

Traitemet automatique d'électrogrammes transmis par des dispositifs électroniques cardiaques implantables

Organisme d'accueil : IHU Liryc

Tuteur de stage : Rémi Dubois

Encadrante universitaire : Marie Chavent

*Stage effectué du
20/02/2023 au 20/07/2023*

Table des matières :

Liste des symboles	3
1 Remerciements :	4
2. Présentation de l'organisme d'accueil	5
2.1 Recherche	5
2.2 Innovation	7
2.3 Soins	7
2.4 Formation	8
3. Travaux effectués pendant le stage	9
3.1 Contexte du stage et problématique abordée	9
3.2 Données à disposition	11
3.3 Organisation de la base de données	13
3.4 Pré-traitement des données	14
3.5 Modèle de réseau de neurones	14
3.5.1 Réseau neuronal résiduel	15
3.5.2 Hyperparamètres du modèle	17
3.5.3 Critère d'évaluation d'un modèle	18
3.5.4 Méthode de sélection d'une architecture et des hyperparamètres	19
3.5.5 Premières architectures	20
3.5.6 Architecture ResNet et résultats	22
3.5.7 Analyse des enregistrements mal classés	30
3.6 Détection hors distribution par incertitudes de prédiction	32
3.6.1 Application sur le jeu de données MNIST	33
3.6.2 Application sur le jeu de données des EGM	36
4. Conclusion	39
5. Références	40
6. Annexes	42

Liste des symboles

CHU : Centre Hospitalo-Universitaire

CIED : Cardiac Implantable Electronic Device - Dispositif électronique cardiaque implantable

ECG : Électrocardiogramme

EGM : Électrogramme

FA : Fibrillation auriculaire

ICD : Implantable Cardiac Device - Dispositif cardiaque implantable

IRM : Imagerie par Résonance Magnétique

IHU : Institut Hospitalo-Universitaire

1 Remerciements :

Avant toute chose, je tenais à exprimer mes remerciements et toute ma gratitude envers toutes les personnes qui ont rendu ce stage possible, à toutes celles et ceux qui m'ont beaucoup appris durant ce stage et qui ont contribué à en faire une expérience gratifiante tant sur le plan professionnelle que sur le plan humain.

Je remercie donc chaleureusement **Rémi Dubois**, mon maître de stage qui m'a formé et accompagné tout au long de cette expérience professionnelle avec beaucoup de patience et de pédagogie.

Je tiens également à remercier tous mes collègues de l'équipe Traitement du signal pour leur sympathie, leur esprit d'équipe et leurs conseils, grâce auxquels j'ai pu mener à bien ce travail. Ensuite, en ce qui concerne l'institut, je remercie l'ensemble de ses employés pour leur accueil chaleureux et leur bienveillance.

Je tiens à exprimer ma gratitude envers mon encadrante universitaire **Marie Chavent**, ainsi qu'à tout le corps professoral de l'Université de Bordeaux pour la qualité de leurs enseignements, leurs efforts ainsi que leur implication pour le bon développement de leurs étudiants dans le milieu universitaire et pour leur bonne insertion professionnelle.

Enfin, que tous ceux ayant contribué de près ou de loin à l'accomplissement de ce travail trouvent l'expression de mes plus sincères remerciements.

2. Présentation de l'organisme d'accueil

L’Institut Hospitalo-Universitaire (IHU) Liryc est un institut de recherche dédié aux maladies du rythme cardiaque et à la prise en charge de ces maladies. Il fut créé en 2012 grâce au Programme d’Investissements d’Avenir mis en place par l’Etat français piloté par le Secrétariat général pour l’investissement. Ces plans d’investissement permettent de financer et de soutenir l’innovation et l’excellence scientifique.^[1]

Les équipes du Liryc concentrent leurs efforts sur 3 principales maladies du rythme cardiaque : la fibrillation auriculaire, la fibrillation ventriculaire et l’insuffisance cardiaque. Ces dernières entraînent des dysfonctionnement électriques et mécaniques du cœur.

L’institut regroupe en son sein des expertises nationales et internationales, engagées dans 4 missions :

- la recherche, pour mieux comprendre et explorer les mécanismes à l’origine des maladies du rythme cardiaque ;
- l’innovation, pour développer et inventer les thérapies de demain ;
- les soins, pour continuer à améliorer la prise en charge des patients et sauver des vies ;
- la formation, pour transmettre largement les avancées des équipes de recherche et clinique et former les professionnels, à l’échelle internationale.

2.1 Recherche

La création du Liryc en 2012 a permis à des médecins et chercheurs de mieux se pencher sur les problématiques d’arythmie et de mortalité cardiaques, domaine de la cardiologie jusqu’ici largement sous-exploré.

L’ensemble des chercheurs et doctorants du Liryc se décompose en plusieurs pôles, à savoir : le pôle modélisation, le pôle technologie pour la santé, le pôle physiopathologie, le pôle image ainsi que le pôle clinique.^[2]

Le pôle modélisation conçoit et analyse des modèles numériques de l’activité électrique du cœur. Ces modèles sont constitués d’équations dont le calcul de solutions approchées nécessitent des algorithmes de calcul de haute performance ainsi que des ressources informatiques importantes. Ils peuvent être personnalisés avec des données expérimentales ou cliniques. L’équipe de ce pôle s’intéresse aux questions que posent les équations de fonctionnement électrique ou mécanique du cœur, à la modélisation de celle-ci, à la production d’ECG ou de biomarqueurs synthétiques ou encore à la résolution de problème inverse de l’imagerie électrocardiographique. Ce dernier consiste à reconstruire l’activité électrique à la surface du cœur à partir de celle enregistrée à la surface du thorax. Les membres de ce pôle mènent ainsi des travaux de recherche en mathématiques, informatique et biomédical pour mieux comprendre les mécanismes de troubles du rythme

cardiaque et améliorer les méthodes de défibrillations cardiaques (défibriller sous le seuil de douleur).

Le pôle technologie pour la santé invente et développe des outils physiques et numériques pour caractériser et analyser la structure cardiaque et son activité électrique. Il se divise en 3 équipes : imagerie, traitement du signal et science de données multimodales. L'équipe imagerie se focalise sur la caractérisation de la structure du myocarde à l'aide d'instruments d'imagerie de pointe tel l'IRM de très haut champ magnétique. Elle développe des outils pour analyser la structure cardiaque simultanément aux signaux électriques. Elle intervient également dans le développement de l'imagerie interventionnelle et de la thermométrie qui permettent de guider le geste du chirurgien en temps réel.

L'équipe traitement du signal, spécialisée en instrumentation et traitement du signal, se concentre sur la cartographie de l'activité électrique du cœur. Elle développe de nouveaux outils à partir de signaux électriques permettant de mettre en évidence les mécanismes électrophysiologiques des arythmies cardiaques, de réaliser des diagnostics précoce, de comprendre la maladie et ainsi de guider la thérapie. L'équipe travaille aussi sur des outils pédagogiques tel le simulateur SIMRIC. Ce dernier permettrait de reproduire l'activité électrique d'un cœur dans un scénario d'arythmie cardiaque, offrant des applications de recherche et de formation.

Enfin, l'équipe science des données multimodales se charge quant à elle de développer des outils de diagnostic et de guidage thérapeutique à partir des données recueillies chez des patients souffrant d'arythmie dans le cadre du soin. Elle combine des données d'imagerie, de cartographie, de signaux électrophysiologiques enregistrées ou simulées et l'intelligence artificielle (IA) grâce notamment au logiciel MUSIC, développé par l'IHU Liryc et l'équipe Inria Epione. Ceci à visée pronostique, diagnostique ou au guidage des interventions sur les arythmies atriales et ventriculaires par l'imagerie.

Le pôle physiopathologie dirige l'ensemble de ses recherches vers l'étude des mécanismes des arythmies cardiaques au niveau des tissus et des cellules, en particulier de la fibrillation auriculaire, ventriculaire ainsi que la tachycardie ventriculaire. Il regroupe des chercheurs spécialistes de l'électrophysiologie et de la bioénergie cardiaque. Ceux-ci développent des modèles de pathologies cardiaques pour des études longitudinales sur les mécanismes des troubles du rythme. Ils emploient également des techniques expérimentales telles la microtomographie par rayon X pour caractériser la microstructure cardiaque, le cœur isolé-perfusé pour des études ex vivo de la fonction cardiaque, ou encore de la cartographie optique pour obtenir une visualisation haute résolution de l'activité électrique du cœur. Enfin, dans le cadre des programmes "Cadence" et "Harmonica", ce pôle a la possibilité de tester ses modèles sur des tissus humains grâce au don d'organes pour la recherche.

Le pôle clinique conduit l'ensemble des études scientifiques observationnelles ou thérapeutiques. L'équipe de ce pôle travaille en étroite collaboration avec les médecins, les patients, les chercheurs (équipe de recherche fondamentale) et les professionnels de la recherche clinique. Leur objectif est de mieux comprendre et traiter les maladies du rythme cardiaque, validant ou non au passage certaines hypothèses scientifiques. À l'aide de différents protocoles expérimentaux rigoureux, et dans le respect de la réglementation en vigueur, ils observent l'efficience des traitements et des thérapies innovantes sur des patients volontaires.

Les chercheurs et médecins de Liryc sont à l'origine d'avancées scientifiques majeures dans les domaines des fibrillations atriales, ventriculaires et la resynchronisation cardiaque qui ont conduit à de nouvelles stratégies thérapeutiques mises en œuvre dans le monde entier. La force du programme de recherche de Liryc réside d'une part dans le fait que les axes de recherche émergent directement des besoins observés en clinique auprès des patients et d'autre part, sur le choix d'une approche hyper focalisée sur les maladies du rythme cardiaque. Le programme tire le meilleur de plateformes de recherche à la pointe de l'innovation et de collaborations internationales ambitieuses.

2.2 Innovation

L'IHU Liryc a pour vocation de transférer la connaissance acquise pour développer des technologies innovantes, pour le diagnostic ou le traitement des maladies du rythme cardiaque. Il mène une recherche d'excellence qui a pour ambition de servir de socle à la mise au point de nouveaux traitements tout en mettant à disposition les connaissances et les compétences de recherche.^[3] Sa politique de valorisation des résultats de recherche va permettre d'accompagner une découverte, une technologie ou une compétence pour la transformer en produit nouveau sur le marché, au bénéfice des patients du monde entier. Les innovations qu'il développe concernent principalement les cathéters d'électrophysiologie et leur optimisation pour de l'intervention et de la formation clinique, ainsi que les logiciels basés sur l'imagerie utilisés pendant les interventions. Les objectifs de l'institut sont de développer des outils de diagnostics et de soins moins invasifs et plus précis, d'identifier et de valider de nouvelles cibles pharmaceutiques ou encore d'accompagner les chercheurs dans la création de start-up vouées à porter les développements futurs.

La politique d'innovation de l'IHU s'appuie sur une stratégie définie collégialement, dont ses axes sont fixés par un comité stratégique. Le Liryc favorise, par ses infrastructures uniques, les relations entre les équipes de recherche, les équipes cliniques et différents acteurs économiques. Il s'entoure ainsi de partenaires industriels et financiers qui contribuent au développement de ces inventions par leurs investissements financiers. Parmi ces derniers, on retrouve des industriels du dispositif médical, du médicament ou de l'imagerie, des grands groupes ou encore des start-up. En quelques chiffres, en 2023, l'IHU a développé plus de 12 logiciels et a déposé 24 brevets depuis sa création en 2012.

2.3 Soins

Aucun patient n'est pris directement en charge au sein du Liryc.^[4] En effet, l'IHU exerce ses activités de recherche au sein d'une organisation médecins-chercheurs-patients. Celle-ci est intégrée sur les sites de l'hôpital cardiologique Haut-Lévêque, du CHU de Bordeaux, formant ainsi un continuum entre ces acteurs.

Les services cliniques apportent une expertise indispensable au diagnostic et à la prise en charge des patients atteints de maladies du rythme cardiaque. Ils représentent à la fois le point de départ et d'aboutissement des recherches.

Le savoir-faire des équipes médicales et paramédicales affiliées étroitement, liées à Liryc, permet d'assurer l'intégralité des étapes de la prise en charge du patient. Ces équipes disposent :

- d'un plateau de consultation mutualisé en lien étroit avec les centres d'exploration non invasive (échographie, ECG, épreuves d'effort, etc.) ;
- d'un service d'accueil des urgences cardiaques et d'une Unité de soins intensifs cardiaques ;
- d'un plateau technique de cardiologie interventionnelle à Haut-Lévêque ainsi qu'un secteur d'exploration invasive à Saint-André ;
- de blocs opératoires de chirurgie, cardiaque et thoracique, en cohabitation avec les unités de réanimation et de surveillance continue.

2.4 Formation

Le Liryc est le premier centre de recherche, d'enseignement et de soins sur les maladies du rythme cardiaque.^[5] Il s'engage dans l'accompagnement des futurs chercheurs et médecins. Dans l'environnement de recherche du Liryc, les jeunes chercheurs peuvent mener des projets de recherche originaux grâce à une équipe pédagogique comptant dans ses rangs des cliniciens, des chercheurs ainsi que des experts internationaux. L'équipe propose une coopération renforcée avec les milieux professionnels et les acteurs socio-économiques. Des liens entre le monde de la recherche et les laboratoires internationaux, les industriels et les entreprises du secteur sont ainsi établis, assurant aux jeunes chercheurs une insertion professionnelle sereine.

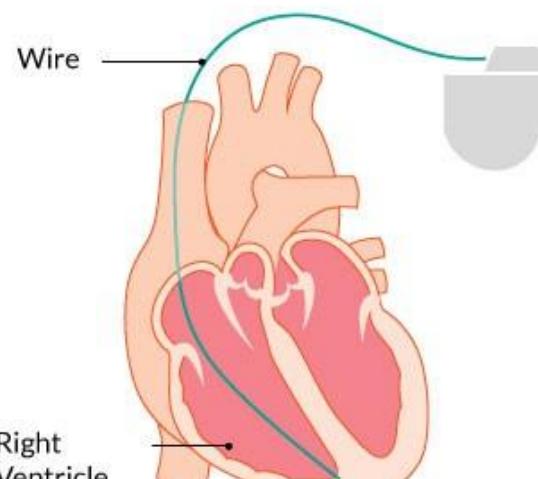
Soucieux de résERVER le meilleur accueil qui soit pour la formation des jeunes chercheurs, le Liryc accompagne alors ces derniers dans les inscriptions, les formations, les programmes internationaux, la médiation scientifique ou encore la professionnalisation.

3. Travaux effectués pendant le stage

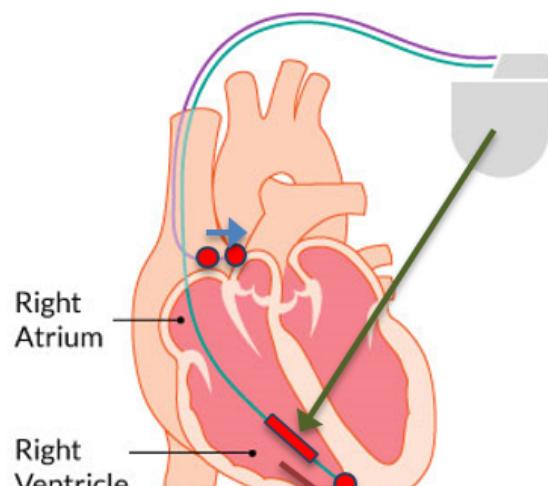
Comme décrit dans la partie précédente, il existe plusieurs pôles de recherche au sein de l'IHU Liryc, tous spécialisés dans des domaines différents de l'étude du cœur, de sa physiologie, de son fonctionnement et des pathologies du rythme cardiaque. Pour ma part, j'intègre l'équipe Traitement du signal, l'une des trois équipes du pôle Technologie pour la santé avec les équipes Imagerie et Science de données multimodales. Mon travail s'inscrit dans le cadre d'un projet de l'équipe Traitement du signal qui porte sur le traitement a posteriori d'électrogrammes provenant de dispositifs électroniques cardiaques implantés.

3.1 Contexte du stage et problématique abordée

En France, plus de 410,000 patients atteints d'arythmie cardiaque se sont vu implanter un appareil électronique cardiaque (CIED) de type défibrillateur pour surveiller leur rythme cardiaque de façon continue. Plus de 80,000 nouvelles implantations d'appareils sont prévues chaque année. Deux types de CIED sont utilisés pour surveiller en continu et à distance l'activité électrique du cœur.



Single Chamber ICD



Dual Chamber ICD

Figure 1 : Single Chamber ICD, appareil mesurant l'activité électrique du cœur au niveau du ventricule droit.

Figure 2 : Dual Chamber ICD, appareils mesurant l'activité électrique au niveau du ventricule droit et au niveau de l'oreillette/atrium droit.

Les mesures de l'activité électrique du cœur à la surface de celui-ci sont enregistrées : on obtient alors un électrogramme. À partir d'un Dual Chamber ICD, on obtient un électrogramme avec plusieurs signaux électriques comme montré ci-dessous :

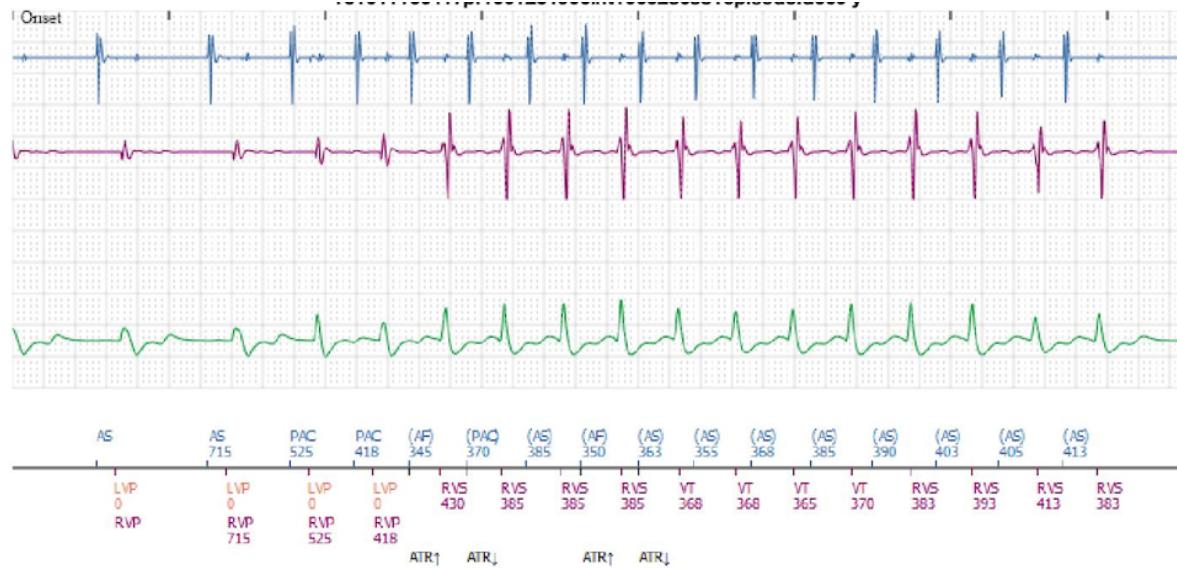


Figure 3 : Exemple d'électrogramme obtenue à partir d'un Dual Chamber ICD. Le tracé bleu correspond à l'activité électrique au niveau de l'oreillette droite (**voie auriculaire**) ; celui en mauve à l'activité électrique au niveau du ventricule droit (**voie ventriculaire**) ; le dernier tracé en vert correspond à la différence de potentiel électrique entre la sonde et le boîtier du dispositif, représentée par la flèche verte sur la figure 2.

Les CIEDs mesurent et enregistrent l'activité électrique du cœur en continu, traitent l'information de manière embarquée, délivrent une thérapie le cas échéant, et ils envoient les tracés pertinents sur des plateformes dédiées pour être traités par le service de suivi. Lorsqu'une anomalie est détectée, dénotée par les marqueurs en bas de la figure 3, le CIED la traite par une stimulation rapide dans un premier temps puis un choc électrique si nécessaire.

Une anomalie fréquente au niveau des oreillettes est la fibrillation auriculaire (FA), ce type d'anomalie ne doit pas être traité par le CIED. Il est donc important de la différencier des anomalies électriques ventriculaires. La FA est caractérisée par une activité électrique anarchique et rapide du muscle des oreillettes qui entraîne alors : [\[6\]](#)

- des contractions des oreillettes très rapides et saccadées (400 à 600 par minute), au point que cette partie du cœur semble immobile ;
- une stagnation du sang dans les oreillettes qui se contractent mal, en particulier dans l'oreillette gauche, pouvant donner lieu à la formation de caillots de sang. Ces caillots ou thrombus peuvent être propulsés dans une artère et entraîner un accident vasculaire cérébral ;
- une accélération de la contraction des ventricules, situés sous les oreillettes. Les ventricules se mettent aussi à battre vite et irrégulièrement au rythme de 150

battements par minute : on parle de tachyarythmie. Ils sont moins efficaces et le débit cardiaque baisse, pouvant être responsable d'une insuffisance cardiaque.

Un algorithme intégré dans l'appareil permet de mettre en évidence la fenêtre de temps où il pense avoir détecté une fibrillation auriculaire dans un électrogramme.

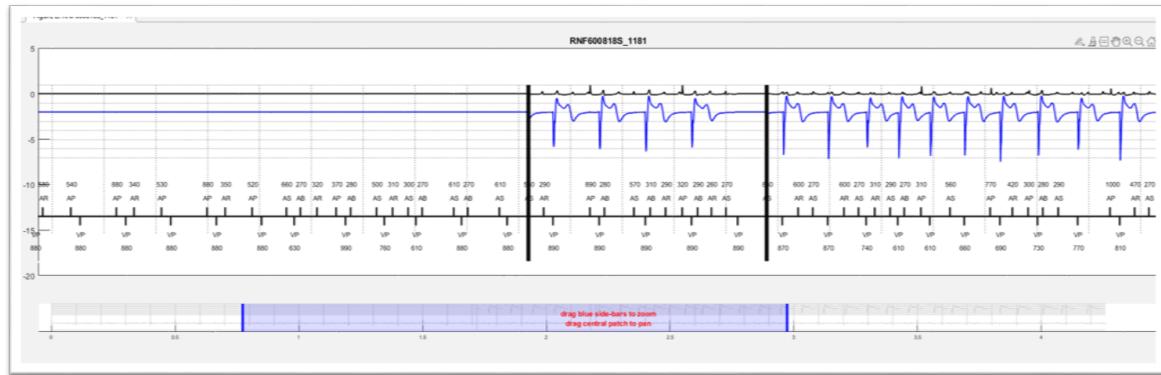


Figure 4 : Électrogramme de FA récupéré par le système de soin (ici CHU de Bordeaux/Liryc) après une détection par l'appareil. La fenêtre de temps où l'appareil a détecté une FA est représenté entre les deux barres noires verticales. Le tracé en noir représente la voie auriculaire et en bleu la voie ventriculaire.

La détection de FA (événement positif) soulève deux problématiques. La première est la détection de FA à tort (faux-positif). Dans ce cas, un rythme rapide est pensé conséquence de cette FA et n'est pas traité par le CIED, ce qui peut avoir de lourdes conséquences pour le patient. La seconde problématique est la non détection de FA (faux-négatif) lorsqu'il y en a effectivement une. Dans ce cas, une thérapie peut être délivrée à tort par le CIED.

Sur l'ensemble des événements remontés et étiquetés 'FA' par le CIED, il est particulièrement important de repérer un type de signal appelé bruit de sonde. Ce bruit peut être précurseur de la rupture de la sonde avec des conséquences graves sur la santé du patient.

Ainsi l'objectif de ce stage est de développer un algorithme, qui sera basé sur une stratégie de Machine Learning, capable de détecter automatiquement les FA sur l'EGM du CIED d'un patient. En particulier, cet algorithme devra être capable de discriminer le signal caractéristique d'une FA de celui d'un bruit pour permettre de ne présenter pour un diagnostic médical que les enregistrements 'suspects' pouvant correspondre à un bruit de sonde.

3.2 Données à disposition

Comme évoqué précédemment, les données des EGM sur les appareils des patients sont récupérées par le Liryc (et anonymisées). L'institut dispose d'une base de données de plus de 32,000 EGM pour lesquels un CIED a détecté une FA. On s'intéresse ici à la fenêtre de temps d'un EGM décrite dans la figure 4, celle-ci constitue un enregistrement dont on se servira. En limitant la base de données à un maximum de 34 enregistrements par patient, on obtient une base de données contenant plus de 10,000 enregistrements. Ces derniers ont été manuellement labellisés par trois médecins différents en deux classes :

- classe 1 ("AF") : l'enregistrement présente bien une FA ;
- classe 0 ("other") : l'enregistrement ne présente pas de FA du tout.

Deux médecins se sont chargés de déterminer si chaque enregistrement présente une FA. Sur les 10,174 enregistrements qu'ils ont eu en commun, 9,129 ont reçu le même label par ces deux médecins contrairement à 1,045 autres. Un troisième médecin a permis de trancher. Un consensus sur les labels a donc été trouvé sur 10,172 enregistrements, ceux-ci constituent la base de données qui va être utilisée. Cette base contient 8,057 enregistrements de classe 1 et 2,115 de classe 0, présentant donc un déséquilibre entre les classes d'environ 80/20. Ces enregistrements proviennent de 652 patients différents. Il y a un nombre d'enregistrements variant entre 1 et 34 selon les patients, dont plusieurs de chaque classe. On constate ainsi sur la figure 5 que 126 patients ont des enregistrements des deux classes. Leurs nombres varient énormément d'un patient à l'autre comme on peut le voir sur la figure 6. Cela représente en tout 2,718 enregistrements de la base de données.

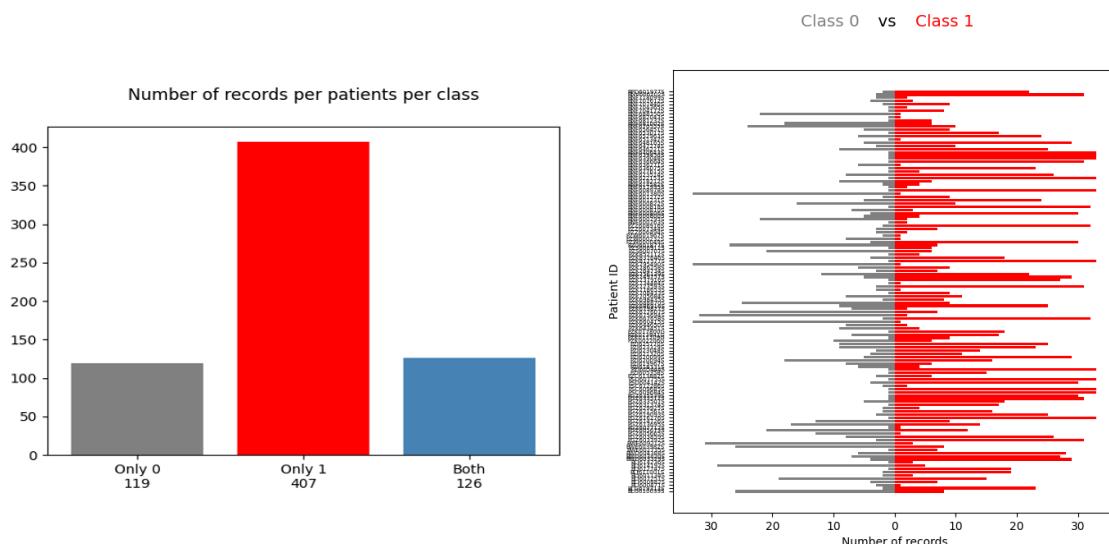


Figure 5 : De gauche à droite, nombre de patients n'ayant que des enregistrements de classe 0, que de classe 1 et ayant des enregistrements des deux classes respectivement.

Figure 6 : Nombre d'enregistrements de classe 0 et de classe 1 pour chaque patient ayant des enregistrements des deux classes.

Chaque enregistrement comporte les signaux des deux voies auriculaire et ventriculaire sur une certaine durée, exprimée en points d'échantillonnage à 128 Hz. Ceux-ci comportent le même nombre de points entre eux, espacés de 7.8078 millisecondes. Cependant, d'un enregistrement à l'autre, la longueur des enregistrements diffère.

Ainsi, un enregistrement comporte au minimum 513 points et 5,137 au maximum, ce qui représente une étalement des fenêtres de temps allant de 4 à 40 secondes.

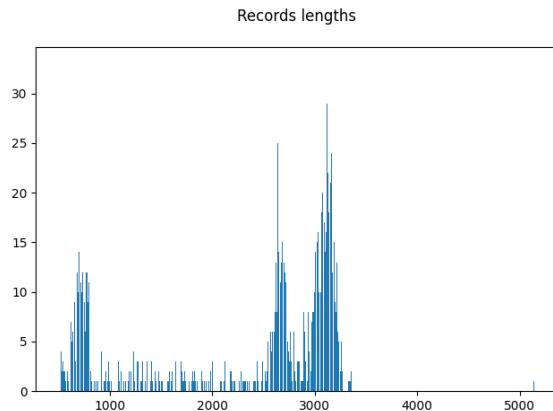


Figure 7 : Histogramme du nombre de points par enregistrement de la base de données

On va donc chercher à construire un algorithme qui puisse, à partir de cette base de données, déterminer la classe d'un nouvel enregistrement parmi les deux classes présentées précédemment. Pour cela, deux approches ont été réalisées : la première par l'utilisation d'un modèle de réseau de neurones convolutionnel pour une discrimination entre les 2 classes et la seconde par la modélisation par un réseau de la densité de probabilité de la classe 'FA' et les incertitudes de prédiction d'un enregistrement par rapport à cette distribution. L'intégralité de ces travaux a été effectué en Python, notamment avec la bibliothèque PyTorch pour concevoir et manipuler des réseaux de neurones.

3.3 Organisation de la base de données

Avec cette problématique et les données décrites dans la partie précédente, on va faire de la classification binaire donc de l'apprentissage supervisé. De plus, pour les deux approches, nous allons utiliser des réseaux de neurones qu'il faudra alors entraîner. On va alors devoir procéder en deux phases pour les deux approches : une phase d'entraînement avec un ensemble de données sur lesquelles l'algorithme va apprendre à discriminer les données d'une classe des données de l'autre classe et une phase de test pour évaluer ses performances sur de nouvelles données. Un découpage apprentissage/test des données est nécessaire pour cela. Puisqu'il y a 10,172 données/enregistrements, on procède à un découpage 80/20 afin d'avoir un maximum de données d'apprentissage, tout en gardant un nombre suffisant de données de test. L'algorithme apprendra donc sur environ 8,000 données et sera testé sur les 2,000 restantes.

Au vu des données, il va y avoir deux contraintes pour le découpage apprentissage/test. Premièrement, le déséquilibre 80/20 des classes : ~80% des données sont de classe 1 et ~20% de classe 0. Afin d'éviter à l'algorithme de n'apprendre que sur une classe de données, on fait en sorte de conserver le déséquilibre de classe dans les deux ensembles d'apprentissage et de test. Ensuite, comme montré dans la figure 5, certains patients possèdent plusieurs enregistrements, ces données sont donc liées entre elles. Pour pallier à

cela, on va faire en sorte que tous les enregistrements d'un patient se trouvent soit dans l'ensemble d'apprentissage, soit dans l'ensemble de test mais jamais dans les deux à la fois. Enfin se pose la question de la spécificité des patients. Il se peut qu'un ou plusieurs patient présentent des spécificités cardiaques. Celles-ci pourraient ne pas être reconnaissables par notre algorithme. La connaissance de ces spécificités et leurs origines sortent malheureusement du cadre posé lors de ce stage puisque nécessitant de plus amples informations sur les patients et leurs données médicales. On supposera donc pour la suite que l'on peut négliger ces hypothétiques spécificités. De plus, comme on ne peut savoir quels ensembles de données sont optimaux pour concevoir le meilleur modèle, on pourra procéder à plusieurs découpages apprentissage/test.

3.4 Pré-traitement des données

Une architecture de réseau de neurones prend en entrée des données sous un unique format. Ici, les données dont on dispose sont les enregistrements d' EGM correspondant à un ensemble de deux vecteurs de taille variable. Il faut donc uniformiser ces tailles.

Pour cela, on peut soit les réduire au minimum (513 points) ou à une taille inférieure, ce qui engendre une perte importante d'information, soit les augmenter. On privilégiera les méthodes qui allongent artificiellement la taille des enregistrements comme la duplication de signal ou le padding. La duplication de signal peut potentiellement poser problèmes au niveau des jonctions entre les duplicates de signaux. On utilisera donc le padding. Cette pratique consiste à allonger la taille d'un signal en ajoutant autant de valeurs que nécessaire (que des 0, des 1 ou des -42 par exemple). Pour nos enregistrements, on ajoutera des zéros puisque cette valeur correspond à l'absence de potentiel électrique au niveau du cœur, ce qui équivaut à une donnée manquante. Ce procédé s'appelle le zero-padding. Pour choisir la taille du padding, on peut prendre la taille maximum (5137) ou une puissance de deux proche de ce maximum (4096 ou 8192). Le choix des puissances de deux comme taille (pas uniquement de padding) est une pratique courante dans le milieu du Deep Learning pour des raisons computationnelles et de réduction de l'architecture couche par couche d'un facteur 2. Afin d'éviter toute perte d'information sur les signaux électriques, on prendra une taille de padding à 8192 : tous les enregistrements seront rallongés.

Une entrée sera donc au format $(2, 8192)$ (signal 1D avec 2 composantes) si l'on considère les deux voies auriculaire et ventriculaire et $(1, 8192)$ (signal 1D avec une composante) si l'on en considère que la voie auriculaire qui est la plus spécifique pour les tracés de FA.

3.5 Modèle de réseau de neurones

La première approche consiste en l'utilisation d'un modèle de réseau de neurones pour classifier les enregistrements parmi les deux classes décrites dans la section 3.2. Les enregistrements d'EGM étant considérés comme des signaux unidimensionnels (1D) avec au maximum deux composantes, on va utiliser une architecture type ConvNet. En particulier, on s'inspirera de l'architecture ResNet pour construire la nôtre.

3.5.1 Réseau neuronal résiduel

Une architecture de réseau de neurones type ConvNet se compose de plusieurs types de couches de neurones : des couches de convolutions, de Pooling, dense, d'activation, de BatchNormalization et de Dropout.

Une couche de convolution opère N convolutions entre le signal en entrée à M composantes et N noyaux (matrices de taille $k \times k$) générant un signal en sortie avec N composantes appelées filtres. Pour des signaux 1D, la couche de convolution utilise des noyaux de taille $(k \times 1)$ et se déroule comme indiqué sur la figure ci-dessous.

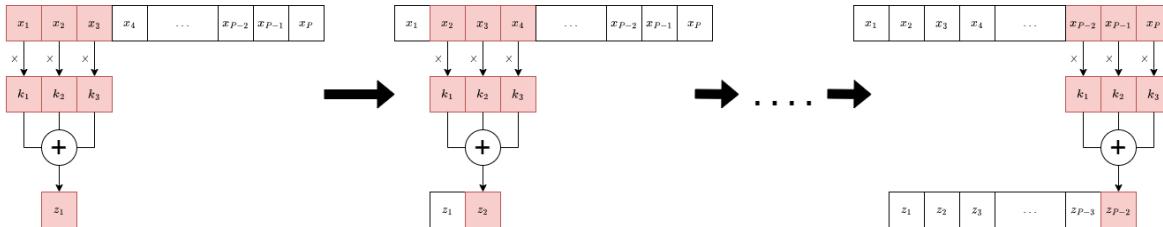


Figure 8 : Schéma de la progression des opérations effectuées lors d'une convolution entre un signal 1D et un noyau de taille (3×1) .

À noter que par défaut, toutes les composantes du signal en entrée de la couche de convolution sont utilisées pour générer tous les filtres en sortie. On peut faire en sorte que la première moitié des composantes du signal (resp. la seconde) en entrée soit utilisée pour générer la première moitié des filtres en sortie (resp. la seconde). Dans le cas des EGM, cela reviendrait à traiter les informations des deux voies auriculaire et ventriculaire indépendamment, avant de les rassembler à un certain point du réseau. Cela se fait grâce à un paramètre, appelé *groups*, valant par défaut 1 et s'illustre sur la figure 9



Figure 9 : Schéma d'une convolution entre un signal 1D à deux composantes et un noyau de taille $(3,1)$ pour plusieurs valeurs du paramètre $groups$. À gauche pour $groups=1$. À droite pour $groups=2$.

Hormis les couches de Pooling, de Dropout et d'activation, chaque couche du réseau possède des poids (appelés aussi paramètres) θ initialisés aléatoirement (par exemple selon une distribution uniforme). Ces poids déterminent les performances d'un modèle à

prédir une sortie qui sera ici de dimension 2 : $z_k = \begin{pmatrix} z_{k,0} \\ z_{k,1} \end{pmatrix}$ servant à calculer respectivement la probabilité d'appartenir à la classe 0 et la probabilité d'appartenir à la classe 1 via la fonction Softmax définie par $\begin{pmatrix} z_{k,0} \\ z_{k,1} \end{pmatrix} \mapsto \begin{pmatrix} \frac{\exp z_{k,0}}{\exp z_{k,0} + \exp z_{k,1}} \\ \frac{\exp z_{k,1}}{\exp z_{k,0} + \exp z_{k,1}} \end{pmatrix}$.

Pour entraîner notre modèle de réseau de neurones, on cherche donc à modifier ces poids. Pour cela, on lui passe en entrée une donnée x_k , il renverra en sortie pour chacune

d'elles un vecteur de nombre réel $z_k = \begin{pmatrix} z_{k,0} \\ z_{k,1} \end{pmatrix}$ grâce auquel on calcule la probabilité que x_k appartienne à la classe 0 et la probabilité que x_k appartienne à la classe 1. On prédit la classe d'un enregistrement selon la probabilité la plus élevée. Cette étape est la propagation en avant ou forward pass.

À noter que la sortie aurait pu n'être qu'un scalaire qui permet de calculer la probabilité d'appartenir à une des deux classes. En prenant le complémentaire de cette probabilité on aurait ainsi obtenu la probabilité d'appartenir à l'autre classe. On ne peut procéder de cette façon pour des raisons d'implémentation dûes à PyTorch en particulier pour l'implémentation de coefficient de rééquilibrage entre les classes.

Afin d'améliorer la capacité du modèle à prédire la classe des données, on mesure l'écart entre les prédictions et la vérité terrain (les vraies classes des données) grâce à une fonction objectif (**Loss**), qu'on cherche à minimiser. Ici, pour de la classification binaire, on prendra l'entropie croisée, dont la formule de calcul (sur PyTorch) s'écrit

$$L_\theta(z, y) = \frac{-1}{\sum_{c=0}^{K-1} w_c} \sum_{i=1}^n \left[\sum_{c=0}^{K-1} w_c \log \left(\frac{\exp(z_{i,c})}{\sum_{k=1}^K \exp(z_{i,k})} \right) 1_{y_i=c} \right]$$

où $y = (y_1, y_2, \dots, y_n)$ sont les classes, parmi les $K = 2$ classes, des n entrées du réseau $x = (x_1, x_2, \dots, x_n)$ dont les sorties du réseau sont notées $z = (z_1, z_2, \dots, z_n)$, et w_0 et w_1 sont des poids associés à chaque classe.

Comme cette fonction objectif est paramétrée par les poids du modèle, on va donc la minimiser par rapport θ . Pour cela, on effectue une descente de gradient pour trouver un nouvel ensemble de poids pour le modèle qui minimise L_θ : c'est la rétropropagation du gradient ou backward pass. Elle se fait grâce à un algorithme d'optimisation (SGD, Adam, etc.).

Le modèle va effectuer alternativement forward pass et backward pass pour toutes les données passées au modèle par paquets (**batchs**). Lorsque tous les batchs ont été vus par le modèle, alors il vient de terminer une itération de l'entraînement du modèle (**epoch**).

François Chollet, dans son livre **Deep Learning with Python**^[2], décrit plus précisément le fonctionnement d'un réseau de neurones et de ces différentes couches, son entraînement ainsi que les différents types de réseau couramment utilisés.

Lors de la rétropropagation, le gradient au niveau de chaque couche est calculé selon le théorème de dérivation des fonctions composées. Pour un réseau à C couches, le gradient à la i^{me} couche est le produit des gradient calculés aux couches i à C . Il peut arriver au cours de la rétropropagation du gradient qu'un ou plusieurs facteurs de ce produit prennent une valeur nulle ou quasi nulle. Le gradient devient donc (quasi) nul ce qui empêche la descente de gradient de converger vers un minimum local ou global. Le gradient calculé étant très faible, les poids du modèle ne seront quasiment pas modifiés. Ce phénomène connu est la disparition du gradient (**vanishing gradient**). Une façon de pallier à cet éventuel problème est d'utiliser une architecture ConvNet spécifique : un réseau neuronal résiduel (ResNet). On désigne par bloc de convolution un ensemble de plusieurs couches de convolution alternées avec des couches de BatchNormalization et d'activation. L'architecture ResNet utilise des blocs de convolution particuliers, schématisés dans la figure ci-dessous.

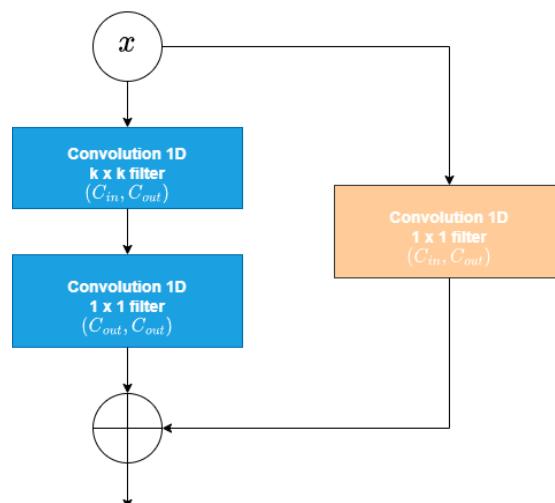


Figure 10 : Schéma d'un bloc ResNet. La sortie du bloc de convolution est sommée avec son entrée avant d'être passée à la prochaine couche du réseau.

Si pour l'entrée x d'un bloc de convolution usuel, l'ensemble de ses opérations est décrit par la fonction $x \mapsto F(x)$, celle d'un bloc ResNet est alors décrite par $x \mapsto F(x) + x$. Le gradient calculé à ce bloc ResNet vaut alors $\nabla F(x) + 1$. De cette façon, même si $\nabla F(x)$ est nul, le gradient calculé vaudra au minimum 1, assurant l'actualisation des poids du modèle dans les couches en amont.

3.5.2 Hyperparamètres du modèle

Les paramètres d'un modèle de réseau de neurones sont ses poids associés à chacune de ses couches. Ces derniers seront modifiés et optimisés pendant l'entraînement du modèle. Cependant, il existe des hyperparamètres liés aux modèles de réseaux de neurones, à optimiser à la main.

Tout d'abord, il y a le nombre d'epochs, c'est-à-dire le nombre d'itération pendant l'entraînement. S'il est trop faible, le modèle n'apprendra pas suffisamment. Au contraire, il provoque le phénomène de surapprentissage résultant en une mauvaise généralisation du modèle. De plus, à cause de l'aléa dans l'initialisation des poids de chaque couche et par conséquent dans l'entraînement, ce nombre peut varier, rendant la recherche d'un nombre optimal laborieux. Il existe néanmoins une méthode, que l'on détaillera plus tard, permettant d'optimiser automatiquement cet hyperparamètre : l'**Early Stopping** (ou arrêt précoce/prématuré).

Ensuite, l'algorithme d'optimisation impliqué dans la descente de gradient possède, comme dit dans le début de cette partie 3.5.1, un taux d'apprentissage γ . Ce taux détermine la largeur du pas lors de la descente de gradient. Il influence la convergence de la descente et donc la qualité de l'entraînement. Selon l'algorithme d'optimisation utilisé, il peut être couplé à d'autres hyperparamètres. Pour l'algorithme de descente de gradient stochastique avec momentum (SDGm), c'est avec un paramètre de momentum μ , pour l'algorithme Adam, c'est avec trois paramètres (β_1, β_2) et ϵ . Alors qu'on cherchera des paramètres optimaux pour le momentum, on ne le fera pas pour les hyperparamètres d'Adam. D'après l'article présentant cet algorithme^[8], les valeurs par défaut de $\beta_1 = 0.9$, $\beta_2 = 0.999$ et $\epsilon = 10^{-8}$ sont convenables pour les problèmes de Machine Learning que les auteurs ont testés. Quatre algorithmes seront évalués pour notre problème, Adam, SGD (Stochastic Gradient Descent), SGD avec momentum et SGD avec gradient accéléré de Nesterov.

Enfin, la fonction objectif utilisée pour tous mes modèles est l'entropie croisée possède un couple d'hyperparamètres w_0 et w_1 qui sont des poids associés aux classes 0 et 1 respectivement. Ils permettent, lors du calcul de la fonction objectif, de donner une importance plus ou moins grande à une des deux classes par rapport à l'autre. Ces poids viennent contrebalancer le déséquilibre des classes. En effet, pendant l'entraînement, on peut donner plus d'importance au fait de mal prédire la classe minoritaire. Au vu du déséquilibre 80/20 de nos données, il est essentiel de trouver un ensemble de poids optimal pour entraîner un modèle.

Par la suite, on va voir que la taille des batchs (paquets de données passés au modèle pendant l'entraînement) va influencer les performances d'un modèle. On va donc chercher à trouver la taille de batch qui permet d'obtenir les meilleures performances pour un modèle.

3.5.3 Critère d'évaluation d'un modèle

Pour sélectionner les meilleurs hyperparamètres, et plus généralement pour sélectionner l'architecture de notre modèle de réseau de neurones, plusieurs fonctions de scores sont à disposition. Comme on est dans un cadre de classification binaire, ces scores seront des fonctions des vrais positifs (TP), faux négatifs (FN), faux positifs (FP) et vrais négatifs (TN). Parmi les fonctions de scores utilisées pour apprécier la qualité d'un modèle, il y a :

- l'accuracy :
$$Acc = \frac{TP + TN}{TP + FN + FP + TN}$$
- le taux de vrais positifs (TPR), aussi appelé sensibilité :
$$TPR = \frac{TP}{TP + FN}$$
- le taux de vrais négatifs (TNR), aussi appelé spécificité :
$$TNR = \frac{TN}{FP + TN}$$
- la valeur prédictive positive (VPP), aussi appelé précision :
$$PPV = \frac{TP}{TP + FP}$$
- la valeur prédictive négative (NPV) :
$$\frac{TN}{FN + TN}$$
- la balanced accuracy, qui est la moyenne arithmétique du TPR et du TNR

La fonction de score qui sera principalement utilisée est le F score, ici pour la classe 0. En effet, même si on souhaite bien prédire toutes les classes, on va plus chercher à discriminer les enregistrements de classe 0 des enregistrements de classe 1. Ce dernier est défini comme la moyenne harmonique entre la sensibilité et la précision. Plus généralement, le F_β score est une moyenne harmonique pondérée par un coefficient $\beta > 0$. En terme de TP, FN, FP et TN, il s'exprime ainsi :

$$F_\beta = \frac{(1 + \beta^2).TP}{(1 + \beta^2).TP + \beta^2.FN + FP}$$

Pour $\beta = 1$, on définit le F_1 score pour la classe 0 par la moyenne harmonique entre NPV et TNR, i.e. :

$$F_1(0) = \frac{2.TN}{2.TN + FN + FP}$$

Ce score a l'avantage de se focaliser sur la bonne prédiction des enregistrements en classe 0 : il varie beaucoup plus selon si un modèle prédit bien ou non un enregistrement en classe 0, ce qui est utile vu le déséquilibre des classes. De plus, il prend en compte la bonne prédiction de la classe 1 par le terme de FN au dénominateur. Supposons qu'un modèle fait autant de bonne prédiction en classe 0 que d'erreurs de prédiction ($TN = FP + FN$), son F_1 score sera alors de 0.66, valeur qu'on peut prendre comme seuil pour la suite.

Sachant qu'au plus, il y a 8057 enregistrements de classe 1 et 2115 de classe 0, un modèle naïf qui les prédit tous en classe 1 aura :

- une accuracy et une PPV à ~0.792
- une balanced accuracy à 0.5
- une TPR à 1
- un F_1 score (pour la classe 0), un TNR et une NPV à 0

À l'exception de la valeur du TPR, ces valeurs de scores serviront de seuil pour considérer un modèle comme bon.

3.5.4 Méthode de sélection d'une architecture et des hyperparamètres

Comme indiqué dans la partie 3.5.1, les couches d'un réseau de neurones possèdent des poids initialisés aléatoirement. Ainsi, plusieurs modèles ayant la même architecture et les mêmes hyperparamètres vont avoir des performances différentes suite à un entraînement sur les mêmes données. Cet aléa va nous contraindre sur la sélection d'une architecture et des hyperparamètres. Pour comparer les architectures et leurs hyperparamètres entre eux, il faudra utiliser les résultats de plusieurs modèles dont les poids sont initialisés aléatoirement. On fera de la validation croisée 5-folds pour sélectionner une architecture et un ensemble d'hyperparamètres. Les données d'apprentissage devront donc être découpées en 5 sous-ensembles (5 folds) avec les mêmes contraintes que lors du découpage apprentissage/test des données. Ce sera l'unique découpage en folds que l'on fera, pour pouvoir comparer sur les mêmes ensembles de données.

On prend une architecture de réseau de neurones et on se fixe un ensemble d'hyperparamètres (poids des classes, taux d'apprentissage et taille de batchs). Pour un fold k parmi les 5, on initialise 5 modèles différents (5 "initialisations") que l'on va entraîner sur les autres folds que le fold k pendant 1000 epochs (les folds d'apprentissage). On mesure à chaque epoch leurs performances sur le fold k (le fold de validation), c'est-à-dire qu'on leur fait prédire les classes des enregistrements présents dans le fold k puis on enregistre leur score, définis en 3.3.5. À l'issue de cela, on a 5 modèles par fold, donc 25 modèles au total. On prend, pour chaque fold, le modèle ayant eu le meilleur F_1 score sur son fold de validation : les 5 meilleurs modèles sont retenus. On va faire la moyenne des F_1 scores de ces 5 meilleurs modèles par epoch. Elle servira à comparer les différentes architectures entre elles et/ou les différents choix d'hyperparamètres et ainsi à sélectionner la meilleure architecture et les meilleurs hyperparamètres par rapport à ce F_1 score moyen sur la classe 0. Le processus de sélection par validation croisée 5-folds est illustré dans la figure 11.

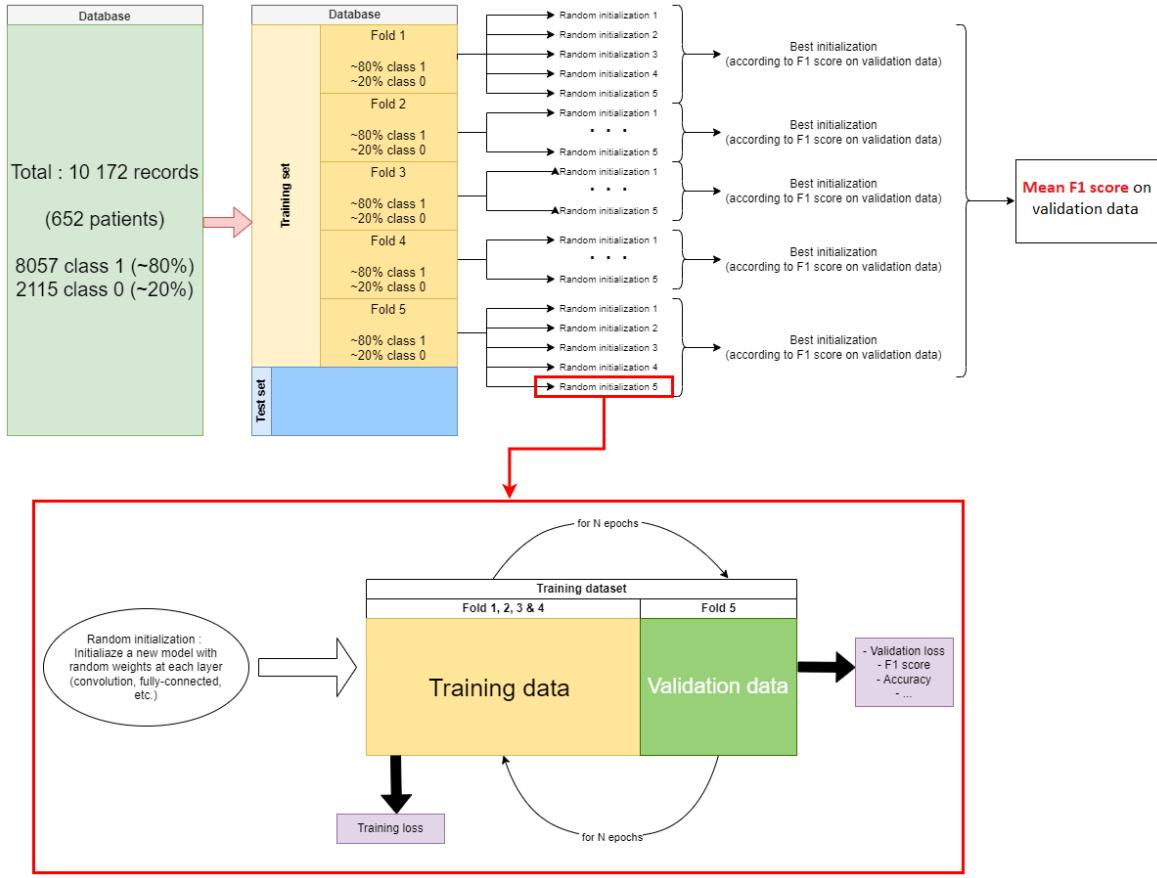


Figure 11 : Schéma d'obtention du F_1 score moyen (pour la classe 0) d'une architecture donnée avec un ensemble d'hyperparamètres fixés. Pendant le procédé de validation croisée, on compare les architectures et/ou les hyperparamètres par rapport à ce F_1 score moyen.

Le choix de prendre un nombre d'epochs à 1000 est purement arbitraire. Il repose sur l'a priori que ce nombre est suffisamment grand pour que n'importe quel modèle, pour ce problème de classification et ces données, voit ses performances stagner après autant d'epoch.

3.5.5 Premières architectures

Pour commencer à manipuler des réseaux de neurones sur PyTorch, j'ai construit des architectures de réseaux convolutionnels simplistes. Les entrées de ces modèles n'étaient alors constituées que de la voie auriculaire des enregistrements d'EGM. La première architecture, meilleure qu'un modèle naïf se présente ainsi :

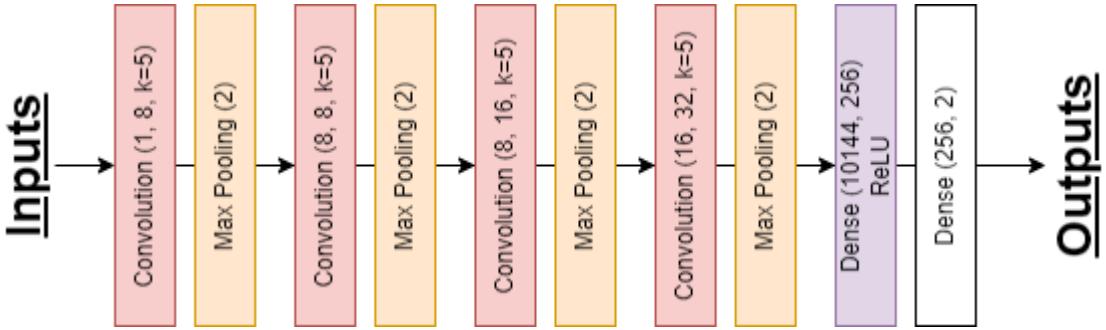


Figure 12 : Schéma de la première architecture ConvNet meilleure qu'un modèle naïf. Les entrées sont les enregistrements avec les deux voies et les sorties sont celles décrites dans la partie 3.5.1.

On teste l'influence des tailles de batchs en entraînant 5 modèles par fold sur les 5 folds, pendant 50 epochs, avec des poids associés à chaque classe à $w_0 = 0.8$ et $w_1 = 0.2$ et avec l'algorithme Adam avec un taux d'apprentissage à 10^{-3} . On affiche dans la figure 13 l'accuracy moyenne de ces 25 modèles avec leur écart-type. La droite horizontale rouge représente le seuil d'accuracy (accuracy d'un modèle naïf). Les résultats sont hétérogènes par rapport à la taille des batchs, il y a donc bien une influence notable de la taille des batchs sur les performances d'un modèle. De plus, on peut noter que plus les batchs sont petits, plus le temps d'entraînement est long. La figure 14 indique le temps moyen en seconde que prend l'entraînement d'un seul modèle selon la taille de batchs. Cela est dû au fait que plus la taille des batchs est petite, plus il y aura de paquet de données à traiter l'un après l'autre et donc plus d'opérations à effectuer.

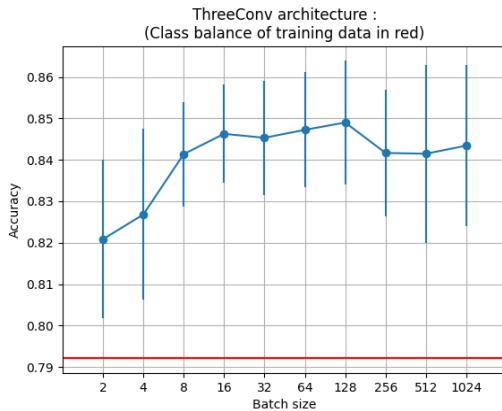


Figure 13 : Accuracy moyenne pour 25 initialisations aléatoires de l'architecture en figure 12 par taille de batch.

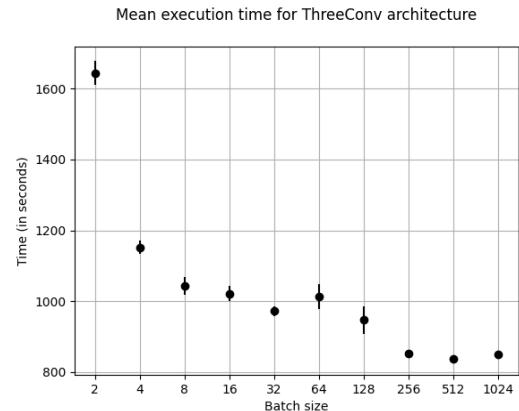


Figure 14 : Temps d'entraînement moyen (en secondes) pour 25 initialisations aléatoires de l'architecture en figure 12 par taille de batch.

On réitère le même genre d'entraînement sur 50 epochs avec le même algorithme d'optimisation pour comparer les poids associés à chaque classe. En supposant que les

résultats selon la taille des batchs suivent les mêmes tendances, on prendra une taille de batchs à 1024 pour minimiser le temps d'entraînement.

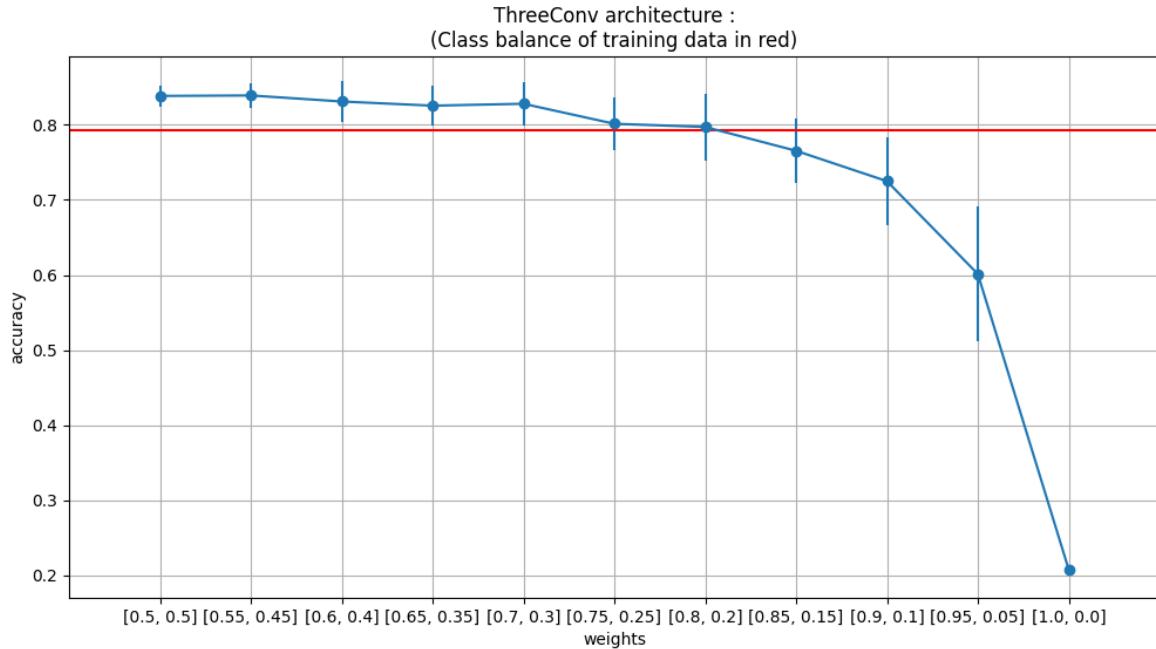


Figure 15 : Accuracy moyenne pour 25 initialisations aléatoires de l'architecture en figure 12 par couple de poids associés à chaque classe. Le premier poids est associé à la classe 0 tandis que le deuxième est associé à la classe 1.

On remarque qu'à mesure qu'on donne un poids important à la classe 0, l'accuracy moyenne, sur 25 modèles, décroît considérablement jusqu'à atteindre environ 20%. Ceci correspond à la proportion de classe 0 dans les données d'apprentissage. En regardant les performances de ces modèles, on s'aperçoit que plus on donne de l'importance au fait de bien prédire la classe 0, plus ils prédisent les enregistrements en classe 0. Un poids associé à la classe 0 adéquat semble être entre 0.5 et 0.8 (donc un poids associé à la classe 1 semble être entre 0.2 et 0.5). Prendre un poids pour la classe 0 inférieur à 0.5 (ou un poids pour la classe 1 supérieur à 0.5) va logiquement forcer un modèle à plus prédire un enregistrement en classe 1.

Comme on a une accuracy à peine supérieure à 80%, non loin de la valeur du seuil, on va maintenant travailler sur des architectures type ResNet.

3.5.6 Architecture ResNet et résultats

Les architectures ResNet possèdent un grand nombre de couches de neurones, donc de paramètres. Comme un trop grand nombre de paramètres peut amener à du surapprentissage (et un long temps d'exécution), on s'inspirera des blocs de ce type

d'architecture. On construit alors une première architecture avec 3 blocs ResNet (en bleu et rouge dans la figure ci-dessous) qui prend en entrée les deux voies auriculaire et ventriculaire d'un enregistrement..

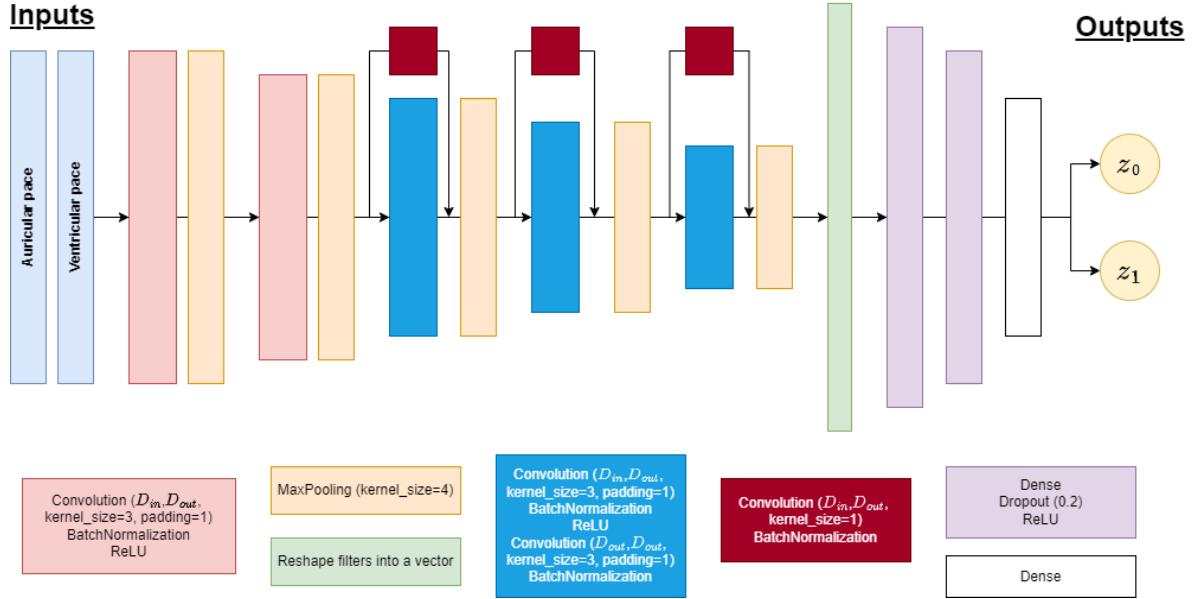


Figure 16 : Schéma d'une architecture inspirée de l'architecture ResNet que l'on a testé.

Comme pour les premières architectures jouets, on regarde les performances moyennes de modèles selon la taille des batchs, en les entraînant sur 50 epochs avec des poids associés à chaque classe de 0.7 et 0.3 pour les classes 0 et 1 respectivement, avec l'algorithme Adam (taux d'apprentissage à 10^{-3}). On obtient les graphiques suivants pour l'accuracy et le F1 score(pour la classe 0) :

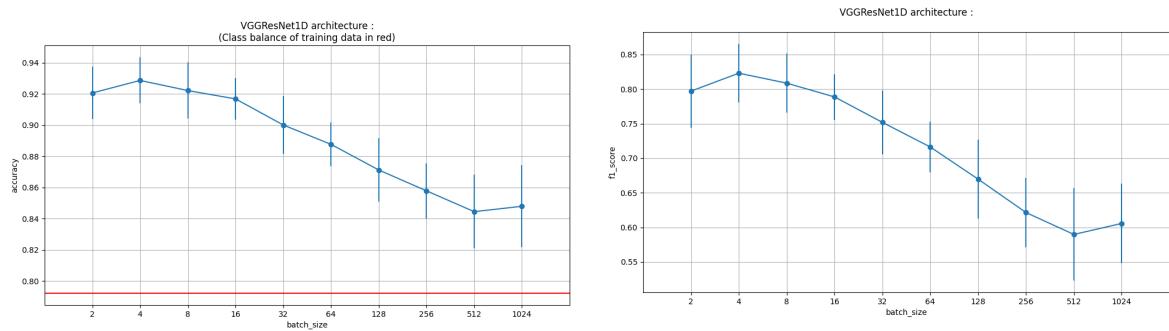


Figure 17 : Accuracy moyenne (à gauche) et F_1 score (pour la classe 0) moyen (à droite) pour 25 initialisations aléatoires de l'architecture en figure 16 par taille de batch.

On constate déjà qu'en moyenne, l'accuracy est plus élevée avec cette architecture qu'avec un simple ConvNet. De plus, le F_1 score est relativement bon, oscillant autour de la valeur seuil de 0,66 selon la taille des batchs.

En adaptant cette architecture à une entrée contenant qu'une seule des deux voies, on entraîne des modèles dans les mêmes conditions selon la taille des batchs.

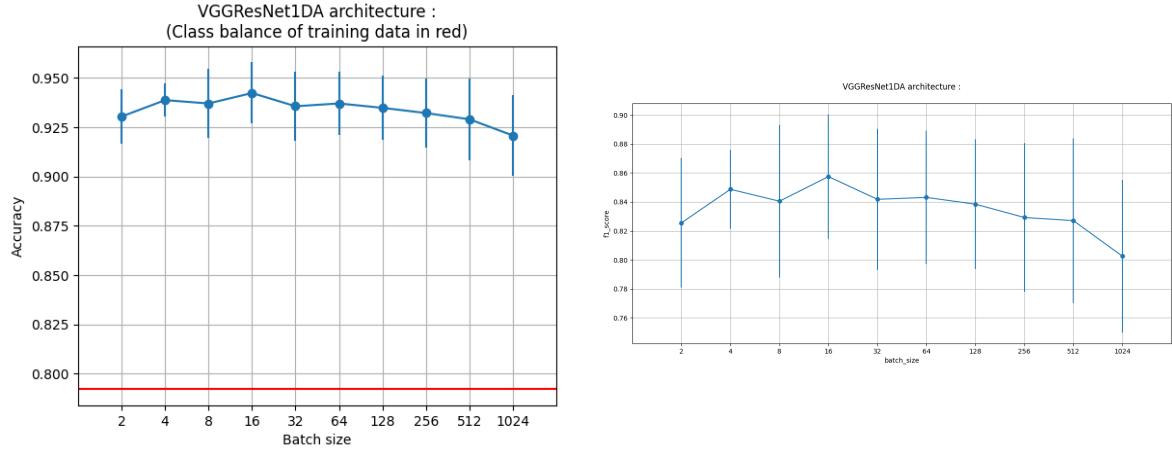


Figure 18 : Accuracy moyenne (à gauche) et F_1 score (pour la classe 0) moyen (à droite) pour 25 initialisations aléatoires par taille de batch. L'architecture testée ici ne prend que la voie auriculaire d'un enregistrement en entrée.

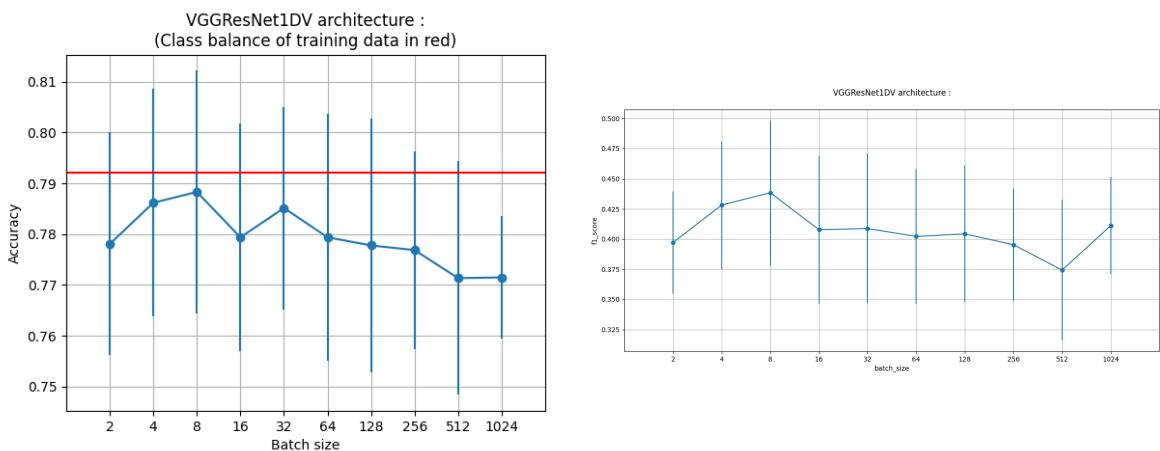


Figure 19 : Accuracy moyenne (à gauche) et F_1 score (pour la classe 0) moyen (à droite) pour 25 initialisations aléatoires par taille de batch. L'architecture testée ici ne prend que la voie ventriculaire d'un enregistrement en entrée.

Sur la figure 18, seule la voie auriculaire a été prise en compte dans l'entrée. L'accuracy et le F_1 score sont en moyenne bien supérieurs à ce qui a été obtenu jusqu'ici. De plus, les résultats sont homogènes par rapport à la taille des batchs. Au contraire, en prenant en

compte seulement la voie ventriculaire (figure 19), on a une accuracy et un F_1 score qui sont moindres et inférieurs aux seuils à ~0.8 et 0.66 respectivement. Il est finalement assez logique d'obtenir ces résultats. À elles seules, les informations obtenues au niveau du ventricule ne suffisent pas à détecter la présence d'une pathologie de l'oreillette. Il est toutefois utile de traiter les deux voies dans la mesure où il est probable que le bruit détecté par le dispositif cardiaque sur quelques enregistrements soit dû à un bruit lointain provenant du ventricule. Par la suite, on peut travailler sur ces deux types d'architectures, prenant en compte la seule voie auriculaire ou les deux.

En regardant la courbe de validation (valeurs de la fonction objective calculées sur l'ensemble de données de validation) des deux architectures, le phénomène de surapprentissage se produit très tôt dans l'entraînement des modèles. Il est alors bon de se questionner sur l'algorithme d'optimisation qui intervient pendant la phase d'apprentissage, en plus des poids associés à chaque classe.

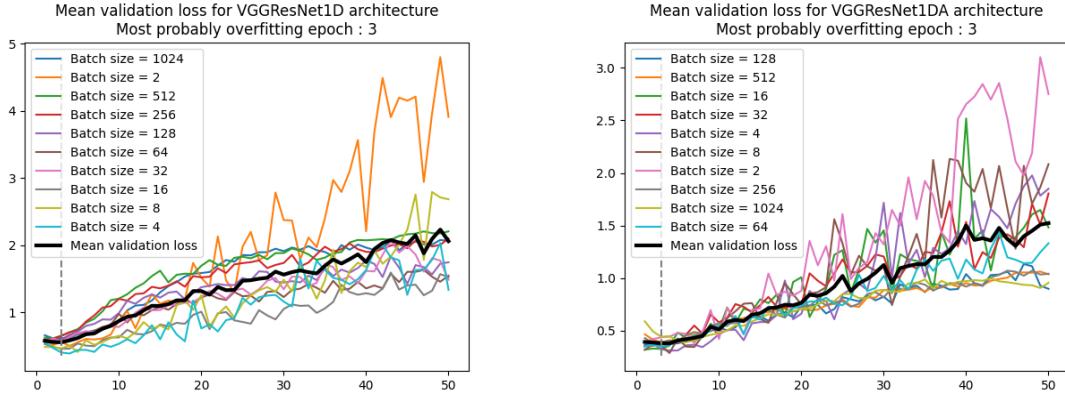


Figure 20 : Courbes moyennes de validation pour 25 initialisations aléatoires de l'architecture à deux voies (à gauche) et de l'architecture à une voie (à droite) par taille de batch. La moyenne de ces courbes est tracée en noir.

Parmi tous les algorithmes d'optimisation, on va regarder les algorithmes Adam et SGD. L'algorithme SGD applique la descente de gradient usuelle $\theta_{t+1} \leftarrow \theta_t - \gamma \nabla f_{\theta_t}(X_t)$ avec un paramètre de momentum (nul par défaut). Dans son implémentation, on peut aussi utiliser une version de l'algorithme avec le gradient accéléré de Nesterov. Les implémentations des trois déclinaisons de l'algorithme SGD sont décrites dans la documentation de PyTorch^[9].

Pour comparer les algorithmes entre eux, on va entraîner des modèles sur 1000 epochs avec différents poids associés à chaque classe et différents taux d'apprentissage. On utilisera $\mu = 0.99$ comme valeur de momentum d'après l'article de Ilya Sutskever et al.^[10]. La figure 37 en annexe A montre les valeurs du F_1 score de chaque modèle sur les données de validation au fil des epochs, par taux d'apprentissage et tout poids de classe confondus. Pour chaque algorithme, un taux d'apprentissage à 0.001 est optimal. C'est avec l'algorithme Adam qu'on obtient globalement les meilleures valeurs de F_1 score. Si on

$$\left[\frac{2}{3}, \frac{1}{3} \right]$$

regarde dans le détail des poids associés à chaque classe, c'est avec les poids $\left[\frac{2}{3}, \frac{1}{3} \right]$ que les valeurs de F_1 scores sont les plus élevés, suivis de près par les poids $[0.6, 0.4]$.

Au vu de ces résultats, Adam est donc l'algorithme d'optimisation le plus efficace pour entraîner nos modèles. Pour ces architectures, un taux d'apprentissage à 0.001 et des poids

$$\left[\frac{2}{3}, \frac{1}{3} \right]$$

de $\left[\frac{2}{3}, \frac{1}{3} \right]$ associés aux classes 0 et 1 respectivement constituent les hyperparamètres optimaux.

Dans la partie 3.5.1, nous avons décrit le paramètre groups d'une couche de convolution qui permet de traiter indépendamment les moitiés des composantes du signal. Afin d'améliorer notre architecture à deux voies, nous allons tester des variantes de cette architecture avec des couches de convolutions à groups=2. L'idée derrière étant de traiter indépendamment les deux voies jusqu'à un certain point du réseau. Ce procédé devrait permettre d'extraire un maximum d'informations indépendamment sur chaque voie avant de les combiner. Notre architecture comporte 11 couches de convolution, nous allons initialiser le paramètres groups à 2 d'abord sur la première couche seulement, puis les deux premières et ainsi de suite. Les convolutions au niveau des connexions des blocs ResNet resteront inchangées étant donné que leur seul objectif est de redimensionner le signal en entrée avant de le sommer à la sortie des multiples convolutions. Également, on va chercher un ensemble de couches denses et de neurones par couches ainsi qu'une probabilité p pour les couches de Dropout qui donneraient de meilleur résultats que cette architecture VGG-ResNet.

À force d'essais, la meilleure architecture prenant en compte les voies auriculaire et ventriculaire trouvée est la suivante :

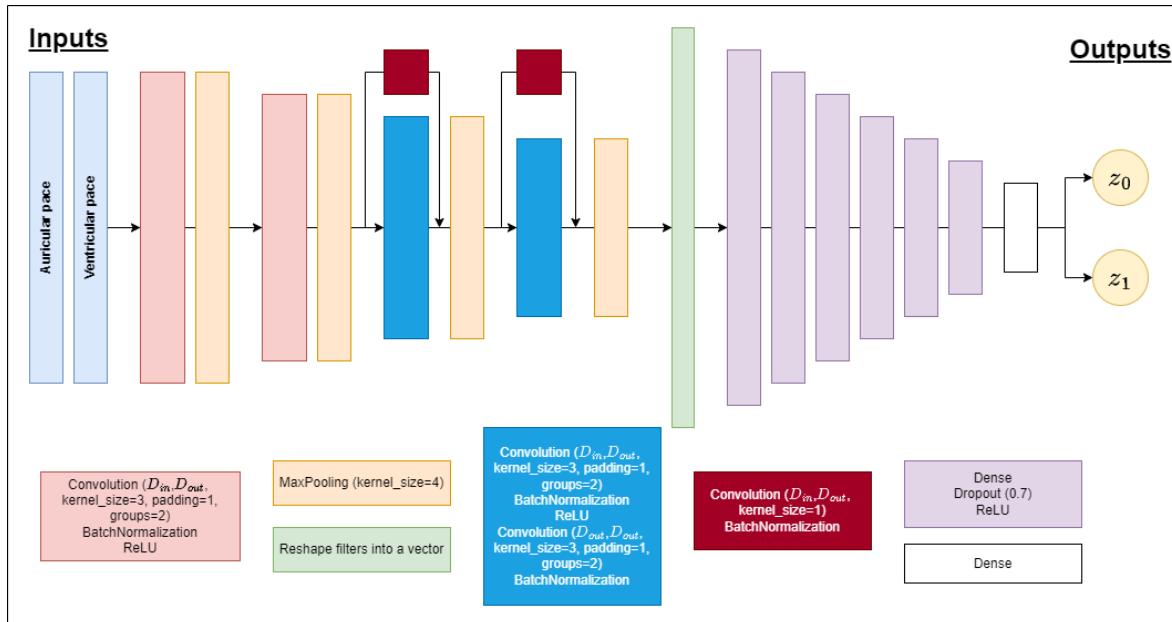


Figure 21 : Schéma de la meilleure architecture à deux voies trouvée.

De la même façon, on a expérimenté pour trouver la meilleure architecture ne prenant en compte que la seule voie auriculaire. Elle se présente ainsi :

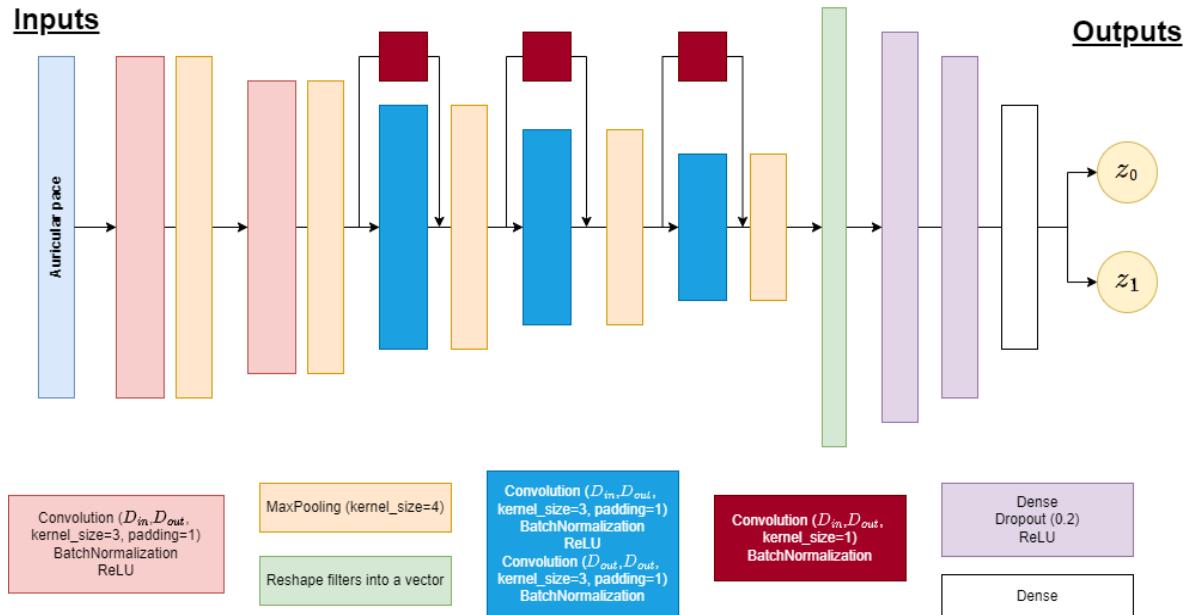


Figure 22 : Schéma de la meilleure architecture à une voie (auriculaire) trouvée.

Pour les deux architectures, on a ensuite cherché les meilleurs taux d'apprentissage et les poids associés à chaque classe avec la validation croisée 5-folds. Pour l'architecture prenant en compte les deux voies, on voit sur la figure 39 en annexe C que le meilleur taux

d'apprentissage est 0.001 et les meilleurs poids sont $\left[\frac{2}{3}, \frac{1}{3} \right]$.

On retrouve les mêmes hyperparamètres optimaux avec l'architecture qui ne prend en compte que la voie auriculaire, comme on peut le voir sur la figure 40 en annexe D.

Avec cet ensemble d'hyperparamètres, on va désormais pouvoir procéder à l'entraînement de modèles avec ces architectures sur toutes les données d'apprentissage.

Afin d'entraîner nos modèles sur un nombre suffisant d'epochs tout en évitant de faire du surapprentissage, on va utiliser le procédé d'Early Stopping. Celui-ci consiste à arrêter l'entraînement selon un critère. Généralement, on utilise un ensemble de données de validation pendant l'entraînement et on l'arrête si la courbe de validation ne décroît pas au bout de E epochs, où E est un paramètre de patience. Dans notre cas, on va arrêter l'entraînement d'un modèle lorsque la courbe d'apprentissage décroît ou stagne.

Les paramètres de l'entraînement pour les deux architectures sont détaillés dans le tableau suivant :

	Arch. à deux voies	Arch. à une voie
Nombre d'epochs max.	1000	1000
Patience	10	10
Poids associés aux classes 0 et 1	$\left[\frac{2}{3}, \frac{1}{3} \right]$	$\left[\frac{2}{3}, \frac{1}{3} \right]$
Algorithme d'optimisation	Adam	Adam
Taux d'apprentissage	0.001	0.001
Taille des batchs	1024 puis 64	1024 puis 16

Table 1 : Hyperparamètres choisis pour entraîner les architectures.

Après avoir entraîné et testé plusieurs dizaines de modèles par architecture, on les teste sur l'ensemble de données test puis on sélectionne le meilleur modèle par architecture selon leurs performances. Il en sort que les meilleurs modèles obtiennent les résultats suivants (tronqué à la quatrième décimale) :

	Arch. à deux voies	Arch. à une voie
F_1 Score (pour la classe 0)	0.9290	0.9594
Accuracy	0.9683	0.9817
TPR	0.9769	0.9892
TNR	0.9379	0.9558
PPV	0.9822	0.9871
NPV	0.9203	0.9629
Balanced Accuracy	0.9574	0.9725

Table 2 : Scores des meilleurs modèles obtenus sur l'ensemble de données test. Les résultats ont été tronqués à la quatrième décimale.

Ces modèles atteignent des performances excellentes et dépassent largement les valeurs seuils pour chaque score sur les données test.

De plus, en entraînant plusieurs modèles avec les mêmes architectures sur d'autres ensembles d'apprentissage/test, on constate une certaine stabilité dans les performances des deux architectures. Les graphiques en annexe B indiquent que les performances de nos architectures ne sont pas seulement dues aux ensembles d'apprentissage et de test.

Pour finaliser l'automatisation de la détection de FA sur l'EGM d'un patient, on va intégrer ces modèles à une application. Pour ce faire, on va utiliser la librairie Flask pour créer l'interface (web) de l'application.

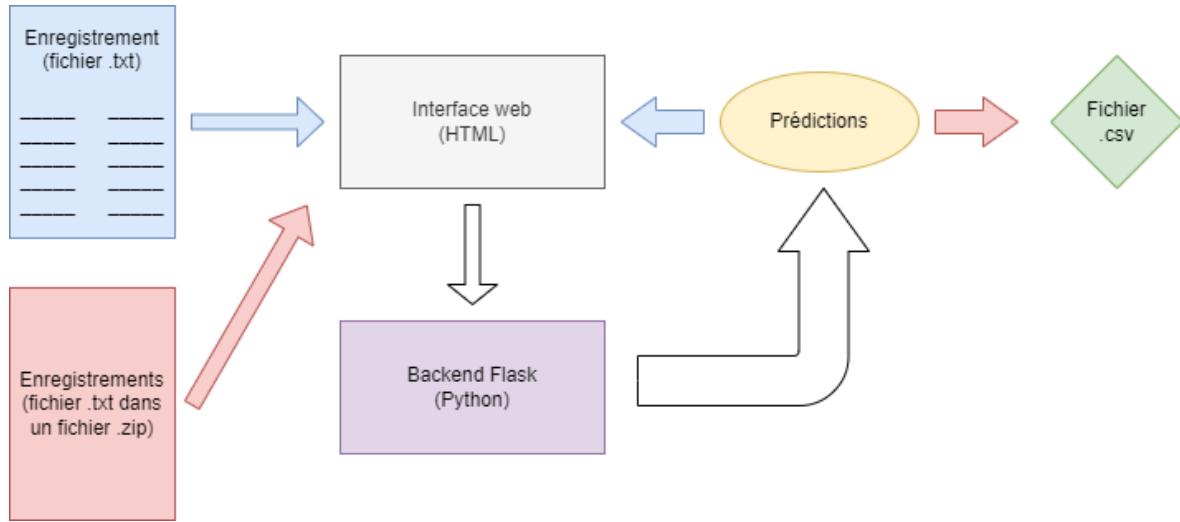


Figure 23 : Diagramme de flux de l'application flask.

Sur l'interface de l'application, on peut passer deux types de fichiers :

- un fichier texte (un seul enregistrement) : l'interface web affichera alors la classe prédite par les deux modèles de réseau de neurones ainsi que la probabilité d'appartenir à cette classe ;
- un fichier zip (plusieurs enregistrements) : l'interface web renverra un fichier csv contenant les classes des enregistrements prédites par les modèles et les probabilités d'appartenir à chaque classe.

3.5.7 Analyse des enregistrements mal classés

Bien que les performances des deux modèles que l'on a obtenus soient excellentes, il subsiste tout de même quelques enregistrements dont la classe a été mal prédite par les deux modèles. Dans cette partie, nous analyserons les caractéristiques de ces EGM.

Premièrement, au vu des valeurs très dispersées de l'histogramme de la figure 24, la taille initiale des enregistrements ne semble pas liée au fait de mal prédire la classe d'un enregistrement. Par corollaire, la quantité de padding appliquée ne semble pas une cause de mauvaise classification.

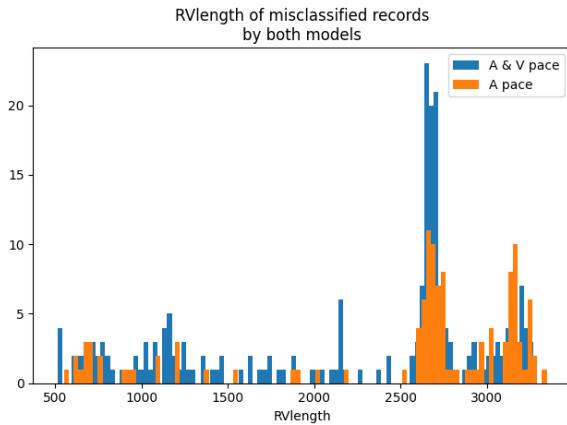


Figure 24 : Histogramme des tailles des enregistrements mal classés par les deux meilleurs modèles.

Rappelons ensuite que sur le signal électrique d'un EGM, une FA est caractérisée par un tracé anarchique sur la voie auriculaire de l'EGM. L'activité électrique anarchique de l'oreillette droite affecte également celle du ventricule droit qui voit ses contractions s'accélérer par rapport à la normale . Ces contractions rapides du ventricule se manifestent sur les tracés par une fréquence plus élevée. La majorité des enregistrements présentant bien une FA ressemblent à ceci :

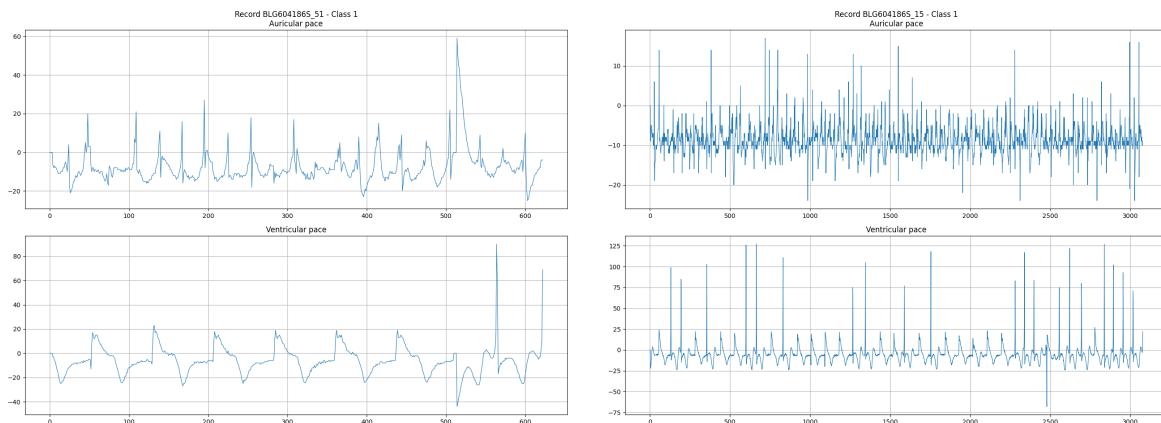


Figure 25 : Exemple d'enregistrements présentant une FA ayant été bien discriminés par les deux meilleurs modèles.

On retrouvera dans les enregistrements de classe 1 (FA) qui ont été prédit en classe 0 (bruit) un signal électrique de la voie auriculaire très peu bruité. Les deux enregistrements du dessous présentent cette particularité.

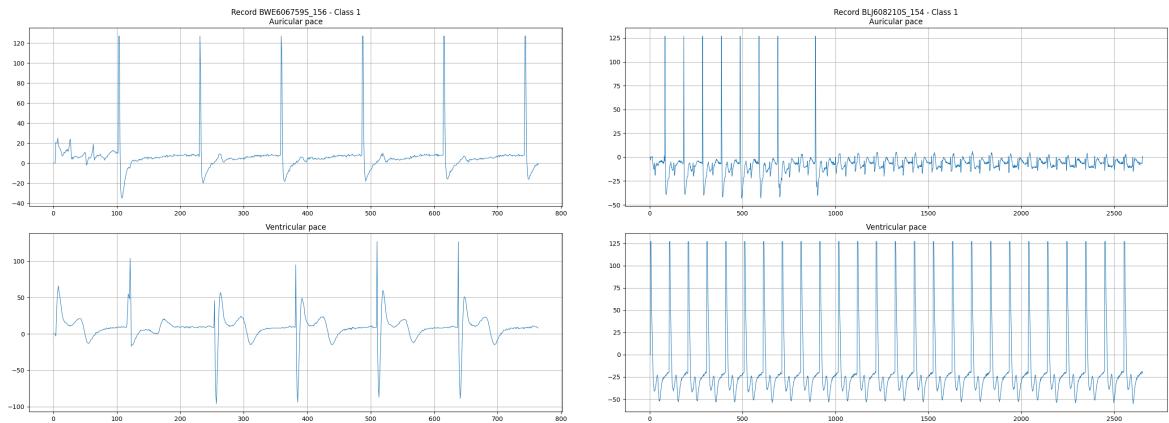


Figure 26 : Exemple d'enregistrements présentant une FA n'ayant pas été bien discriminés par les deux meilleurs modèles.

On regarde maintenant les enregistrements de classe 0. Ils présentent en général un signal électrique de la voie auriculaire très bruité, bien plus que ceux de classe 1, ce qui peut effectivement faire penser à une FA. On le remarque bien dans la figure 27 ci-dessous.

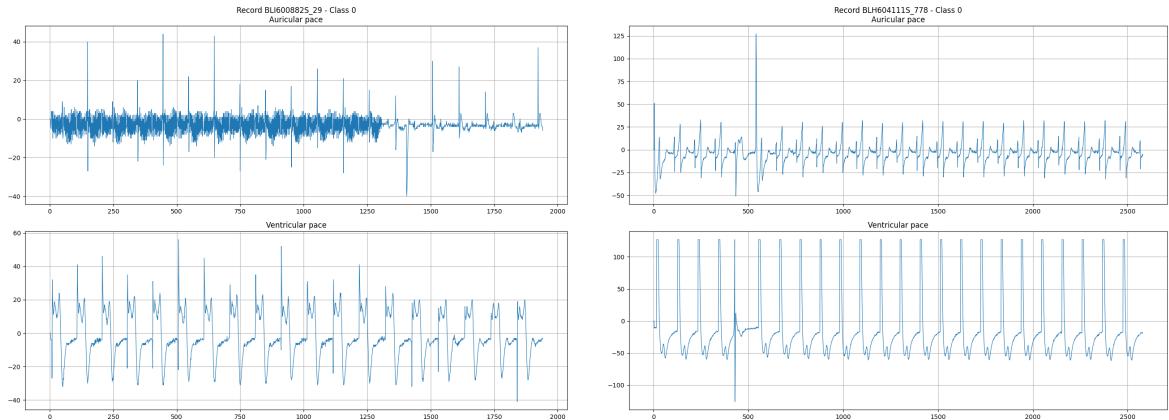


Figure 27 : Exemple d'enregistrements ne présentant pas de FA ayant été bien discriminés par les deux meilleurs modèles.

On remarque que certains enregistrements dont la classe a été mal prédite tel celui de gauche ont un signal de la voie ventriculaire rapide. Comme les patients desquels sont issus les enregistrements de classe 0 peuvent présenter d'autres anomalies et/ou pathologies cardiaques, ici non connues, celles-ci peuvent donc être confondues par les modèles avec une FA.

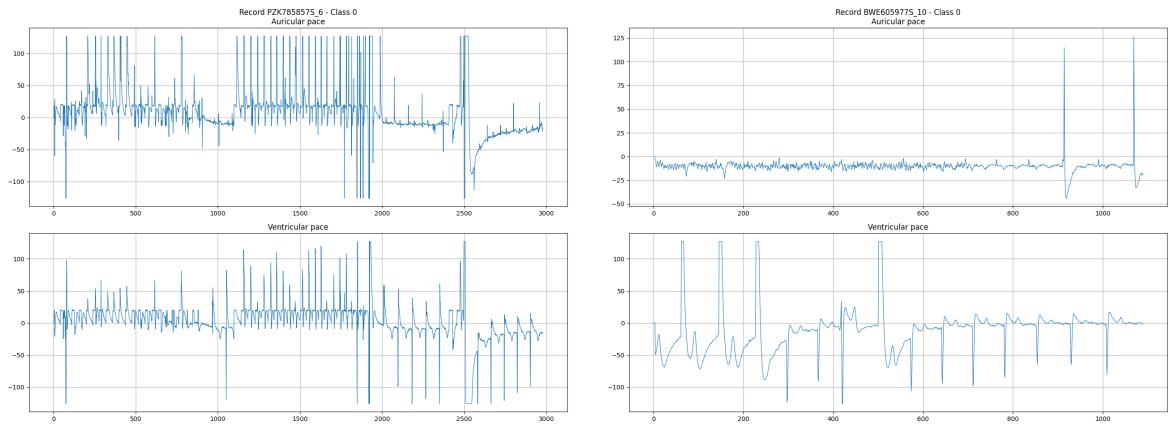


Figure 28 : Exemple d'enregistrements ne présentant pas de FA qui n'ont pas été bien discriminés par les deux meilleurs modèles.

3.6 Détection hors distribution par incertitudes de prédiction

Une seconde approche a été abordée pendant ce stage pour discriminer les enregistrements d'une classe par rapport à l'autre : la détection hors distribution par incertitude de prédiction.

Usuellement, un modèle entraîné à faire de la classification binaire va prédire qu'une donnée appartient à l'une des 2 classes sur lesquelles il a été entraîné, quelle que soit la donnée. En lui passant une donnée qui n'appartient aucune classe, il l'a prédira quand même en toute confiance dans l'une des 2 classes. Par exemple, un modèle entraîné à reconnaître un chat ou un chien sur une image prédira une image de grenouille en classe "chien" ou "chat" avec beaucoup de confiance.

La détection de données hors distribution permet de gérer ce genre de cas. Ici, on va utiliser les incertitudes de prédiction comme moyen de prédire si une donnée est hors distribution ou non. Cette approche se base sur l'article de *Kamil Ciosek et al.*^[11] où les auteurs parviennent à estimer les incertitudes de prédiction d'un modèle de réseau de neurones à l'aide d'un modèle a priori aléatoire, sans les sous-estimer.

Notons $x = (x_1, \dots, x_n)$ les données d'une (ou plusieurs) classe qui vont constituer un ensemble de données d'apprentissage et $x^* = (x_1^*, \dots, x_K^*)$ un ensemble de données test. $B = (x, x^*)$ constitue une base de données.

Un premier modèle de réseau de neurones est d'abord créé avec des poids initialisés aléatoirement : c'est le Prior (modèle a priori). Ce modèle, qui ne sera jamais entraîné, va renvoyer, pour chaque donnée x_i un vecteur z_i , d'une certaine taille M . L'ensemble $z = (z_1, \dots, z_n)$ représente alors la distribution du modèle Prior sur ces données d'apprentissage.

Ensuite on crée un second modèle de réseau de neurones, le Posterior (modèle a posteriori), de telle façon qu'il puisse représenter le premier. En pratique il s'agit d'un modèle dont l'architecture est celle du Prior avec plusieurs couches supplémentaires. Il renvoie donc

pour chaque x_i un vecteur \hat{z}_i aussi de taille M . Ce modèle va être entraîné sur les données d'apprentissage à renvoyer les mêmes sorties que le Prior. On projette par ce modèle Prior la densité de probabilité de la classe (ou des classes) des données d'apprentissage dans un espace de dimension M , et on apprend avec le modèle Posterior à modéliser cette projection de la densité de probabilité

Pour chaque donnée de la base B à disposition, on va mesurer la distance entre les vecteurs z_i et \hat{z}_i qui donnera alors l'incertitude de prédiction $\hat{\sigma}_i^2$ par rapport à la distribution des données d'apprentissage dans cet espace de dimension M .

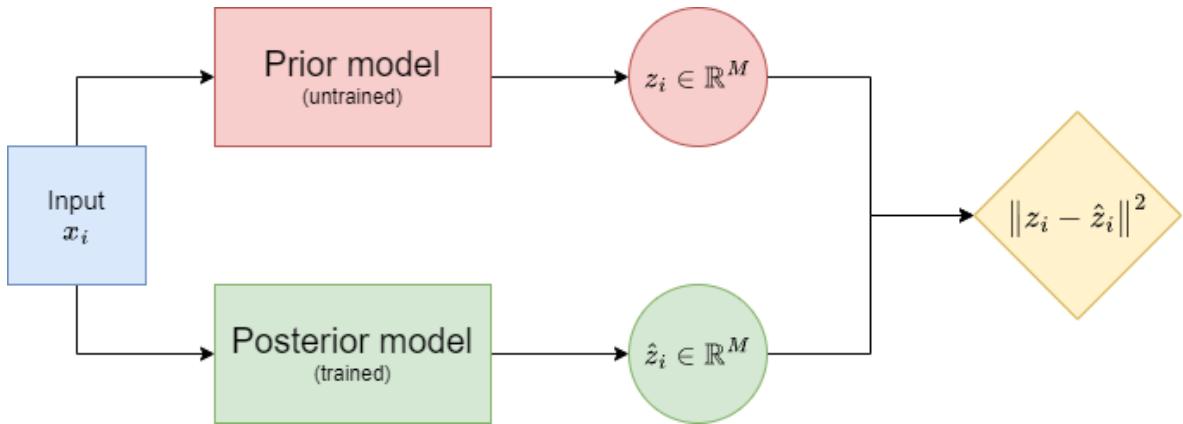


Figure 29 : Estimation de l'incertitude de prédiction d'une donnée x_i par le modèle Posterior entraîné.

À partir de ces incertitudes, on peut faire de la détection hors distribution. Puisqu'on a entraîné notre modèle a posteriori à apprendre la distribution du modèle a priori sur les données d'une ou de certaines classes, on va calculer les incertitudes de prédictions pour toutes les données dont on dispose. On peut supposer que le modèle Posterior aura des incertitudes plus grandes sur les données éloignées de la distribution des données d'apprentissage. En se fixant un seuil de décision α , on va donc prédire qu'une donnée x_i est hors distribution si son incertitude $\hat{\sigma}_i^2$ est supérieure au seuil α .

3.6.1 Application sur le jeu de données MNIST

Dans un premier temps, on cherchera à confirmer l'efficacité de la méthode sur un jeu de données standard : ici le jeu de données MNIST. Ce dernier se compose de 70,000 images des chiffres 0 à 9 manuscrits, appartenant à 10 classes différentes. Le jeu de données est divisé en un ensemble d'apprentissage de 60,000 images et un ensemble de test de 10,000 images. Les 10 classes sont équitablement réparties dans chacun de ces ensembles. On va entraîner un modèle Posterior sur les sorties d'un Prior pour les images de classe 4 seulement. On va donc vouloir discriminer les images n'étant pas de classe 4. Pour les modèles Prior et Posterior, on reprend les architectures utilisées par les auteurs de l'article, représentées sur la figure ci-dessous.

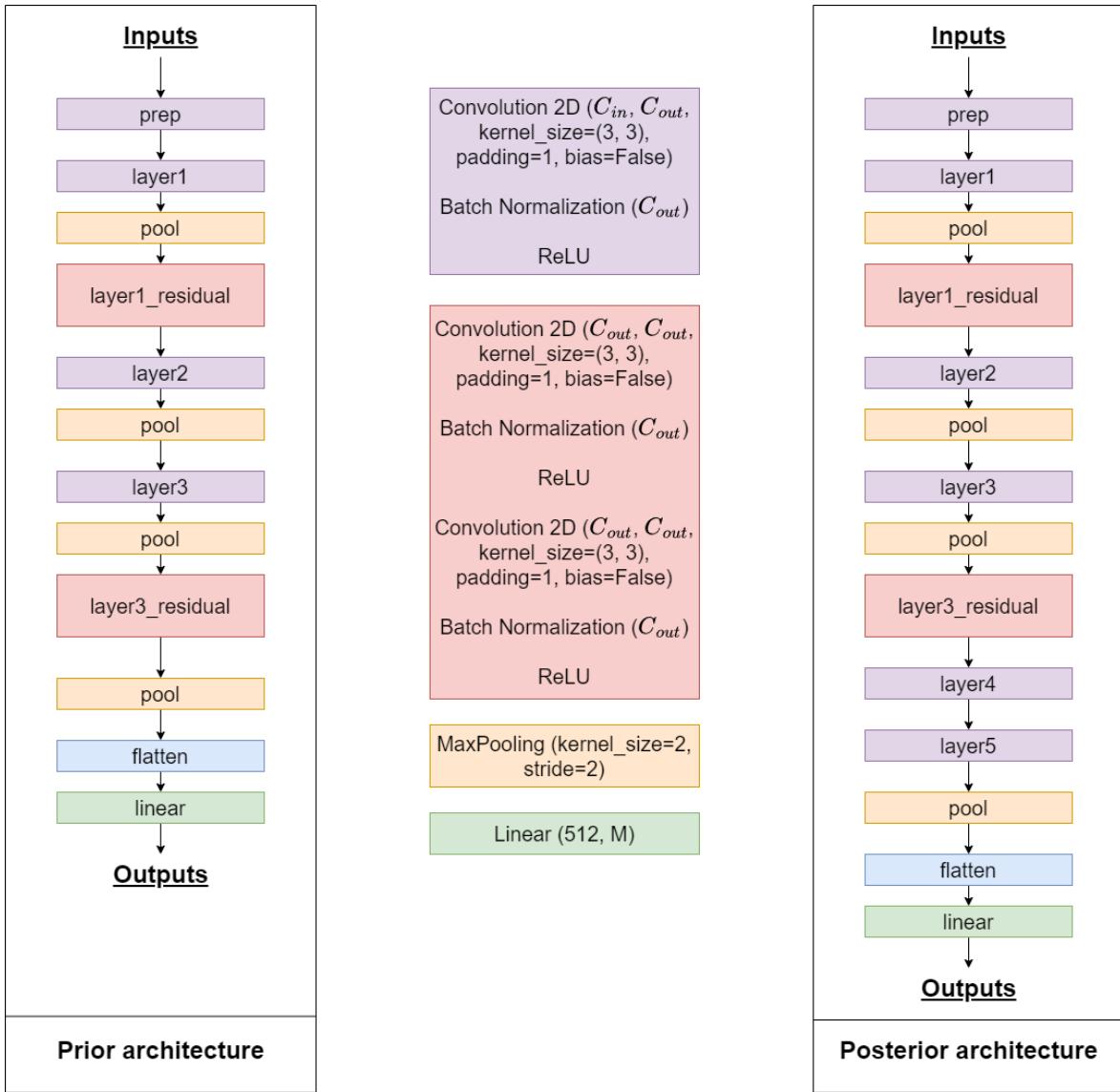


Figure 30 : Architectures des modèles Prior et Posterior utilisées par Kamil Ciosek et al.^[11] dans leurs codes sur GitHub^[12].

La taille des vecteurs de sortie des deux modèles est fixée à 512, comme vu dans le code des auteurs^[12]. Ce serait un paramètre à optimiser en perspective du travail présenté ici. Comme on fait de la régression et non plus de la classification avec nos deux modèles, la fonction objectif change : on prend la MSE définie par :

$$MSE(z_i, \hat{z}_i) = \frac{1}{M} \sum_{k=1}^M \|z_{i,k} - \hat{z}_{i,k}\|^2$$

On prendra une taille de batch à 128 et l'algorithme Adam, avec les paramètres par défaut, pour entraîner le Posterior. Afin d'éviter le surapprentissage, on implémente une méthode

d'Early Stopping sur la courbe de validation. Pour vérifier que le modèle Posterior apprend bien, on mesure la valeur de la fonction objective sur l'ensemble de données d'apprentissage (training loss) mais également sur un ensemble de données de validation (validation loss). Le modèle n'apprend pas sur cet ensemble de validation qui, à l'instar de l'ensemble d'apprentissage, ne contient que des images de classe 4. On va également calculer la loss sur des ensembles ne contenant que des images d'une classe (que des 1, que des 2, etc.). Les courbes de valeurs moyennes de la loss calculées sur les images des différentes classes sont présentées dans l'annexe E.

Pour chacune des classes, on constate une différence nette entre la MSE moyenne calculée sur les images de classe 4 et la MSE moyenne calculée sur les images des autres classes pendant l'entraînement. On s'attend donc à avoir des incertitudes de prédictions pour les images de classe 4 qui sont moindres comparées aux incertitudes pour les autres classes. On applique alors le procédé illustré dans la figure 29 pour toutes les images de l'ensemble d'apprentissage de MNIST et on calcule les incertitudes avec la MSE. On obtient les valeurs suivantes :

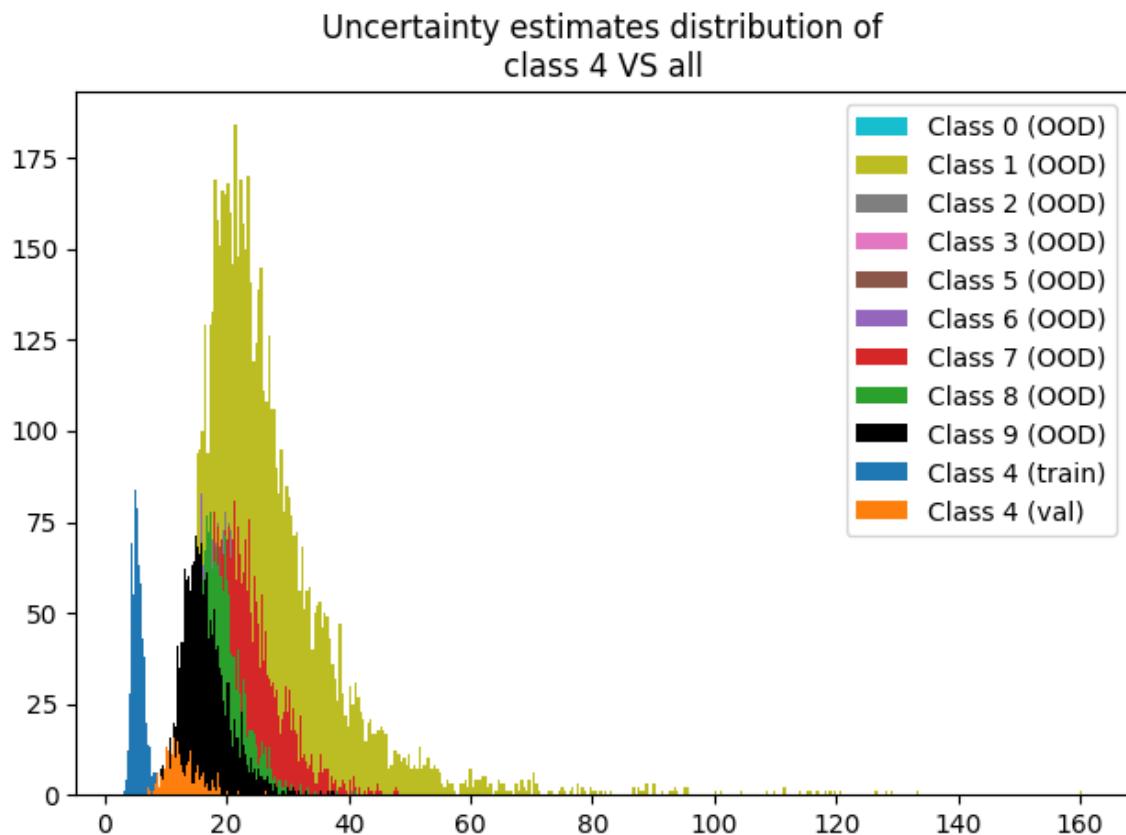


Figure 31 : Histogramme des incertitudes de prédictions du modèle Posterior pour chaque classe. (train) désigne les 4674 images de classe 4 ayant servi à l'apprentissage (données que le modèle a vu pendant l'entraînement). (val) désigne les 1168 images de classe 4 ayant servi à la validation qui, comme les images des autres classes, n'ont jamais été vues par le modèle pendant l'entraînement.

Comme espéré, les incertitudes de prédiction pour les images de classe 4 sont plus faibles que les incertitudes pour les images des autres classes. La différence entre la distribution des incertitudes pour les images de classe 4 de validation et celle pour les incertitudes des autres images semble en revanche plutôt faible.

À noter qu'en moyenne la MSE diffère également entre les images de classe 4 ayant servi à l'apprentissage du modèle Posterior et celles ayant servi à la validation à partir d'un certain nombre d'epochs. En poussant l'entraînement plus loin on a pu observer que cette différence s'accroît avec le nombre d'epochs, ce qui indique le surapprentissage du modèle Posterior sur les données d'apprentissage. C'est la raison pour laquelle on a préféré implémenté un procédé d'Early Stopping.

Malgré cela, nous pouvons, au vu de ces distributions, chercher un seuil $\hat{\alpha}$ à partir duquel on pourra estimer qu'une donnée est dans la distribution de la classe des données d'apprentissage si ou si elle est hors distribution. Pour chercher ce seuil, on prend une grille de 10,000 valeurs comprises entre le minimum et le maximum des incertitudes calculées précédemment. Notons pour chaque image x_i son incertitude de prédiction estimée $\hat{\sigma}_i^2$. Pour chaque valeur de seuil α_t , on classe une image :

- “dans la distribution” (positif) si $\hat{\sigma}_i^2 < \alpha_t$;
- “hors distribution” (négatif) dans le cas contraire.

De façon similaire à une classification binaire, on définit deux labels pour les images : “dans la distribution” pour les images de classe 4 et “hors distribution” pour les images qui ne sont pas de classe 4.

On peut calculer alors le taux de vrais positifs et le taux de faux positifs pour chaque seuil. On obtient alors la courbe ROC suivante :

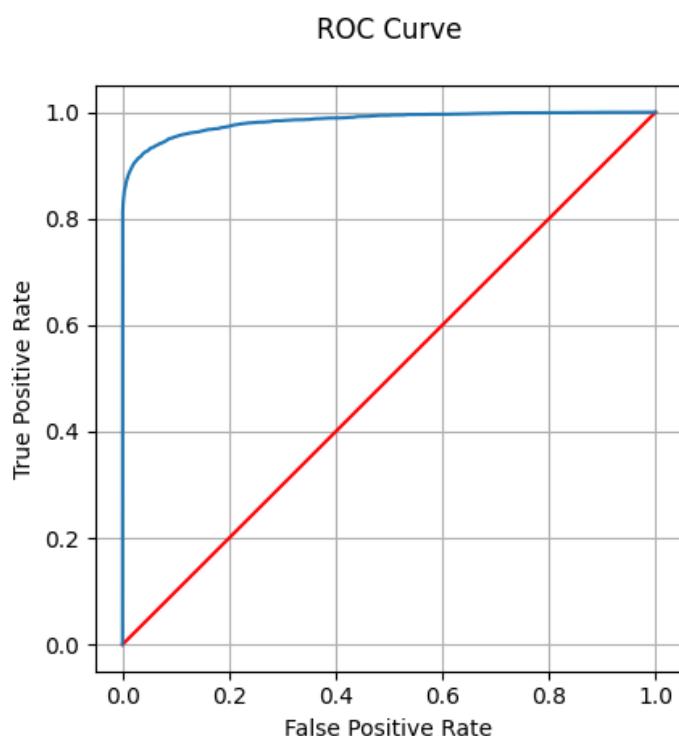


Figure 32 : Courbe ROC pour 10,000 valeurs de seuil de décision pour la détection hors distribution.

Au vu de cette courbe, on constate qu'il existe une valeur qui permet de très bien discriminer les données de la même classe que les données d'apprentissage des autres. Cependant aucune ne permet de les discriminer parfaitement puisqu'aucun de point de la courbe ROC n'atteint le point de coordonnées $(0, 1)$. Pour choisir la meilleure valeur pour le seuil de décision, on prend ici la valeur pour laquelle $TPR - FPR$ est maximal où TPR est le taux de vrais positifs et FPR le taux de faux positifs. On trouve comme seuil optimal $\hat{\alpha} = 12.7391$

Avec ce seuil optimal, on effectue une détection hors distribution avec le même procédé décrit plus haut. On trouve alors les résultats suivants sur l'ensemble d'apprentissage du jeu MNIST :

Vrais Positifs	Faux Négatifs	Faux Positifs	Vrais Négatifs	Accuracy
5,342	500	1,627	52,531	96.455 %

Table 3 : Résultats de la détection hors distribution sur l'ensemble de données d'apprentissage du jeu MNIST (total : 60,000 images) avec le seuil de décision à 12.7391.

On applique la détection hors distribution au jeu de données de test du jeu MNIST (10,000 images) avec ce même seuil et on obtient les résultats suivants :

Vrais Positifs	Faux Négatifs	Faux Positifs	Vrais Négatifs	Accuracy
594	388	1,260	7,758	83.52 %

Table 4 : Résultats de la détection hors distribution sur l'ensemble de données de test du jeu MNIST (total : 10,000 images) avec le seuil de décision à 12.7391.

On applique également cette méthode avec des données d'un tout autre jeu de données : CIFAR10, qui un ensemble de 50,000 images d'animaux ou de véhicules. Les résultats sont les suivants :

Vrais Positifs	Faux Négatifs	Faux Positifs	Vrais Négatifs	Accuracy
0	0	0	50,000	100 %

Table 5 : Résultats de la détection hors distribution sur l'ensemble de données CIFAR10 (total : 50,000 images) avec le seuil de décision à 12.7391.

Au vu des résultats présentés dans les tables 3 et 4, la méthode de détection hors distribution avec les incertitudes de prédictions d'un modèle Posterior permet de bien discriminer les données d'une classe (ici la classe 4) des données des autres classes d'un

même jeu (ici MNIST). En choisissant notre seuil de décision sur l'ensemble d'apprentissage du jeu MNIST, on arrive à bien détecter pour **94.6071%** des images du jeu entier si elles sont bien dans la distribution des données d'apprentissage du modèle Posterior ou si elles sont hors distribution. Si toutes les images du jeu MNIST entier avaient été naïvement détectées comme étant hors distribution (au vu du faible nombre de données d'apprentissage), on aurait eu une accuracy de 90.2772%. Donc notre méthode est meilleure qu'une détection naïve. De plus, on arrive à détecter que toutes les images d'un autre jeu de données (ici CIFAR10), très différentes des images du jeu MNIST, sont bel et bien hors distribution.

On montre ainsi que cette approche fonctionne pour discriminer des données d'une classe de données d'autres classes, voire d'un autre jeu de données.

3.6.2 Application sur le jeu de données des EGM

On va implémenter à présent l'approche par incertitudes de prédictions sur notre jeu de données constitué des 10,172 EGMs. On souhaite discriminer les enregistrements de classe 1 des autres puisqu'on veut discriminer le signal caractéristique d'une FA d'un bruit. On entraînera donc un modèle Posterior à renvoyer les mêmes sorties qu'un modèle Prior sur des données de classe 1 (FA).

Pour cela, nous allons utiliser des versions légèrement modifiées des meilleures architectures trouvées avec l'approche précédente (figures 21 et 22). Les modèles Prior auront les architectures de ces meilleurs modèles mais avec seulement deux couches denses à la fin du réseau. Les modèles Posterior auront une architecture avec deux couches de convolution et deux couches denses de plus que les Prior. Comme pour la première approche, on va appliquer un zero-padding à 8192 sur les signaux des EGM. Comme avec les données MNIST, on prend l'algorithme Adam avec ses paramètres par défaut, la MSE comme loss et une taille de sortie des réseaux à 512. On reprend le découpage apprentissage/test de la première approche. De plus, une partie des enregistrements de classe 1 dans l'ensemble d'apprentissage seront placés dans un ensemble de validation. Afin d'éviter un surapprentissage des modèles Posterior, on implémente un procédé d'Early Stopping.

On considère d'abord un couple de modèle Prior/Posterior dont l'architecture est calquée sur le meilleur modèle prenant les deux voies en entrée du chapitre précédent. On visualise les valeurs de la loss sur les données d'apprentissage, de validation mais également sur les données de classe 0 au cours de l'entraînement du modèle Posterior.

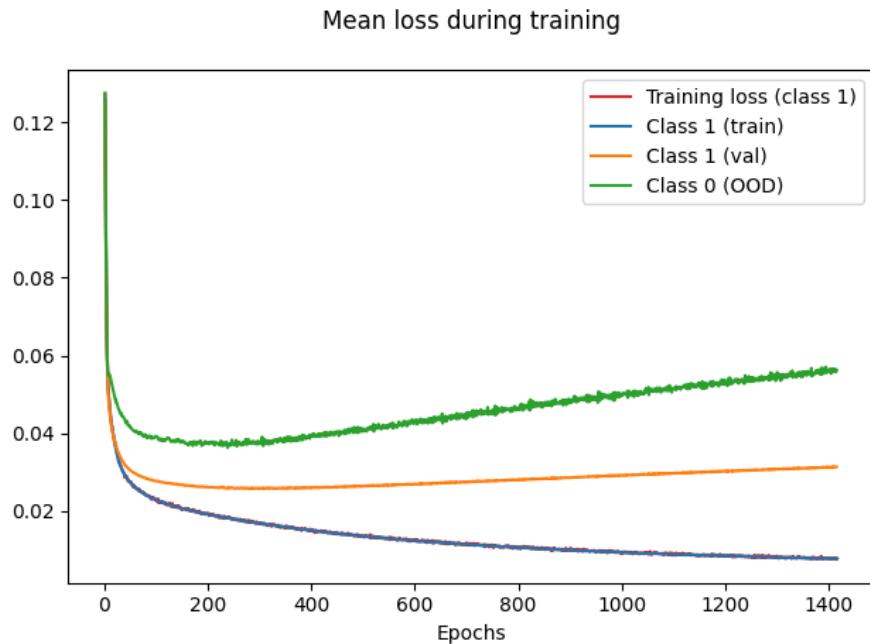


Figure 33 : Courbe des valeurs moyenne de la loss calculée sur les données d'apprentissage, de validation et les données de classe 0.

Comme précédemment, on retrouve une MSE moyenne plus faible pour les données d'apprentissage que pour les autres données. Par contre, la différence n'est ici que de quelques centièmes contrairement aux écarts de MSE de l'ordre de la dizaine dans le cas du jeu de données MNIST.

Une fois le modèle Posterior entraîné, on estime les incertitudes de prédictions sur les données d'apprentissage, de validation et de classe 0. On affiche la distribution de ces incertitudes dans les histogrammes ci-dessous.

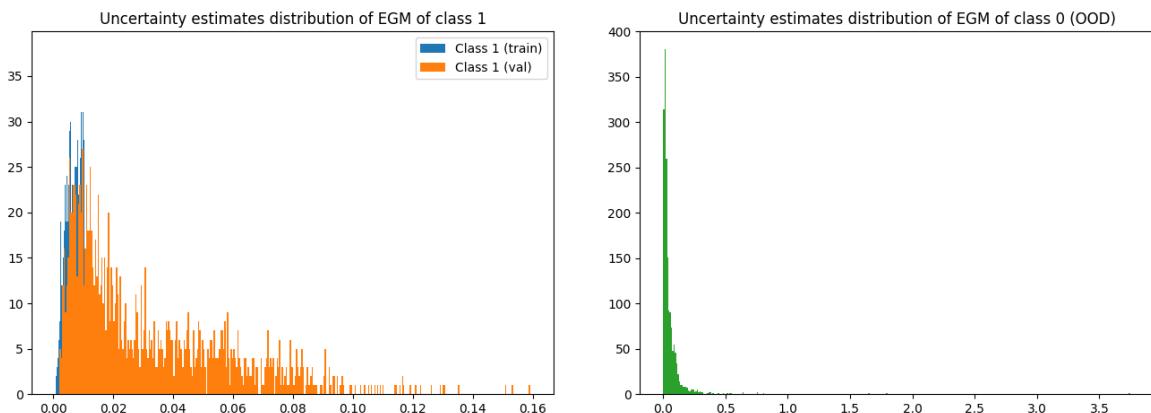


Figure 34 : Histogrammes des incertitudes de prédition du modèle Posterior pour les enregistrements de classe 1 (à gauche) et de classe 0 (à droite).

Ici, on observe une faible différence entre les distributions des incertitudes pour les enregistrements de classe 1 et ceux de classe 0. La détection hors distribution est tout de même possible. On réitère les opérations effectuées dans la partie 3.6.1. On cherche alors un seuil de décision pour cela parmi une grille d'un milliers de valeurs comprises entre le minimum et le maximum de ces incertitudes. On obtient la courbe ROC suivante :

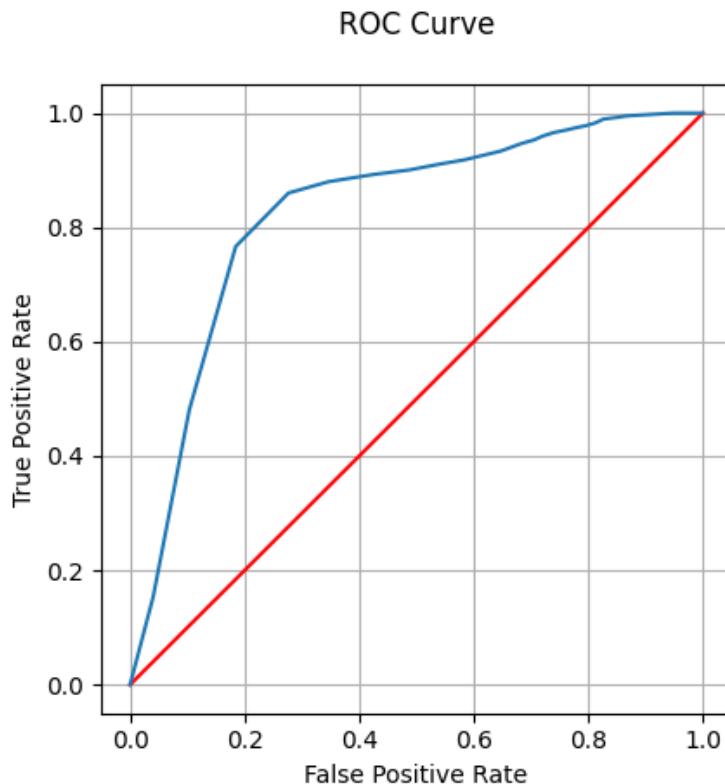


Figure 33 : Courbe ROC pour 1,000 valeurs de seuil de décision pour la détection hors distribution.

Au vu de cette courbe ROC, il ne semble y avoir aucune valeur pour laquelle on peut avoir d'excellents taux de vrais positifs et de faux positifs. On prend comme seuil la valeur qui maximise $TPR - FPR$, ici $\hat{\alpha} = 0,0158$. Les résultats de la détection hors distribution avec ce seuil sont présentés dans la table ci-dessous :

Ensemble de données	Vrais Positifs	Faux Négatifs	Faux Positifs	Vrais Négatifs	Accuracy
Apprentissage	5,545	901	468	1,224	83.1776 %
Test	636	975	138	285	45.2802 %

Table 6 : Résultats de la détection hors distribution sur l'ensemble de données des EGM avec le meilleur seuil à 0,0158.

Sur l'ensemble d'apprentissage de notre jeu de données, cette méthode ne nous permet de discriminer qu'à peine 83% des enregistrements présentant une FA des enregistrements n'en présentant pas. Avec une détection naïve qui détecterait tous les enregistrements comme étant dans la distribution, on aurait une accuracy de 79.2076%. Or l'accuracy sur les données d'apprentissage est à peine meilleure et celle sur les données test est bien pire.

On réitère les mêmes opérations avec un couple de modèle Prior/Posterior dont l'architecture est calquée sur celle du meilleur modèle ne prenant en entrée que la voie auriculaire d'un EGM du chapitre précédent.

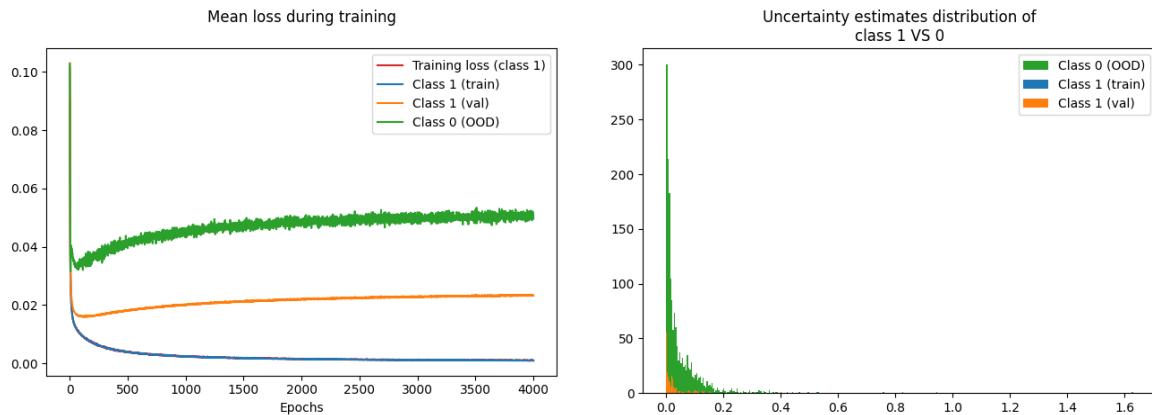


Figure 35 : À gauche la courbe des valeurs moyenne de la loss calculée sur les données d'apprentissage, de validation et les données de classe 0. À droite l'histogramme des incertitudes de prédiction du modèle Posterior pour les enregistrements de classe 1 et 0.

L'allure des courbes d'apprentissage et la distribution des incertitudes sont semblables à ce qu'on a obtenu avant. On cherche maintenant le seuil de décision parmi une grille d'un millier de valeurs comprises entre le minimum et le maximum des incertitudes estimées. On obtient la courbe ROC suivante :

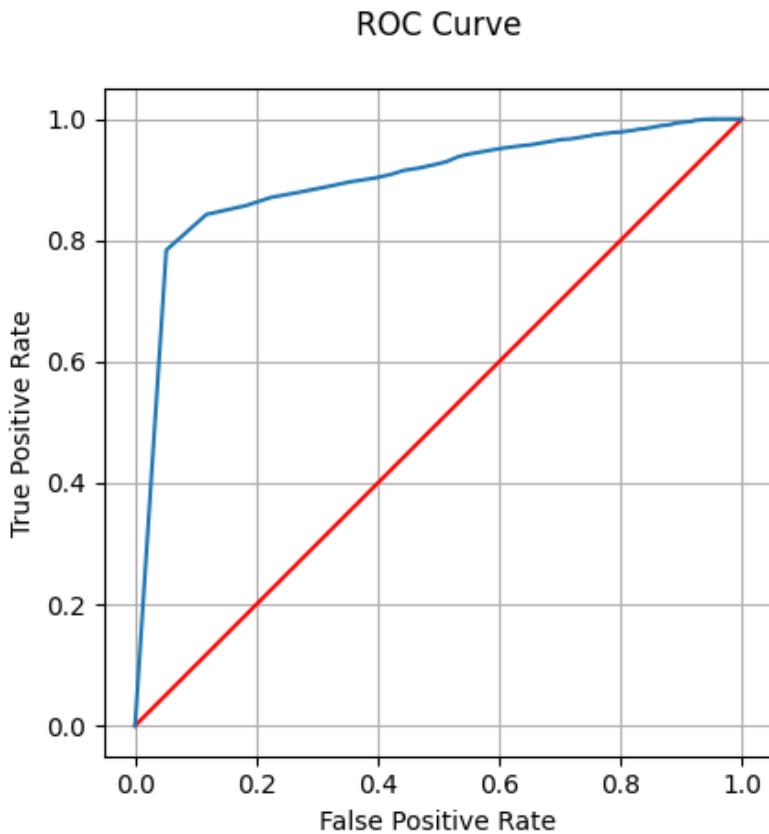


Figure 36 : Courbe ROC pour 1,000 valeurs de seuil de décision pour la détection hors distribution.

De même qu'avec le couple Prior/Posterior précédent, il ne semble y avoir aucune valeur pour laquelle les taux de vrais positifs et de faux positifs sont excellents. La valeur qui maximise $TPR - FPR$ vaut $\hat{\alpha} = 0,0017$. La détection hors distribution donne les résultats suivants sur notre jeu de données :

Ensemble de données	Vrais Positifs	Faux Négatifs	Faux Positifs	Vrais Négatifs	Accuracy
Apprentissage	5,053	1,393	87	1,605	81.8137 %
Test	98	1513	37	386	23.7954 %

Table 7 : Résultats de la détection hors distribution sur l'ensemble de données des EGM avec le meilleur seuil à 0,0017.

Avec le couple Prior/Posterior dont l'architecture est calquée sur le meilleur modèle à une voie, on obtient des résultats encore pire qu'avec le couple précédent.

Bien que sur le jeu de données MNIST la méthode de détection hors distribution avec les incertitudes de prédiction a donné des résultats satisfaisants, son implémentation sur notre ensemble d'enregistrements d'EGM donne des résultats au plus médiocres. De plus, avec cette seconde approche, on arrive moins bien à discriminer les enregistrements présentant une FA de ceux n'en présentant pas par rapport à la première approche.

Au vu des incertitudes estimées, il est tout à fait possible que les densités de probabilité des enregistrements soient trop similaires à cause du zero-padding à 8192 ou encore à cause de la taille de la sortie. Cependant, par manque de temps, nous n'avons pas pu explorer plus en profondeur les raisons qui font que cette seconde approche n'ait pas fonctionné aussi bien qu'espéré sur notre jeu de données.

4. Conclusion

Dans un contexte où il y a de plus en plus d'appareils électroniques cardiaques implantés, la surveillance des mauvaises performances de détection de ces appareils devient de plus en plus importante. En particulier, pour la détection à tort de fibrillation auriculaire qui peut avoir de lourdes conséquences pour un patient.

Au cours de ce stage, j'ai pu développer un algorithme, qui détecte automatiquement si un enregistrement d'un EGM présente une FA ou non. J'ai dû le concevoir avec une base de données déséquilibrée, composées de 10,172 enregistrements où l'appareil implanté a détecté une FA. Parmi ceux-ci, après une vérification manuelle par trois médecins, 20% sont des détections à tort.

Pour développer l'algorithme, nous avons utilisé deux approches : la première consiste en l'utilisation d'un modèle de réseau de neurones convolutionnel et la seconde par la modélisation par un réseau de la densité de probabilité de la classe 'FA' et les incertitudes de prédiction d'un enregistrement par rapport à cette distribution.

Avec la première approche, j'ai pu construire deux modèles dont l'architecture est inspirée des modèles ResNet et qui peut prédire dans plus de 95% des cas si une FA est présente ou non sur un enregistrement. Ces modèles prennent en entrée soit la voie auriculaire d'un enregistrement uniquement, soit les deux voies auriculaire et ventriculaire. Au vu des performances, il apparaît que l'architecture ne prenant qu'une seule voie est meilleure. On remarque aussi que les enregistrements présentant une FA qui sont peu bruités se retrouvent mal classés par les deux modèles. Les modèles ont ensuite été intégrés à une application utilisant une interface web via la bibliothèque Flask qui s'en sert pour discriminer automatiquement les enregistrements présentant une FA de ceux n'en présentant pas.

La deuxième approche se base sur l'apprentissage de la distribution de probabilité d'un modèle *a priori* par un modèle *a posteriori* décrite dans l'article de *Kamil Ciosek et al.*¹¹. Afin de vérifier sa validité, je l'ai testé sur le jeu de données MNIST de chiffres manuscrits. En reprenant les modèles utilisés par les auteurs de l'article sur lequel se base cette approche, j'ai cherché à discriminer les images de 4 des autres images. Les incertitudes de prédiction du modèle *a posteriori* sont estimées par l'écart, ici la MSE, entre ses sorties et les sorties du modèle *a priori*. L'objectif ensuite étant de pouvoir discriminer via ces incertitudes les images de 4 des autres. En choisissant un seuil, les images ayant une incertitude excédant ce seuil sont considérées comme étant hors distribution. Par cette méthode, 94% des données du jeu MNIST sont bien détectées comme étant dans la distribution ou hors distribution.

J'ai ensuite implémenté cette méthode sur le jeu de données des EGM pour discriminer les enregistrements présentant une FA de ceux n'en présentant pas. Pour cela, j'ai repris et adapté les architectures des meilleurs modèles trouvées dans la première approche. Les résultats obtenus ne sont pas satisfaisants, à peine plus de 80% des données sont correctement détectées dans ou hors distribution.

Ainsi, via les modèles de réseau neuronal résiduel trouvés avec la première approche, j'ai pu développer un algorithme capable de détecter automatiquement les FA sur l'EGM du CIED d'un patient. Cet algorithme est utilisé dans une application flask qui permet de présenter les enregistrements 'suspects' pouvant correspondre à un bruit de sonde pour un diagnostic médical.

5. Références

- [1] Prévenir et guérir les maladies du rythme cardiaque - IHU Liryc. (2023, 4 janvier). IHU Liryc. <https://www.ihu-liryc.fr/prevenir-et-querir-les-maladies-du-rythme-cardiaque/>
- [2] Les pôles de recherche - IHU Liryc. (2023, 24 mai). IHU Liryc. <https://www.ihu-liryc.fr/les-poles-de-recherche/>
- [3] Inventer les traitements de demain - IHU Liryc. (2023, 6 janvier). IHU Liryc. <https://www.ihu-liryc.fr/inventer-les-traitements-de-demain/>
- [4] La prise en charge des patients au CHU de Bordeaux - IHU Liryc. (2023, 22 février). IHU Liryc. <https://www.ihu-liryc.fr/la-prise-en-charge-des-patients-au-chu-de-bordeaux/>
- [5] Les formations diplômantes - IHU Liryc. (2023, 5 mai). IHU Liryc. <https://www.ihu-liryc.fr/les-formations-diplomantes/>
- [6] Comprendre la fibrillation auriculaire ou atriale. (s. d.). ameli.fr | Assuré. [https://www.ameli.fr/assure/sante/themes/fibrillation-auriculaire/definition-facteurs-favorisants#:~:text=La%20fibrillation%20auriculaire%20est%20d%C3%A9finie.rapide%20des%20ventricules%20\(tachyarrhythmie\)](https://www.ameli.fr/assure/sante/themes/fibrillation-auriculaire/definition-facteurs-favorisants#:~:text=La%20fibrillation%20auriculaire%20est%20d%C3%A9finie.rapide%20des%20ventricules%20(tachyarrhythmie))
- [7] Chollet, F. (2017). Deep learning with python. Manning Publications.
- [8] Kingma, D.P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. CoRR, abs/1412.6980.
- [9] SGD — PyTorch 2.0 documentation. (n.d.). <https://pytorch.org/docs/stable/generated/torch.optim.SGD.html>
- [10] Sutskever, I., Martens, J., Dahl, G. E., & Hinton, G. E. (2013). On the importance of initialization and momentum in deep learning. Dans International Conference on Machine Learning (p. 1139-1147). <http://proceedings.mlr.press/v28/sutskever13.pdf>
- [11] Kamil Ciosek, Vincent Fortuin, Ryota Tomioka, Katja Hofmann, & Richard Turner (2020). Conservative Uncertainty Estimation By Fitting Prior Networks. In International Conference on Learning Representations.
- [12] microsoft. (n.d.). GitHub - microsoft/conservative-uncertainty-estimation-random-priors: Source code for paper Conservative Uncertainty Estimation By Fitting Prior Networks (ICLR 2020). GitHub. <https://github.com/microsoft/conservative-uncertainty-estimation-random-priors>

6. Annexes

Annexe A : Comparaison des différents algorithmes d'optimisation par rapport au F_1 score moyen

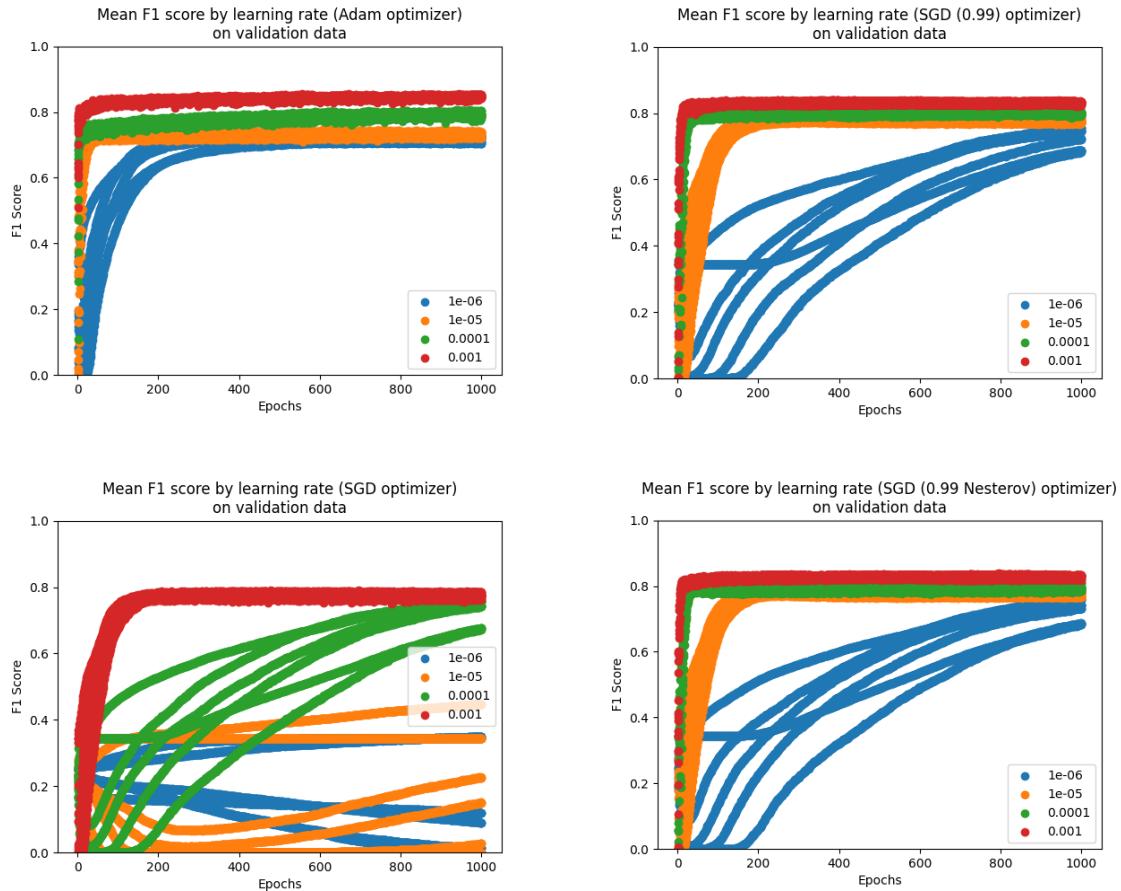


Figure 37 : F_1 scores moyens pour 25 initialisations aléatoires de l'architecture de la figure 16 par algorithme d'optimisation et par taux d'apprentissage, tous poids confondus.

Annexe B : Scores moyen des meilleures architectures pour différents ensembles d'apprentissage et de test.

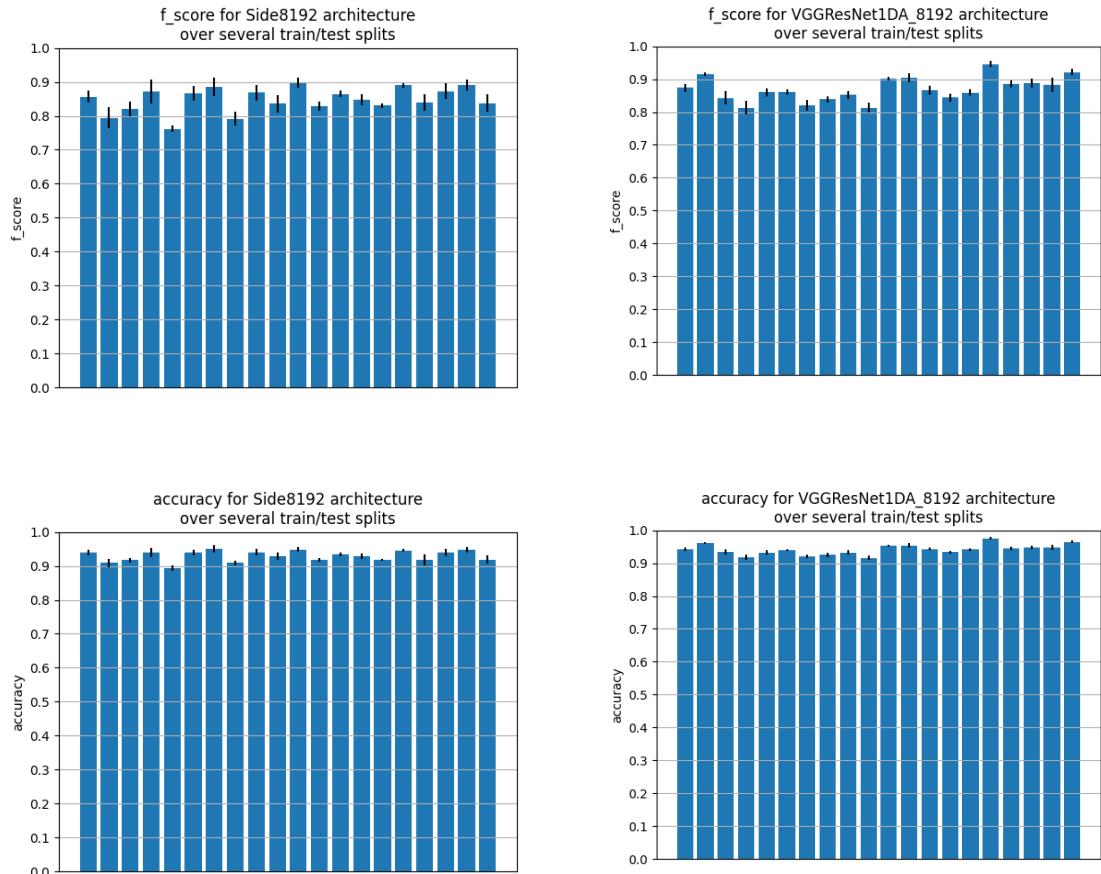


Figure 38 : F_1 scores moyen et accuracy moyenne des modèles avec la meilleure architecture à deux voies (à gauche) et la meilleure à une voie (à droite) entraînés sur différents ensembles d'apprentissage.

Annexe C : Résultats de la validation croisée pour la meilleure architecture prenant en entrée les deux voies d'un enregistrement.

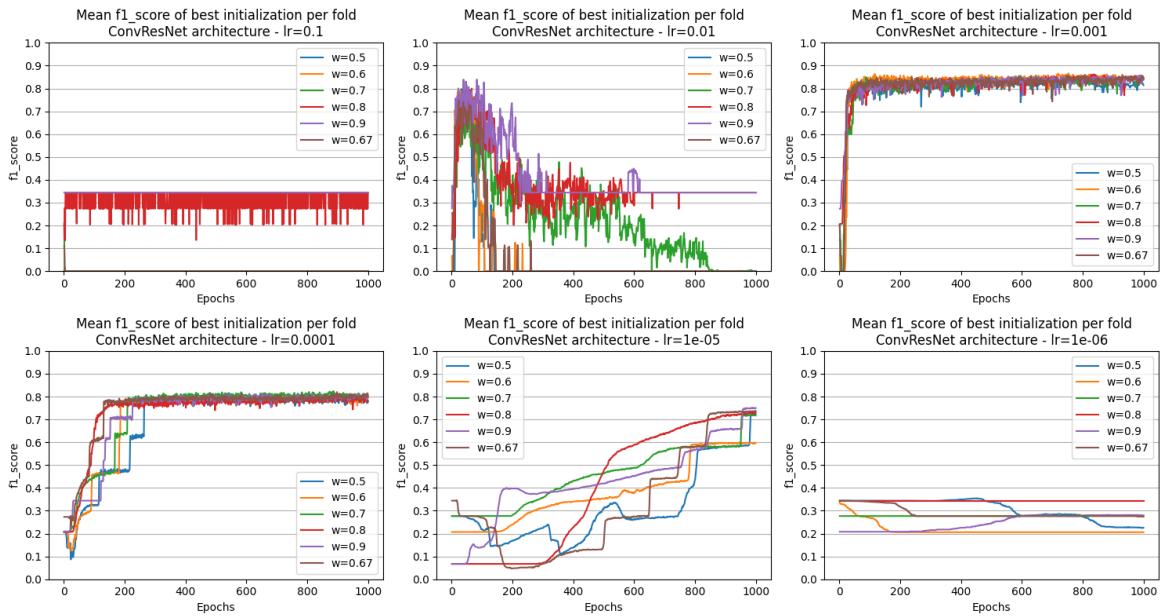


Figure 39 : F_1 scores moyens pour 25 initialisations aléatoires de l'architecture présentée en figure 21 par taux d'apprentissage et par poids associés à chaque classe.

Annexe D : Résultats de la validation croisée pour la meilleure architecture prenant en entrée la voie auriculaire d'un enregistrement.

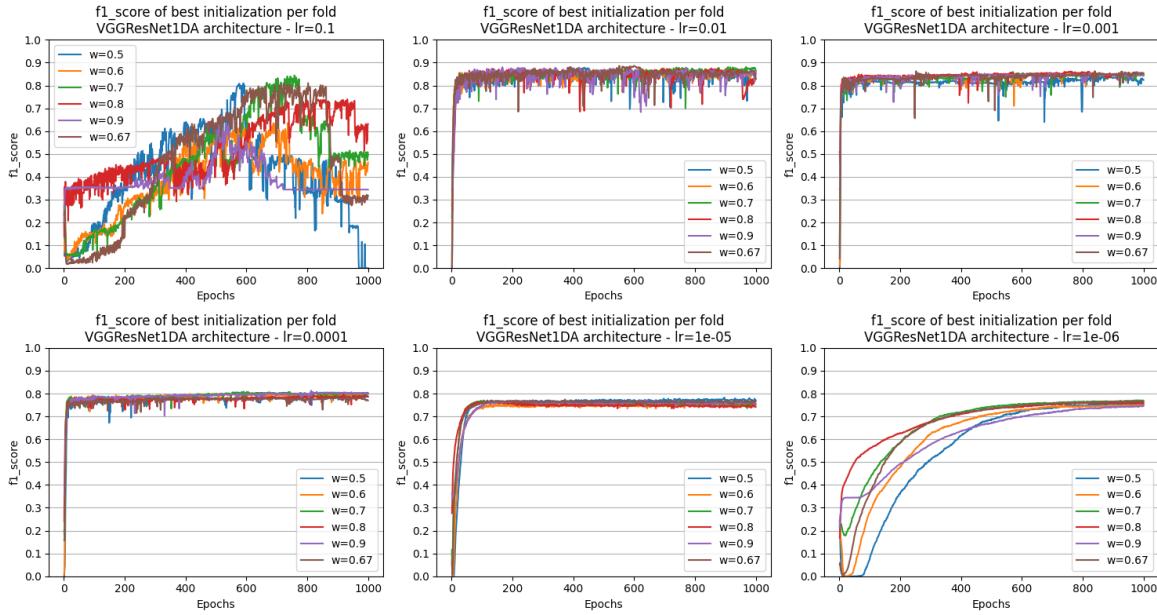


Figure 40 : F_1 scores moyens pour 25 initialisations aléatoires de l'architecture présentée en figure 22 par taux d'apprentissage et par poids associés à chaque classe.

Annexe E : Comparaison des valeurs de loss calculées sur les ensembles d'apprentissage, de validation et sur les images des autres classes du jeu de données MNIST.

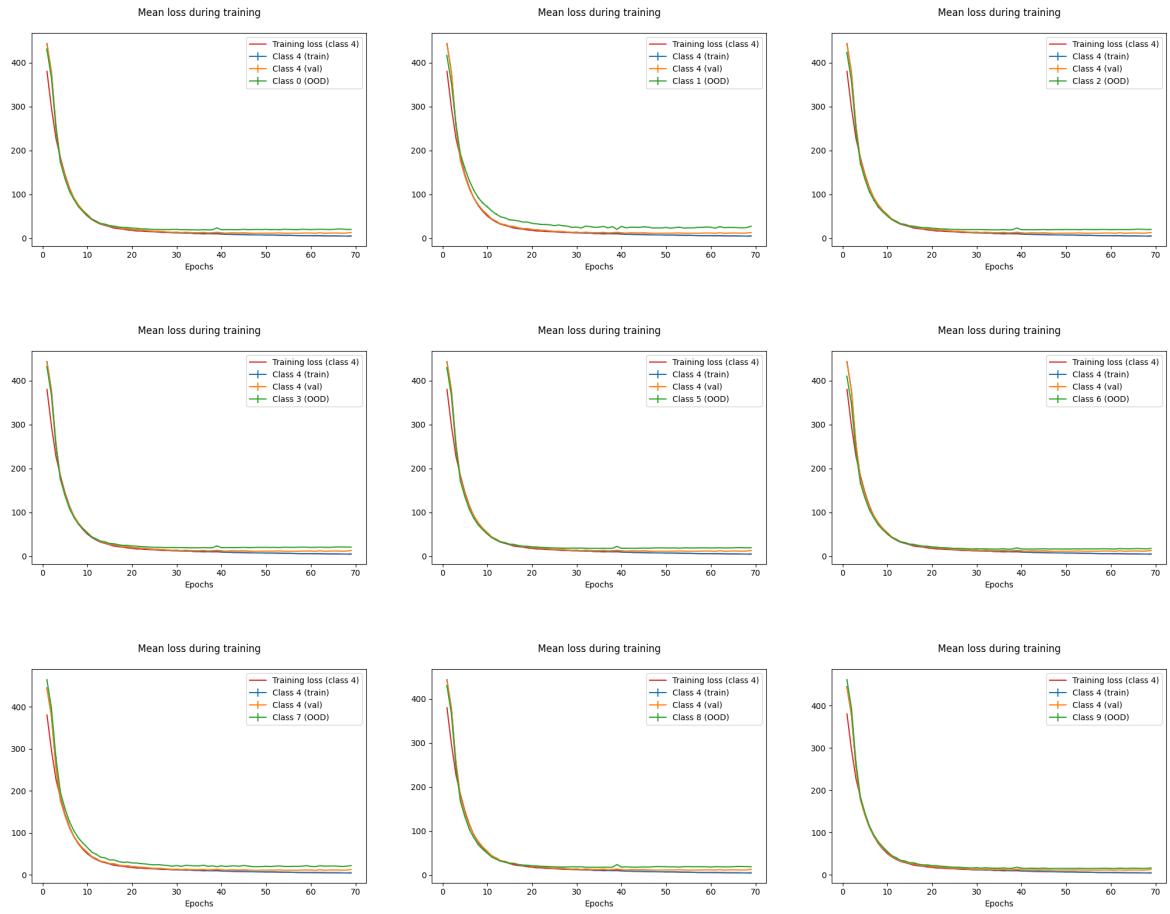


Figure 41 : Courbes de la MSE moyenne calculée sur différents ensemble d'images pendant l'entraînement du modèle Posterior de la figure 30.