (une majorité en dessous de 0.3)
- Pour la classe Lung Opacity et la classe Normal, les résultats sont extrêmement confus (comme le montre déjà la matrice de confusion) avec des prédictions quasi-certaines incorrectes, mais aussi des prédictions quasi-juste (zone 0,4 – 0,6 sur les graphiques)
- Pour la classe Viral Pneumonia, déjà très bien détectée par le modèle, il n'y a pas de tendance particulière à observer.

Conclusions tirées :

Ce projet s'inscrit dans le cadre du développement d'un outil d'aide au diagnostic médical par vision par ordinateur, en s'appuyant sur des images de radiographies pulmonaires classées en quatre catégories : Normal, Lung Opacity, Viral Pneumonia et COVID-19. Le jeu de données utilisé, mis à disposition sur Kaggle, a été constitué grâce à une collaboration internationale entre chercheurs du Qatar, de Dhaka, de Malaisie et du Pakistan, et documenté dans deux publications scientifiques. Ce jeu de données présente des annotations fiables, des masques pulmonaires pour l'isolation des zones pertinentes, mais également un déséquilibre important entre les classes, nécessitant des techniques de rééquilibrage par sur- et sous-échantillonnage.

Une phase d'exploration des données a permis d'évaluer visuellement la diversité des sources d'images ainsi que plusieurs variabless d'images (luminosité, contraste, aire des poumons, etc.), afin d'identifier les limites potentielles à adresser dans la phase de prétraitement. À ce titre, plusieurs techniques ont été testées empiriquement, telles que la suppression des doublons, l'augmentation d'images, l'application de filtres de Laplacien, ou encore l'autoencodage pour la réduction de dimensions. Ces dernières se sont révélées peu efficaces,

voire contre-productives, au contraire de l'augmentation d'images qui a permis d'améliorer la robustesse sans surapprentissage.

Dans une logique d'optimisation du temps de calcul et de la performance, une approche de transfert learning a été privilégiée. Des modèles préentraînés comme VGG19 et ResNet ont été adaptés à la tâche de classification via un fine-tuning partiel, avec notamment l'ouverture de certaines couches et l'ajout de couches spécifiques (Dropout, Dense + Softmax). Les performances ont été évaluées sur un split Train/Test/Validation indépendant, à l'aide d'indicateurs classiques comme la precision, le F1-score, le recall, et via des outils d'interprétabilité comme Grad-CAM, permettant de visualiser les zones d'attention du modèle sur chaque image.

Enfin, l'évaluation comparative des modèles a permis de sélectionner des configurations optimales reposant sur un jeu de données nettoyé, équilibré, et sans autoencodeur, avec une architecture EfficientNet et MobileNet fine-tunée, qui offre un bon compromis entre robustesse, précision, et interprétabilité.

Difficultés rencontrées lors du projet :

Plusieurs difficultés ont été rencontrées tout au long du cycle de vie du projet. Cependant, elles sont toutes de nature organisationnelle. En effet, une utilisation plus poussée et automatique du répertoire GitHub aurait permis une plus grande coordination, alors que la technique privilégiée durant ce projet a été les échanges de documents et de scripts par l'interface de Slack. De plus, une coordination dans les environnements de travail en amont du développement individuel de scripts aurait permis une mise en commun des scripts fluide et intuitive, de même pour l'enregistrement des variables et des paramètres des modèles.

D'une manière générale, nous avons rencontré différents problèmes dus à notre manque d'expérience en termes de projet de Deep Learning.

Voici les principaux, ainsi que les leçons tirées de ces expériences :

1. Impact de l'absence de venv sur le projet :

Nous avons rencontré des problèmes dû à des absences d'environnement virtuel ou à leur incompatibilité mutuelle au long de ce projet (comme la création de dossiers corrompus de versions de Tensorflow, résolue par une suppression manuelle du Disque dur de toute la librairie et de ses dépendances). Cela a permis de résoudre les alertes VSCode de fonctions et de bibliothèques non-trouvées et d'harmoniser les différentes versions de nos librairies entre collègues.

En effet, sans venv, toutes les bibliothèques Python sont installées globalement sur la machine locale. Si on travaille sur plusieurs projets, chacun avec des exigences de versions de

bibliothèques différentes (différentes versions de Tensorflow, NumPy, Keras, etc.), l'installation globale tentera de satisfaire toutes ces exigences. Cela conduit inévitablement à des conflits où une version installée écrase une autre, ou bien encore où une version une version d'une librairie est incompatible avec une autre. Quand on exécute le code, Python importe la version globalement disponible, qui n'est pas forcément celle attendue par le projet.

Le code peut crasher avec des ImportError, AttributeError, ou des comportements inattendus, car des fonctions peuvent manquer ou se comporter différemment. Cela donne l'impression que la bibliothèque est corrompue ou mal installée, alors qu'il s'agit d'un problème de version. Les deux problèmes se superposant par la suite, on obtient un problème multi-causal beaucoup plus dur à résoudre.

Python et le système d'exploitation doivent savoir où trouver les exécutables et les bibliothèques. Sans venv, le PATH du système peut devenir encombré et pointer vers des versions inattendues de Python ou de modules (comme un dossier ~ensorflow à la place de Tensorflow, par exemple). Cela rend la reproduction des bugs difficile, car l'environnement n'est pas isolé et contrôlé. Le débogage devient un vrai cauchemar, car les erreurs peuvent être dues à des interactions inattendues entre des versions de bibliothèques disparates.

Leçon à retenir : TOUJOURS utiliser **un environnement virtuel par projet** pour contrôler l'environnement de travail, et donc être certains de pouvoir débuguer de manière simple. De plus, cela élimine des causes de crash de l'algorithme testé.

2, Impact de l'absence de standardisation des protocoles communs :

Les causes de ce deuxième problème rencontré sont intrinsèquement liées au premier.

Tout au long de ce projet nous avons, vous l'avez vu, établi certains standards pour travailler de façon plus organisée (utilisation de Datasets communs, création d'un tableau de métriques communes, etc.).

Pour autant, les résultats de nos travaux ne sont néanmoins pas pleinement comparables : tout simplement en raison de notre manque d'expérience, qui nous a fait établir des standards communs pour mieux communiquer, certes, mais où nous avons fait preuve d'une rigueur de coordination clairement insuffisante : nous n'avons pas défini de versions communes des librairies utilisées ni d'environnement unifié (cf. Impact de l'absence de venv), nous n'avons le nombre d'epochs à utiliser lors de l'entraînement, et nous n'avons pas défini de format d'enregistrement unifié (H5 ? Keras ?).

Une réflexion plus approfondie en amont du projet, et surtout en amont de l'étape de modélisation, nous aurait permis d'éviter ces désagréments et d'autres, et donc de beaucoup mieux synchroniser nos résultats

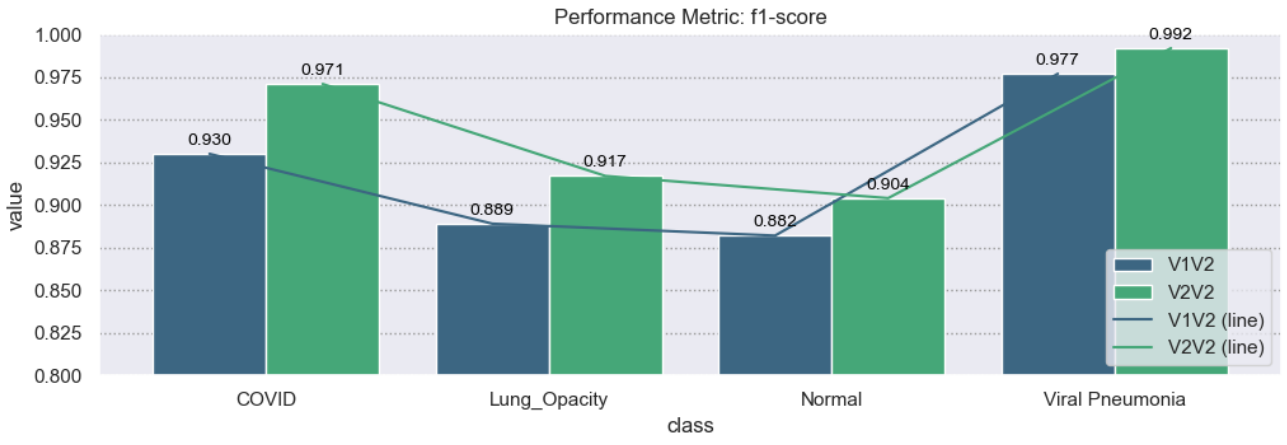
Leçon à retenir : dans le cadre d'un travail en équipe, **TOUJOURS prendre le temps de réfléchir à un cadre et à des protocoles communs,** en particulier avant de tenter une quelconque entrée dans l'étape de modélisation d'un projet de ML/Deep Learning. Cela entraînera non seulement un gain d'efficacité considérable, mais également de s'assurer d'avoir des résultats comparables en utilisant des protocoles beaucoup plus rigoureux, qui permettront donc aux conclusions tirées des comparaisons d'avoir d'avantage de valeur, en termes d'interprétabilité.

3. Fuite de données due à l'augmentation d'images :

Nous avons vu que dans notre preprocessing, nous avons eu recours à l'augmentation d'images sur les classes «COVID» et «Viral Pneumonia» afin de combler le déséquilibre des différentes classes du Dataset.

Cependant, ayant appliqué ces transformations avant la séparation entre échantillons de training, de test et de validation, il est tout de même possible – et même hautement probable, vu le nombre de données – que des images dans l'échantillon de test de ces deux catégories puissent trouver leurs originales dans les sous-catégories du dossier d'entraînement, ce qui cause bien évidemment de la fuite de données (bien qu'elle ne soit pas visible ici).

Par exemple, on peut supposer que la différence de F1-score, sur le modèle EfficientNet avec un poids de 2 sur la classe COVID, entre le Dataset V1V2 purgé des doublons et le Dataset V2V2 (comme illustré par le graphique ci-dessous pour les différentes classes) aurait pu être plus importante sans ce problème de fuite de données :



Leçon à retenir : TOUJOURS faire la séparation entre données d'entraînement et données de test/validation AVANT d'y appliquer une quelconque forme de preprocessing prenant en compte l'ensemble des données de l'échantillon considéré.

Perspectives :

Le développement de modèles de classification et d'aide au diagnostic pulmonaire a donné des résultats très pertinents. En effet, les résultats de précision et de F1-score obtenus sont très intéressants et répondent parfaitement aux attentes métier. Cependant, dans une optique de déploiement des modèles produits à grande échelle, certains points sont à prévoir pour alimenter et améliorer les modèles.

À titre d'exemple, l'intégration de notions de *reinforcement learning* permettant l'adaptation dynamique du modèle en fonction d'un retour d'information (*reward*), de *Human-in-the-loop RL* avec des notations de validation par patients et médecins. Mais aussi la possibilité pour les utilisateurs d'augmenter la base de données, validée par la suite par des spécialistes.

L'intégration de ces notions permettra de dynamiser l'apprentissage sans réentraîner de nouveaux modèles (exemple du fonctionnement d'Ecotaxa : https://ecotaxa.obs-vlfr.fr/).

En présence de plus de temps, plusieurs possibilités d'approfondissement auraient été également possibles :

1. Création d'échantillons de cross-validation des modèles à partir du dataset de base (puis réapplication du preprocessing) ;

2. Exploration de poids différents pour privilégier les 3 autres classes et comparer les résultats ;

3. Utilisation et comparaison de différents types de gradients et d'algorithmes d'optimisation adaptés aux CNNs (SGD avec Momentum pour être moins sensible aux minima locaux, AdaGrad pour prendre en compte les données éparses, etc.).

Bibliographie :

1. Can AI Help in Screening Viral and COVID-19 Pneumonia? | IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/document/9144185.

2. Pondaven-Letourmy, S., Alvin, F., Boumghit, Y. & Simon, F. Comment réaliser un prélèvement rhinopharyngé chez l'adulte et l'enfant en période de la pandémie de la maladie COVID-19. *Annales Franc?aises D'Oto-Rhino-Laryngologie et De Pathologie Cervico-Faciale* **137**, 301–303 (2020).

3. Tahir, A. M. *et al.* COVID-19 infection localization and severity grading from chest X-ray images. *Computers in Biology and Medicine* **139**, 105002 (2021).

4. Covid-19-lettre-info-n10-coronavirus-sars-cov-2-rt-pcr.pdf.

5. Tan, M. & Le, Q. V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Preprint at https://doi.org/10.48550/arXiv.1905.11946 (2020).

6. Rahman, T. *et al.* Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images. *Computers in Biology and Medicine* **132**, 104319 (2021).

7. Fonctionnement et fiabilité des tests RT-PCR pour la détection du SARS-CoV-2. *Institut Pasteur* https://www.pasteur.fr/fr/espace-presse/documents-presse/fonctionnement-fiabilite-tests-rt-pcr-detection-du-sars-cov-2 (2020).

8. Les fondamentaux - Chapitre 25 - Imagerie thoracique - CNP MN. https://www.cnp-mn.fr/les-fondamentaux-chapitre-25-imagerie-thoracique/.

9. Medical Image Segmentation Review: The Success of U-Net | IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/10643318.

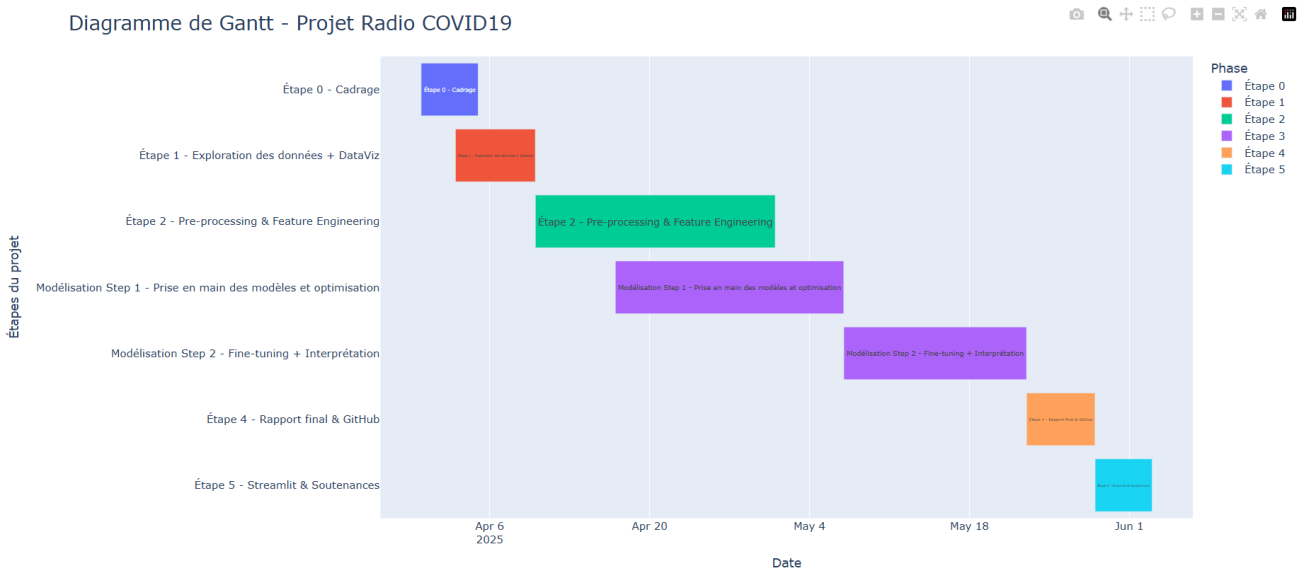
10. QaTa-COV19 Dataset. https://www.kaggle.com/datasets/aysendegerli/qatacov19-dataset.

11. Tests de dépistage Covid-19 : quel prix ? Quel remboursement Sécu ? *Previssima* https://www.previssima.fr/actualite/tests-de-depistage-covid-19-quel-prix-quel-remboursement-secu.html (2020).

12. Boesch, G. Very Deep Convolutional Networks (VGG) Essential Guide. *viso.ai* https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/ (2021).

Annexes :

Annexe 1 : Déroulement du projet.



Annexe 2 : Script Github

Annexe 3 : Streamlit

Annexe 4 : arboressance du projet

Annexe 5 : Exemple d'une image du jeu de données, de son masque et de la superposition des deux.

Annexe 6 : Tests statistiques sur les variables exploratoires

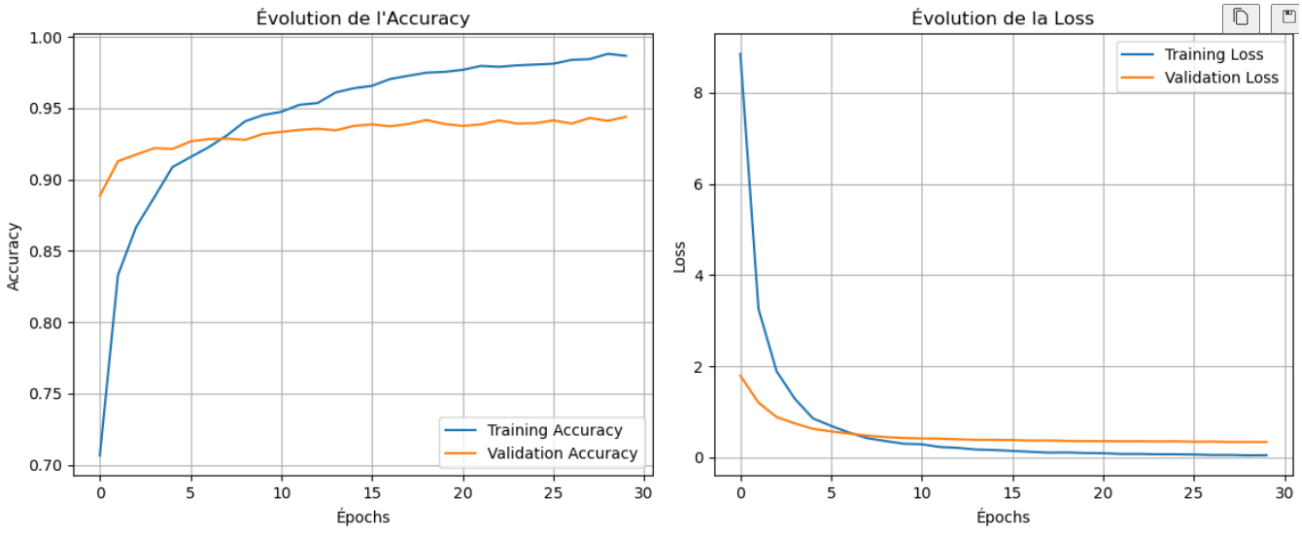
L'homogénéité des variances ainsi que leur covariance ont été tester en amont.

	feature	PR(>F)	F
0	cont_RMS	9.346916e-273	432.543849
1	luminiosite	4.007484e-182	286.197361
2	mask_area	1.214369e-304	484.711387
3	masked_lum	0.000000e+00	842.676297
4	masked_cont	4.001209e-210	331.109165

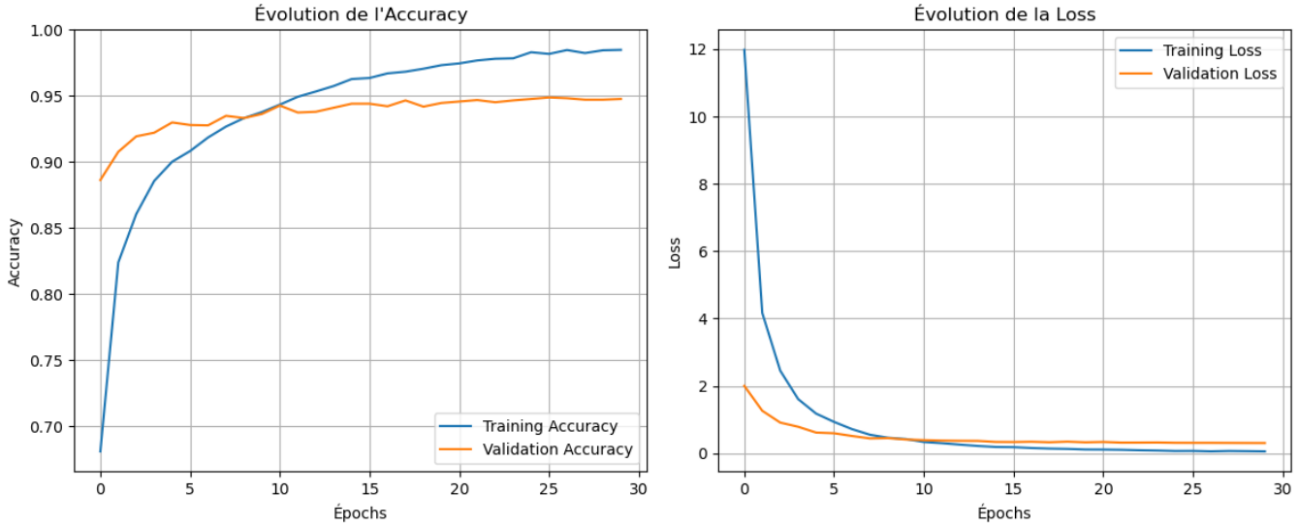
```
Post-hoc Tukey HSD sur l'intensité :
       Multiple Comparison of Means - Tukey HSD, FWER=0.05
=============================================================================
   group1         group2     meandiff p-adj  lower    upper   reject
-----------------------------------------------------------------------------
       COVID    Lung_Opacity   -15.51   0.0 -16.6021 -14.4178   True
       COVID          Normal -18.8126   0.0 -19.9047 -17.7205   True
       COVID Viral_Pneumonia  -2.8009   0.0   -3.893  -1.7087   True
Lung_Opacity          Normal  -3.3027   0.0  -4.3948  -2.2105   True
Lung_Opacity Viral_Pneumonia  12.7091   0.0   11.617  13.8012   True
      Normal Viral_Pneumonia  16.0118   0.0  14.9197  17.1039   True
-----------------------------------------------------------------------------

Post-hoc Tukey HSD sur le contraste :
       Multiple Comparison of Means - Tukey HSD, FWER=0.05
=============================================================================
   group1         group2     meandiff p-adj   lower   upper  reject
-----------------------------------------------------------------------------
       COVID    Lung_Opacity   4.4556    0.0  3.9241  4.9871   True
       COVID          Normal   0.4329 0.1555 -0.0986  0.9645  False
       COVID Viral_Pneumonia   9.2739    0.0  8.7424  9.8054   True
Lung_Opacity          Normal  -4.0227    0.0 -4.5542 -3.4911   True
Lung_Opacity Viral_Pneumonia   4.8183    0.0  4.2868  5.3498   True
      Normal Viral_Pneumonia    8.841    0.0  8.3094  9.3725   True
-----------------------------------------------------------------------------
```

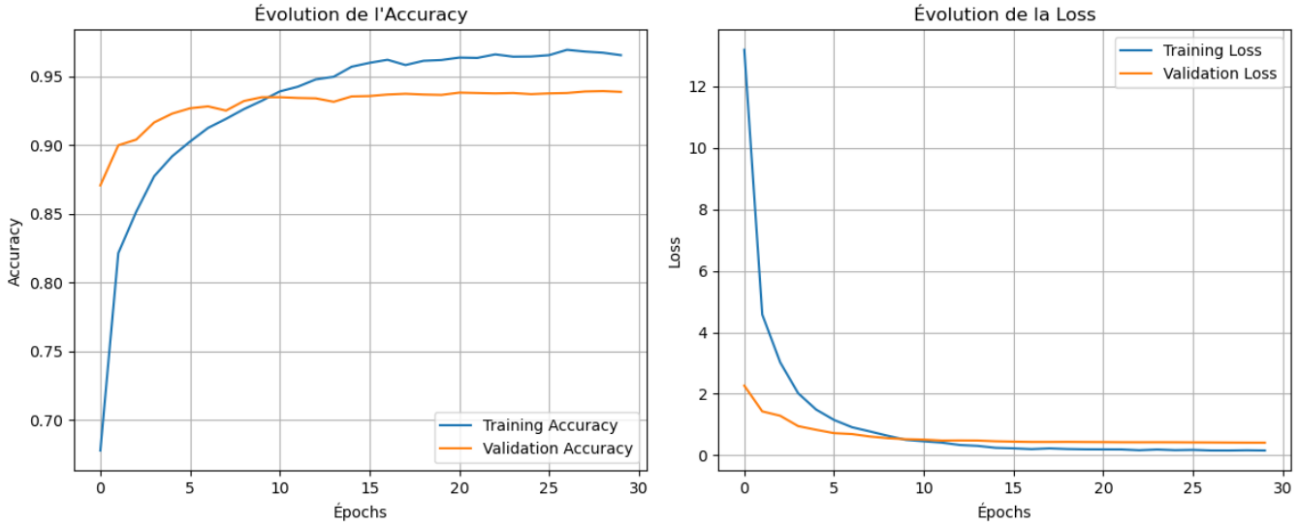
Annexe 7 : Entrainement du modèle VGG19 avec un poids pour la classe *COVID* de 1



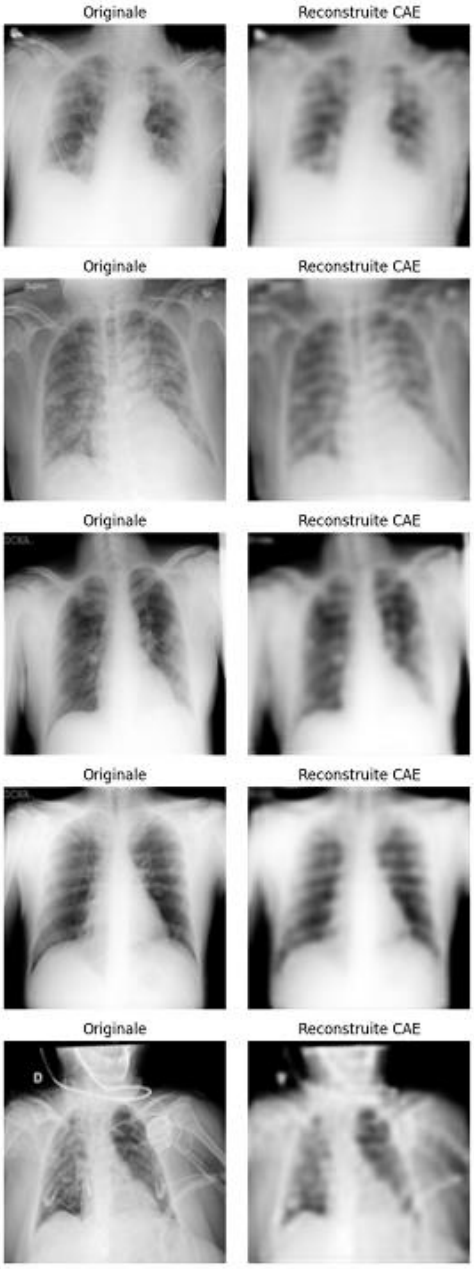
Annexe 8 : Entrainement du modèle VGG19 avec un poids pour la classe *COVID* de 2



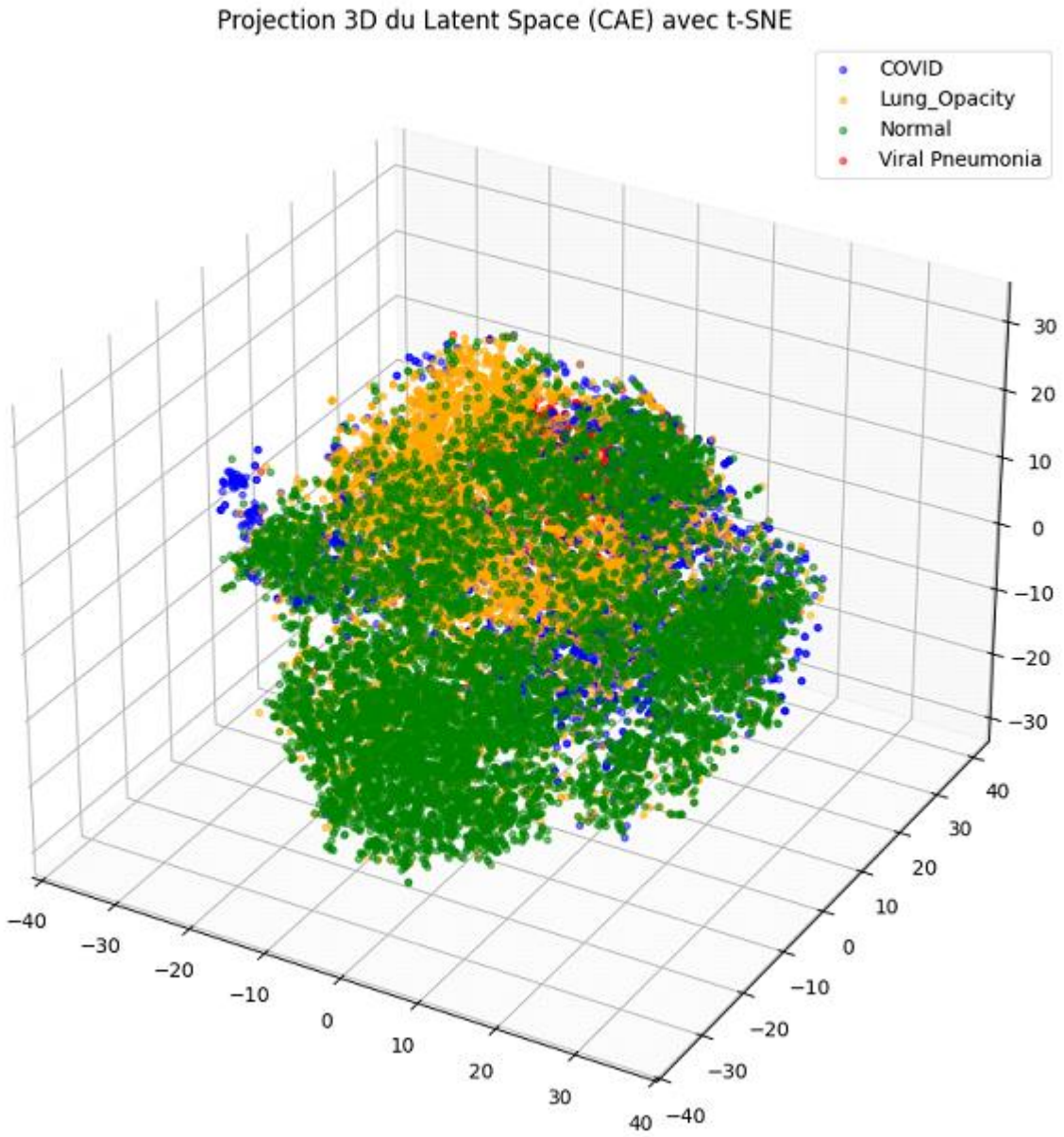
Annexe 9 : Entrainement du modèle VGG19 avec un poids pour la classe *COVID* de 3



Annexe 10 : reconstruction d'images après autoencodeur covolutionnel :


Originale
Reconstruite CAE
Originale
Reconstruite CAE
Originale
Reconstruite CAE
Originale
Reconstruite CAE
Originale
Reconstruite CAE

Annexe 11 :


Projection 3D du Latent Space (CAE) avec t-SNE
COVID
Lung_Opacity
Normal
Viral Pneumonia
−40
−30
−20
−10
0
10
20
30
40
30
20
10
0
−10
−20
−30
40
30
20
10
0
−10
−20
−30
−40