

HeartBeat Classification

Projet Datascience

Jérôme VERNIER
Marouane ESSOUGDALI

1. Table des matières

1. TABLE DES MATIÈRES	2	
2. INTRODUCTION	4	
2.1. Classification	6	
2.2. Objectifs à Atteindre	6	
2.3. Equilibre des classes	6	
2.4. Méthodologies Envisagées	7	
2.5. Normalisation des Signaux – Vérification de l'Homogénéité des Datasets	8	
2.6. Équilibrage du Jeu de Données	9	
2.6.1. Augmentation du nombre de signaux anormaux	9	
2.6.2. Perturbations du signal	10	
3. VISUALISATION DES SIGNAUX	12	
3.1. Exemples de Signaux	12	
3.2. Similarités entre les Signaux	13	
3.3. Conclusion	14	
4. DIFFICULTÉS	15	
4.1. Durée du signal effectif	15	
4.2. Signaux étranges de la classe normale	17	
5. PRÉTRAITEMENT DES DONNÉES	19	
5.1. Pour un modèle basé sur des caractéristiques	19	
5.1.1. Analyse du Début du Signal	19	
5.1.2. Utilisation de Catch22	20	
5.1.3. Analyse des Pics	20	
5.2. Pour un modèle de deep learning	22	
6. PERSPECTIVES ET PROCHAINES ÉTAPES	23	
7. MODÉLISATION	24	
7.1. Classification par un classifier SVM	24	
7.1.1. Base don données d'entraînement :	24	
DST	avr24_continu_ds	2

7.1.1.	Résultats de modélisation :	24
7.1.2.	Conclusions	25
7.2.	Modèles de Deep Learning	26
7.2.1.	Réseaux de neurones simples	27
7.2.2.	Réseaux de neurones convolutifs	31
7.2.3.	Réseaux de neurones convolutifs et récurrents	39
7.2.4.	Amélioration du modèle CNN	41
7.2.5.	Transfer learning	42
7.3.	Classification binaire	47
8.	DISCUSSION GÉNÉRALE ET CONCLUSION	50

2. Introduction

Le projet présenté vise à évaluer les troubles cardiaques à partir d'enregistrements d'électrocardiogrammes (ECG). Les signaux cardiaques étudiés proviennent à la fois de patients en bonne santé et de patients atteints d'arythmie ou d'infarctus du myocarde. L'objectif principal de ce projet est de développer des architectures de réseaux de neurones profonds pour la classification des signaux cardiaques. En complément, l'étude des techniques de transfert d'apprentissage (transfer learning) sera également explorée dans le cadre de ce projet.

Deux ensembles de données (datasets) sont utilisés pour cette étude. Les signaux ont été préalablement transformés et segmentés, chaque segment correspondant à un battement de cœur. Un ECG est une mesure de l'activité électrique du cœur, qui se manifeste par plusieurs pics représentant l'activité des différentes cavités cardiaques. Le pic principal, appelé complexe QRS, correspond à l'activation des ventricules, l'onde P à l'activation des oreillettes, et l'onde T à la repolarisation des ventricules.

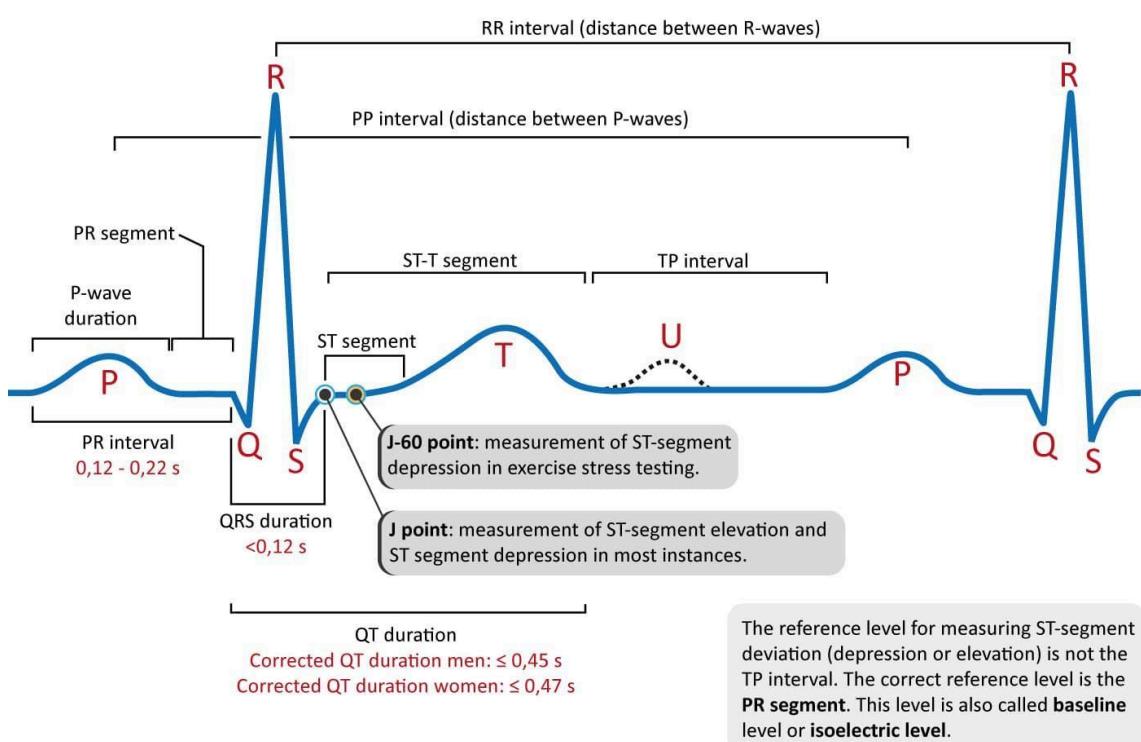


Figure 1: Ondes synthétiques d'un ECG normal (source : <https://ecgwaves.com/>)

L'analyse de ces ondes permet d'identifier des anomalies dans le fonctionnement du cœur, telles que l'arythmie, qui est un trouble du rythme cardiaque.

Un ECG anormal se distingue d'un ECG normal par des variations dans les amplitudes relatives des ondes, les intervalles de temps entre les ondes, ainsi que par des distorsions des formes d'onde.

L'utilisation du machine learning dans ce contexte a pour objectif de limiter les erreurs humaines et d'avoir un diagnostic temps réel de l'ECG.

Aucun de nous n'a d'expérience dans le domaine médical. Jérôme a une expérience en tant qu'expert métier géosciences sur un projet de classification d'images par deep learning dans lequel un modèle standard de classification d'image été fine-tuné pour classifier des signaux acoustiques 2D.

2. Description des Datasets

2.1. Classification

Chaque dataset contient des signaux de battements de cœur pré-segmentés. Les deux datasets sont homogènes en termes d'amplitude, de découpage et de durée. Cependant, la classification des signaux dans ces deux datasets diffère. Nous désignerons par "dataset 1" le jeu de données fourni par l'association de l'hôpital de Boston et du MIT (MIT-BIH). Le "dataset 2" est fourni par la Physikalisch-Technische Bundesanstalt (PTB, Allemagne).

Le dataset 1 propose une classification en cinq classes, de 0 à 4, correspondant à des annotations standard dans le domaine médical, dont les significations sont disponibles [\[ici\]](#) :

- Classe 0 : Annotation ECG N : ECG normal
- Classe 1 : Annotation ECG S : Battement prématué supraventriculaire
- Classe 2 : Annotation ECG V : Contraction ventriculaire prématurée
- Classe 3 : Annotation ECG F : Fusion de battements ventriculaires et normaux
- Classe 4 : Annotation ECG Q : Battement non classifiable

Le dataset 2, quant à lui, propose une classification binaire : ECG normal et ECG anormal.

Le dataset 1 est fourni avec des ensembles d'entraînement et de test, tandis que le dataset 2 ne contient qu'un ensemble non segmenté. Le dataset 1 comprend 87 554 signaux, tandis que le dataset 2 en contient 14 552.

2.2. Objectifs à Atteindre

L'objectif est de prédire, à partir d'un enregistrement ECG, si le signal est normal ou anormal. En cas d'anomalie, il s'agit également de pouvoir fournir une classification précise de cette anomalie.

2.3. Equilibre des classes

Les datasets sont fortement déséquilibrés avec une majorité forte de signaux de classe 0 (normaux) comme le montre la figure ci-dessous.

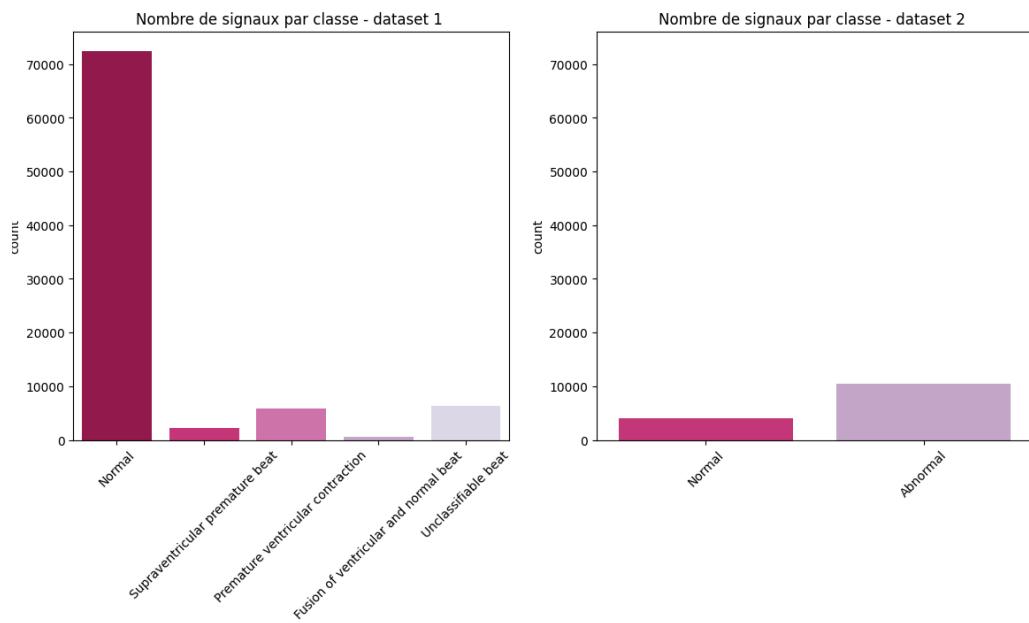


Figure 2: Nombre de signaux par classe et par dataset

2.4. Méthodologies Envisagées

Étant donné que les deux bases de données ne sont pas homogènes, plusieurs solutions sont envisagées :

1. N'utiliser qu'une seule des deux bases dans ce projet.
2. Fusionner les deux datasets en un seul avec trois classes : Normal, Anormal, Inclassifiable.
3. Entraîner un modèle sur le dataset 1, l'appliquer sur le dataset 2, augmenter le dataset 1 avec les classifications les plus probables du dataset 2, puis réentraîner le modèle.

Étant donné que le dataset 1 contient un nombre de données bien plus important et qu'il est fourni avec une base de test, les solutions 1 et 3 sont à privilégier. La solution 3 permettrait d'augmenter la taille du jeu d'entraînement, car le dataset 2 contient plus de signaux anormaux, ce qui est bénéfique pour rééquilibrer le dataset 1, qui est déséquilibré avec une dominance de signaux normaux.

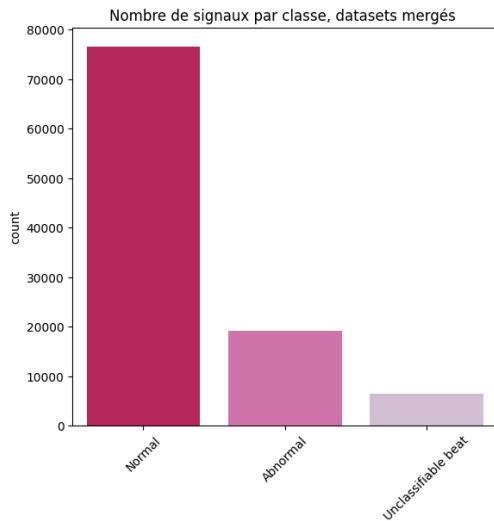


Figure 3: Nombre de signaux par classe des bases de données mergées

2.5.Normalisation des Signaux – Vérification de l'Homogénéité des Datasets

Chaque dataset a été normalisé entre 0 et 1, sans valeurs supérieures ou inférieures à cette plage. La distribution des valeurs est similaire pour les deux datasets. Les figures suivantes présentent les valeurs minimales et maximales des signaux sous forme d'histogrammes. Une échelle logarithmique est utilisée, car en échelle linéaire, on pourrait croire que tous les signaux ont une valeur minimale de 0 et une maximale de 1. Cela montre que la quasi-totalité des données est bien normalisée entre 0 et 1, à quelques exceptions près. Cette observation est confirmée par la distribution de l'intervalle (valeur maximale – valeur minimale). Le dataset 2 est encore mieux normalisé que le dataset 1. La distribution de l'énergie moyenne du signal (représentée par l'écart-type) montre que les deux datasets sont similaires.

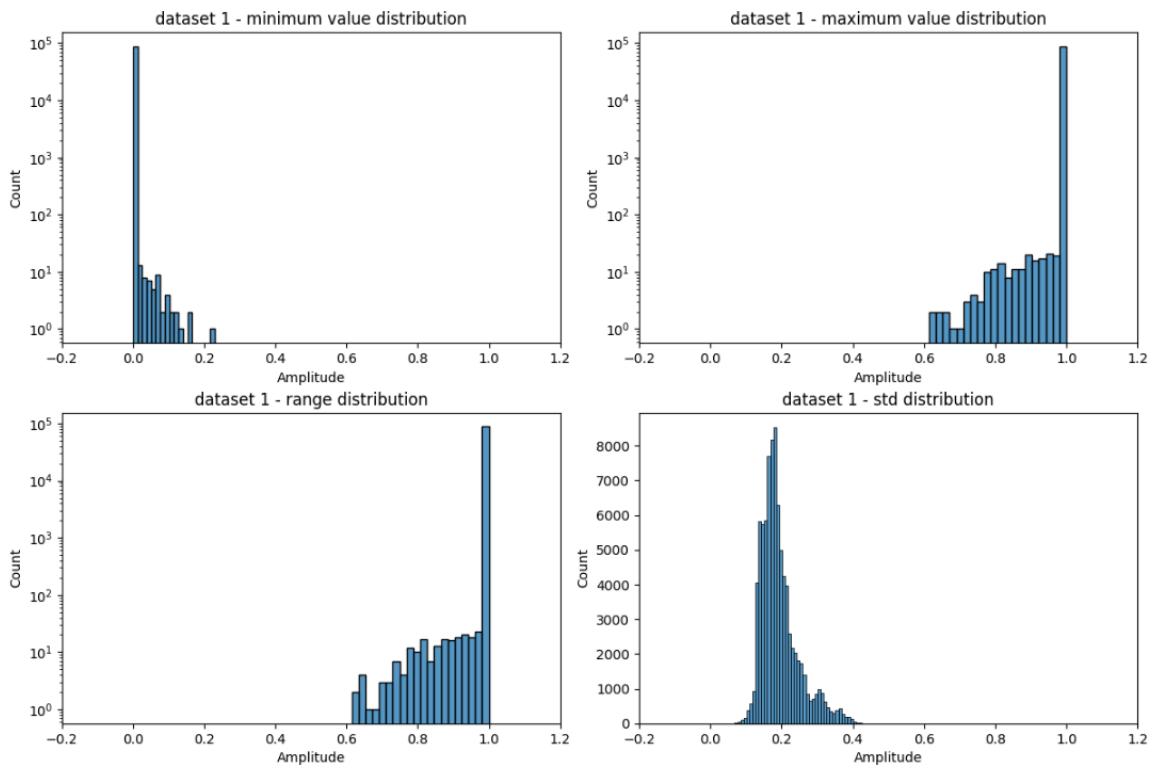


Figure 4: Dataset 1, distribution des valeurs minimales, maximales, du range et de la standard déviation

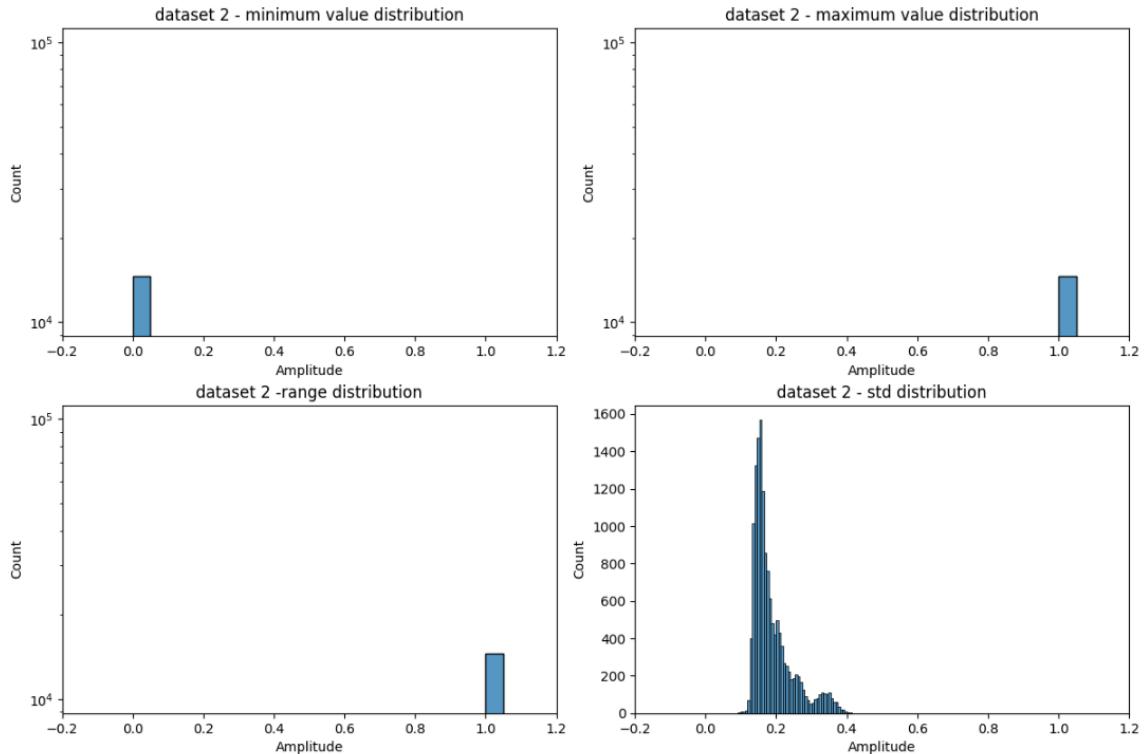


Figure 5: Dataset 2 distribution des valeurs minimales, maximales, du range et de la standard déviation

En conclusion, il n'y a pas de travail de normalisation du dataset à effectuer.

2.6. Équilibrage du Jeu de Données

Le dataset 1 est très déséquilibré, avec une majorité de signaux normaux. Pour y remédier, plusieurs options sont envisagées :

- Augmenter le nombre de signaux anormaux.
- Réduire le nombre de signaux normaux par undersampling.
- Pondérer les classes non normales lors de l'entraînement.

2.6.1. Augmentation du nombre de signaux anormaux

L'augmentation du nombre de signaux dans les classes minoritaires peut se faire par des fonctionnalités intégrées au module scikit-learn tel que le SMOTE ou par la perturbation des signaux.

Nous avons donc constitué différentes bases de données d'entraînements :

- Une base sous échantillonnée dont le nombre de signaux par classe correspond au nombre de la classe minoritaire (641)
- Une base sur échantillonnée par deux méthodes distinctes :
 - SMOTE appliqués sur X signaux de la classe majoritaire et au maximum X signaux de chaque classe
 - Une base constituée de Y autres signaux de la classe majoritaire et Y signaux perturbés par classe.

2.6.2. Perturbations du signal

Étant donné que le domaine médical n'est pas notre spécialité, il est crucial d'être très prudent lors de l'augmentation des signaux anormaux. Plusieurs techniques peuvent être envisagées pour augmenter les signaux tout en respectant les caractéristiques physiologiques des ECG :

1. Dilatation du Signal : La dilatation du signal modifie la fréquence des ondes. Par conséquent, cette technique doit être appliquée de manière limitée. Par exemple, un signal

d'une durée effective de 80 échantillons pourrait être utilisé pour créer un signal d'une durée effective de 81 ou 82 échantillons.

2. Amplification du Signal : Le signal peut être amplifié tout en restant dans la plage de 0 à 1. Des facteurs d'amplification faibles, tels que $x^{1.1}$, $x^{1.2}$ et $x^{1.3}$, peuvent être utilisés pour éviter de dénaturer les caractéristiques du signal.

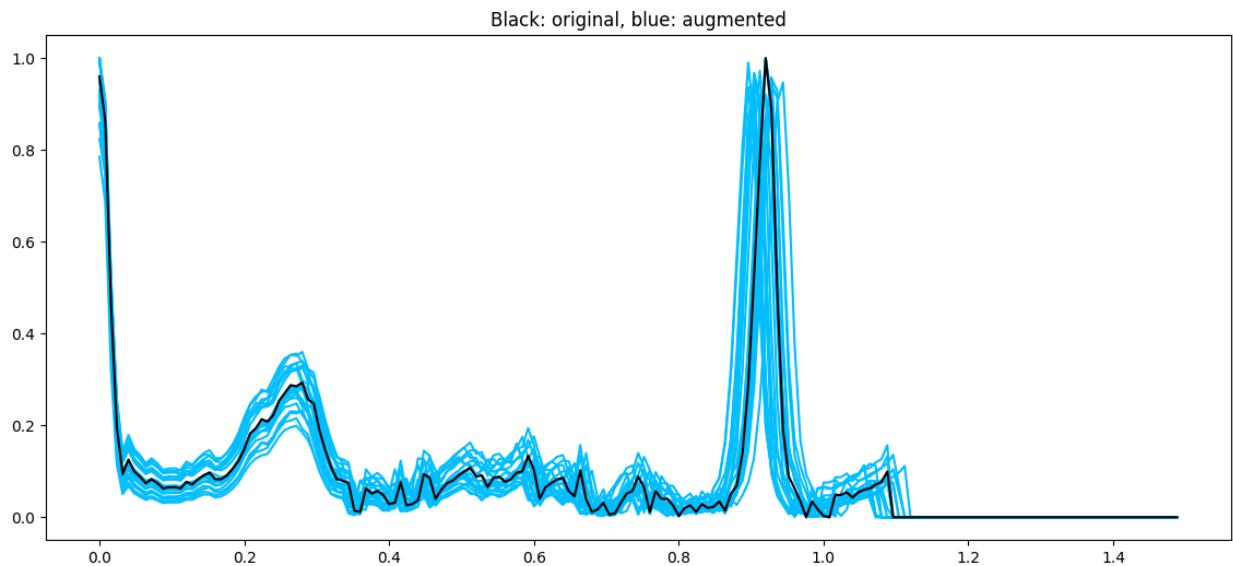


Figure 6: Augmentation Temporelle

3. Augmentation Temporelle : Le signal peut également être amplifié avec un facteur linéaire dépendant du temps. Cependant, ces coefficients doivent rester faibles, par exemple, un facteur de 1.1 au maximum. La modification appliquée est de type $f_p(t) = (a * t + b) * f(t)$

Le nombre minimal de signaux se trouve dans la classe F (Fusion of ventricular and normal beat) avec seulement 641 signaux. L'objectif serait d'amener l'ensemble des classes à un minimum de 6 000 signaux, même si cela implique un léger déséquilibre dans la classe F.

3.3. Visualisation des Signaux

3.1. Exemples de Signaux

Nous avons d'abord examiné le dataset 1, qui comprend 5 classes, en affichant quelques signaux de chaque classe pour identifier d'éventuelles similarités.

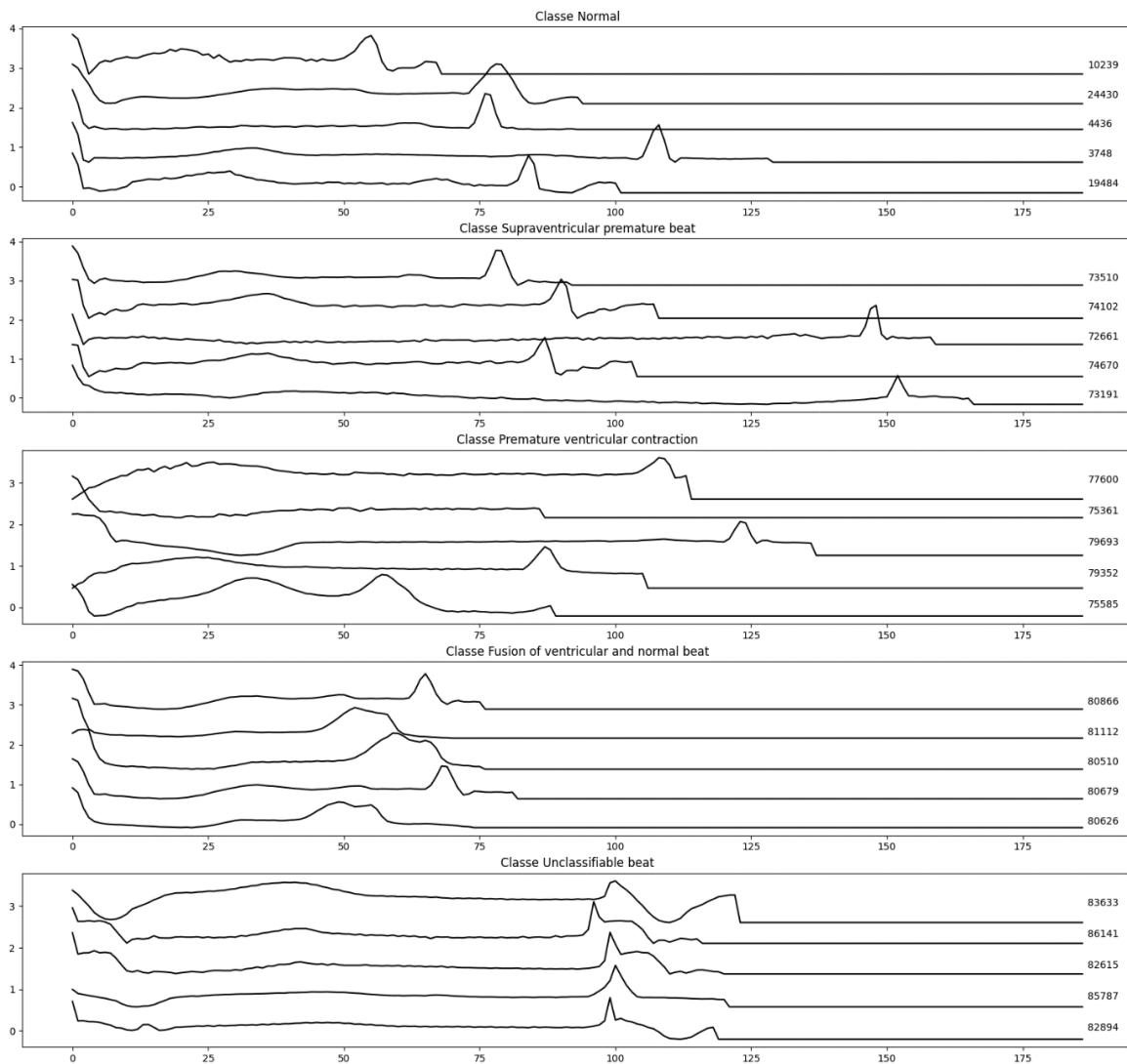


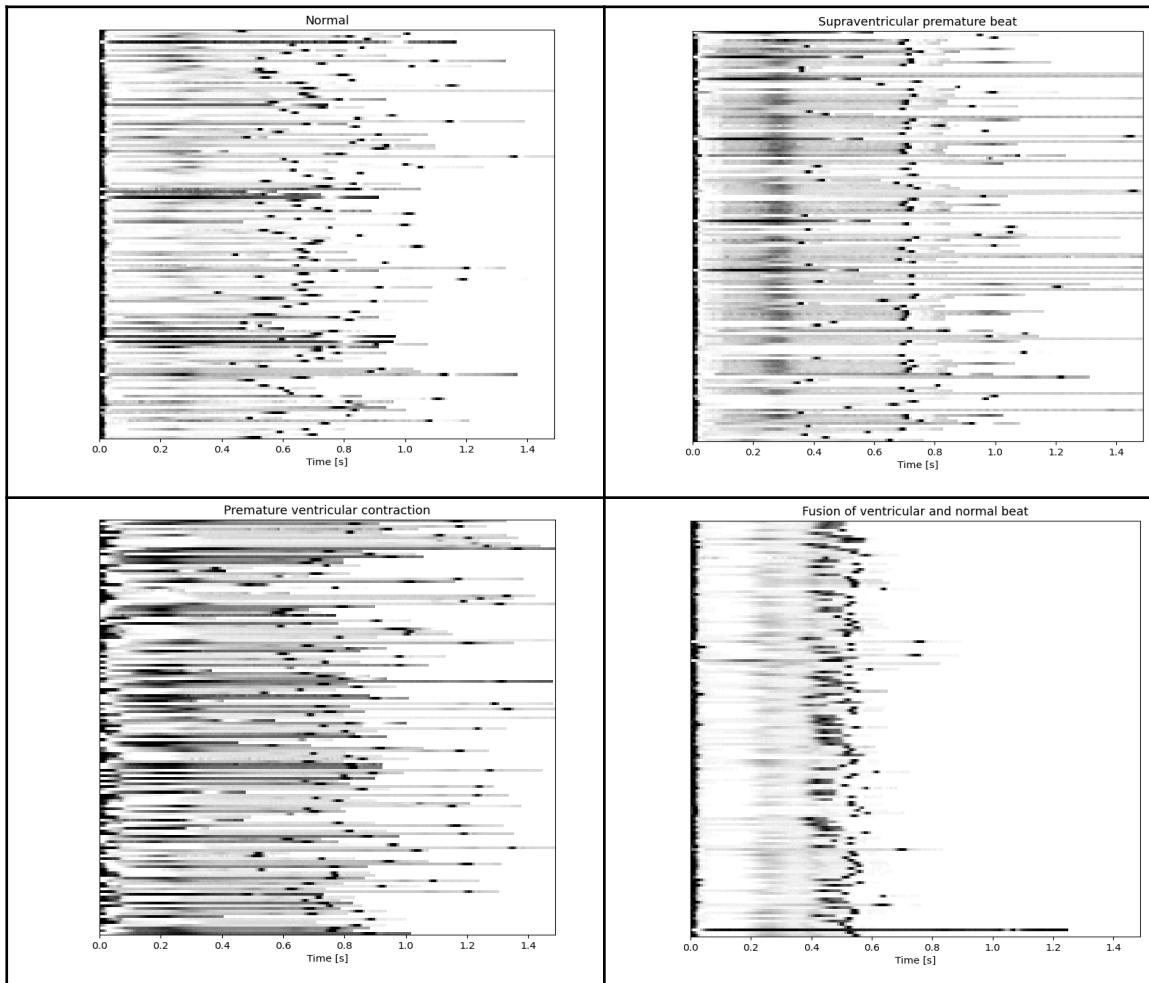
Figure 7: 5 exemples de signaux par classe

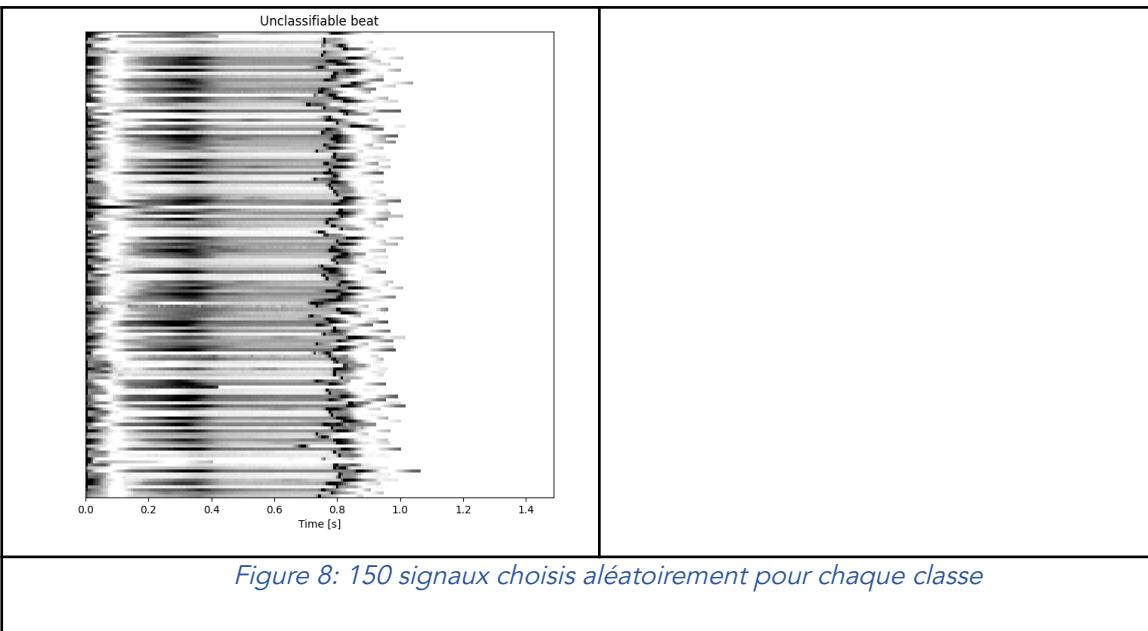
Il semble assez difficile d'identifier un ECG normal par rapport à un ECG normal théorique. Le début du signal semble correspondre à un pic R et se poursuivre jusqu'au prochain complexe QRS. Les ondes P et T, qui suivent ou précèdent le complexe principal, ne sont pas toujours

évidentes sur ces exemples. Toutefois, on observe une distorsion de la première onde R pour les classes 2 et 4 (Premature Ventricular Contraction et Unclassifiable).

3.2. Similarités entre les Signaux

Une autre méthode pour évaluer la similarité des classes consiste à afficher un plus grand nombre de signaux sous forme d'image. Pour cela, 150 signaux ont été affichés pour chaque classe, concaténés les uns au-dessous des autres et présentés sous forme d'images. Un pixel noir isolé correspond généralement au complexe QRS.





Les observations suivantes peuvent être faites :

- La classe F (Fusion of Ventricular and Normal Beat) est majoritairement constituée de signaux courts, inférieurs à 0,6s.
- Les classes S (Supraventricular Premature Beat) et Q (Unclassifiable) affichent une onde plus énergétique autour de 0,3s.
- Les classes Q (Unclassifiable) et V (Premature Ventricular Contraction) présentent une première descente d'amplitude à une fréquence plus basse.
- Les classes Q (Unclassifiable) et F (Fusion of Ventricular and Normal Beat) sont de durées relativement homogènes.

3.3. Conclusion

Bien que ce projet se concentre sur l'implémentation d'un réseau de neurones, il pourrait être intéressant de comparer les performances avec un modèle de machine learning basé sur des attributs. D'après cette analyse, la forme de la première onde R, la durée du signal, et l'amplitude de l'onde T semblent être des critères corrélés à la classification.

4. Difficultés

4.1. Durée du signal effectif

Une des principales difficultés rencontrées avec ce jeu de données est la classification des signaux courts. Bien que les signaux soient complétés par des zéros (padding), certains d'entre eux restent très courts. Cela signifie que la quantité d'information disponible pour la classification est faible, ce qui complique la tâche du modèle.

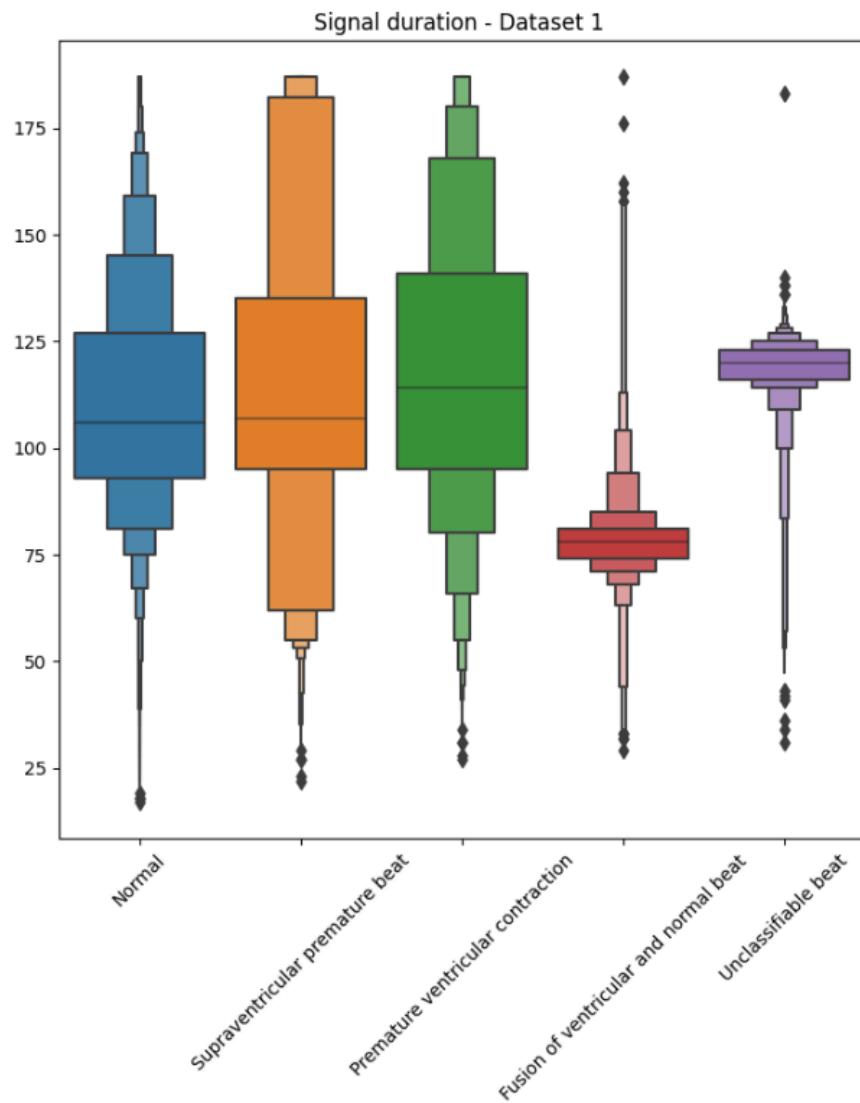


Figure 9: Durée du signal Dataset 1

Il est possible de calculer facilement la durée effective d'un signal en identifiant le premier échantillon non nul en partant de la fin du signal. La durée du signal peut être corrélée à la classe. Par exemple, les signaux de la classe F (Fusion of ventricular and normal beat) sont

majoritairement courts, avec une distribution très concentrée autour de 80 échantillons. On observe également des signaux extrêmement courts, de moins de 40 échantillons, dans chaque classe. Il semble difficile de classifier avec si peu d'information, ce qui suggère que la classification pourrait ne nécessiter que la phase descendante de la première onde R. Notamment, dans la classe normale, près de 300 signaux sont très courts.

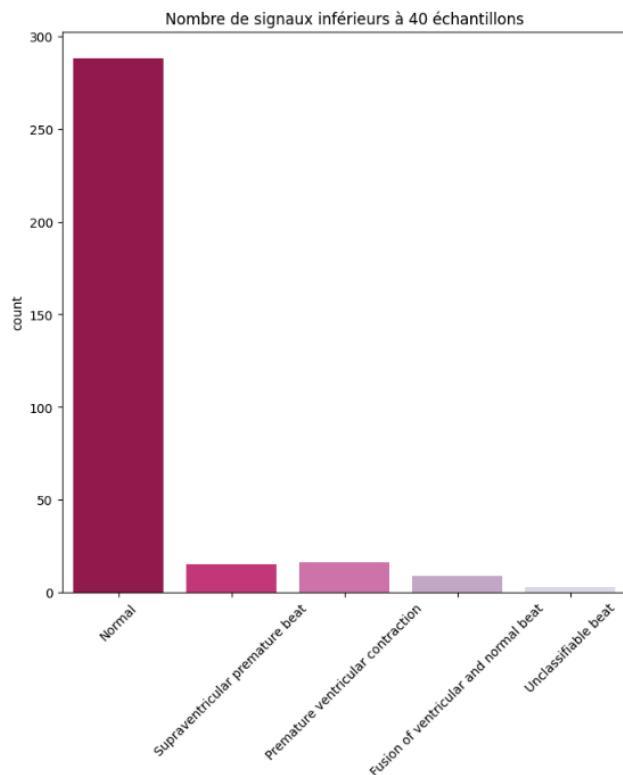


Figure 10: Nombre de signaux inférieurs à 40 échantillons par classe

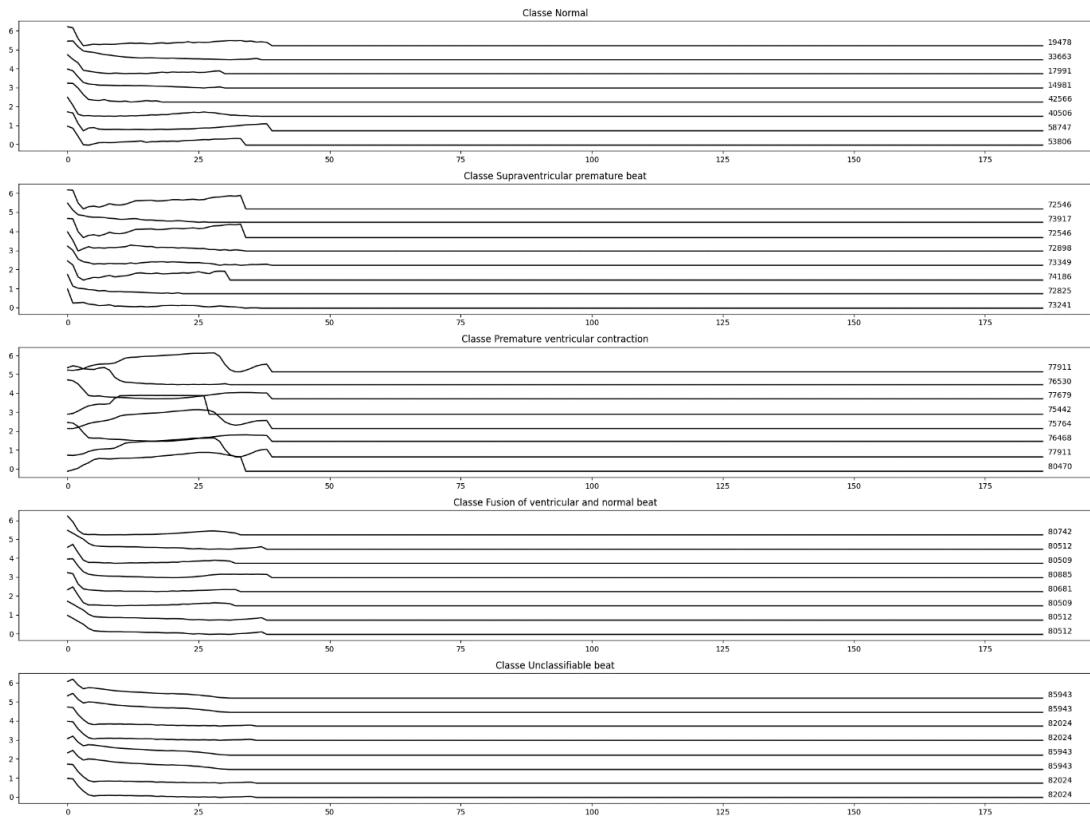


Figure 11: 5 signaux inférieurs à 40 échantillons par classe

Bien qu'une différence puisse être distinguée pour la classe V (premature ventricular contraction), un signal trop court ne semble pas contenir suffisamment d'information pour être classifié correctement. Deux solutions peuvent être proposées :

1. Retirer de la base d'entraînement les signaux courts.
2. Reclassifier les signaux courts dans une classe Q.

Étant donné que la classe V semble être caractérisée par ses premiers échantillons, il est préférable de retirer ces signaux courts pour ne pas perturber le modèle. Dans le dataset 2, qui est plus propre, le nombre minimal d'échantillons est de 51. Je propose donc de retirer du dataset 1 tous les signaux de moins de 50 échantillons.

4.2. Signaux étranges de la classe normale

Certains signaux de la classe normale ne semblent pas correspondre au signal synthétique typique d'un ECG. On peut notamment remarquer une inversion de signe de l'onde R. en temps que non spécialiste, il est étonnant que ces signaux soient dans la classe normale. Nous espérons que le modèle saura réussir à correctement les classifier.

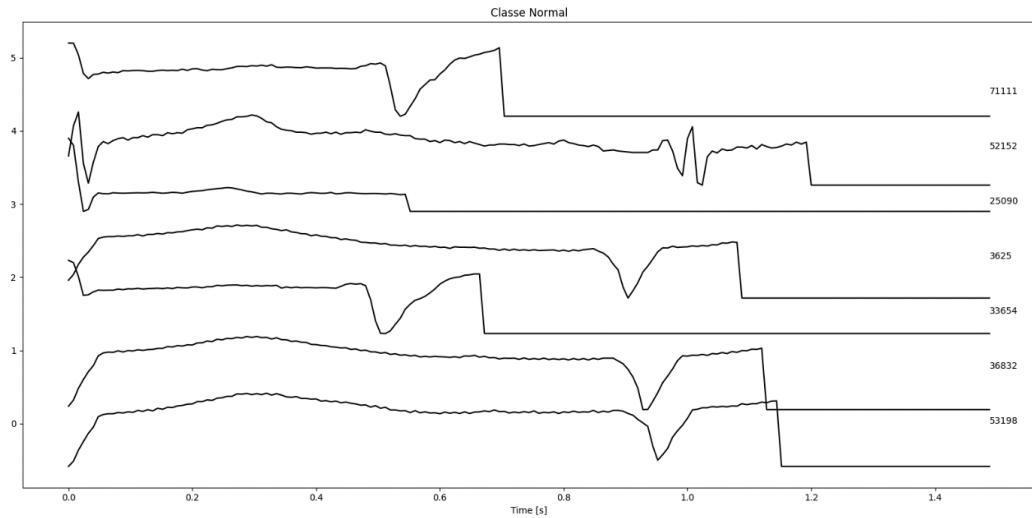


Figure 12: signaux classe normale

5. Prétraitement des données

5.1. Pour un modèle basé sur des caractéristiques

5.1.1. Analyse du Début du Signal

Nous avons observé que le début du signal est caractéristique des différentes classes. Par conséquent, nous calculons certains attributs qui devraient faciliter la classification :

- La pente moyenne des 5 premiers échantillons.
- L'étendue (range) des 5 premiers échantillons.

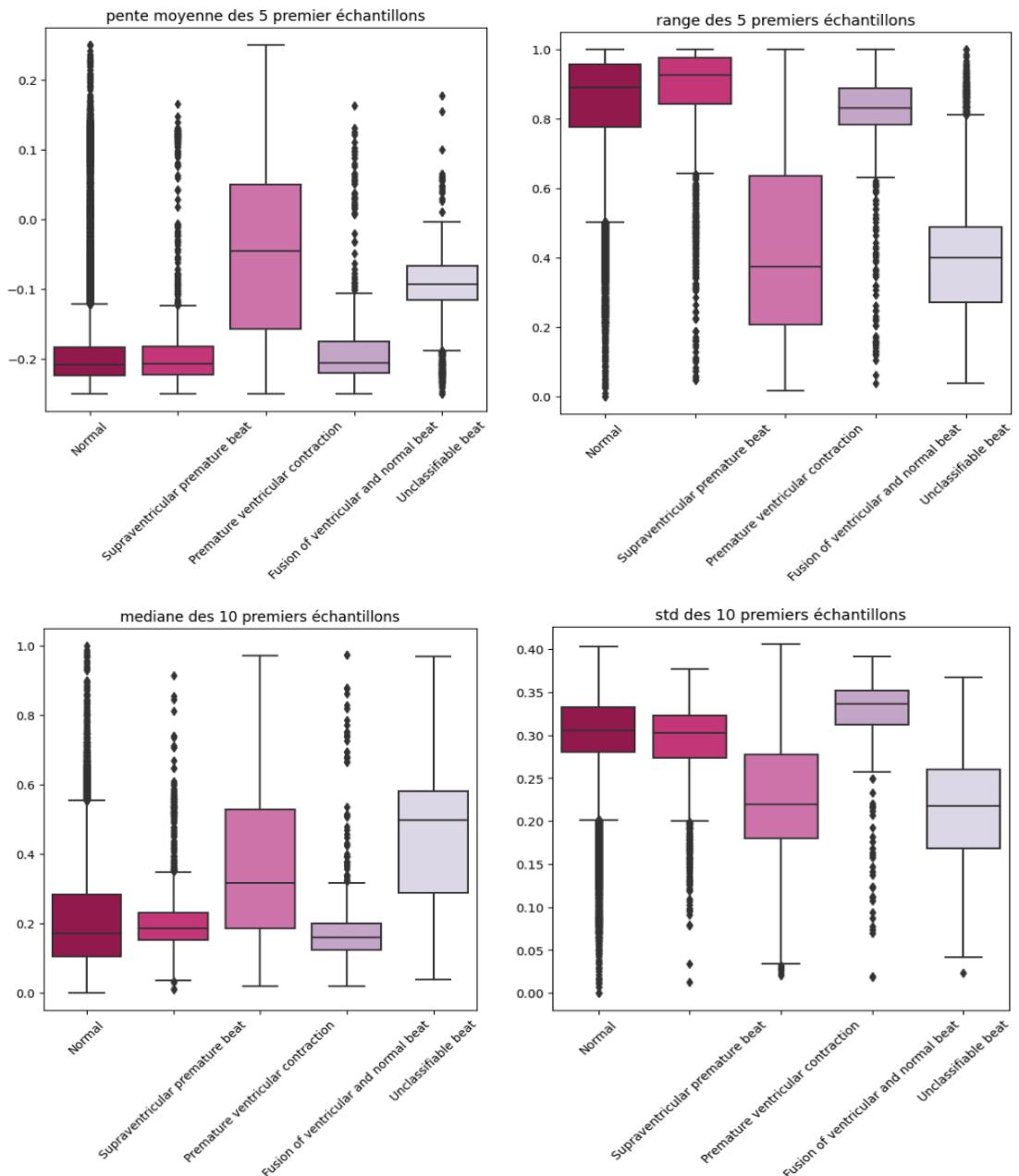


Figure 13: attributs sur le début du signal

5.1.2. Utilisation de Catch22

Nous prévoyons d'utiliser l'ensemble de caractéristiques Catch22, spécialement conçu pour l'analyse de séries temporelles. Catch22 calcul 22 attributs sur les signaux. Ces attributs ont été sélectionnés en fonction de leurs performances de classification sur un ensemble de 93 problèmes de classification de séries chronologiques du monde réel. A ces 22 attributs, la moyenne et la standard-déviation peuvent être ajoutés.

5.1.3. Analyse des Pics

La classification dépend probablement des caractéristiques des ondes de l'ECG, et donc des pics du signal. Nous proposons de calculer une série d'attributs sur les 3 pics principaux du signal. Dans le cas d'un signal normal, nous espérons identifier les ondes correctes qui devraient avoir des signatures communes.

L'identification des pics est plus facile sur un signal présentant des variations lentes. Nous proposons de filtrer légèrement le signal pour atténuer les variations de haute fréquence. Différents tests ont été effectués, et nous avons une préférence pour un filtre gaussien ou un filtre médian. Cela peut être un paramètre de notre modélisation. L'objectif est de conserver l'amplitude des pics tout en atténuant les variations de haute fréquence. La figure suivante montre 4 filtrages différents sur le même signal d'entrée toujours affiché en noir. Les deux premiers correspondent à l'application d'un filtre gaussien, la troisième un filtre médian et la dernière un filtre moyen.

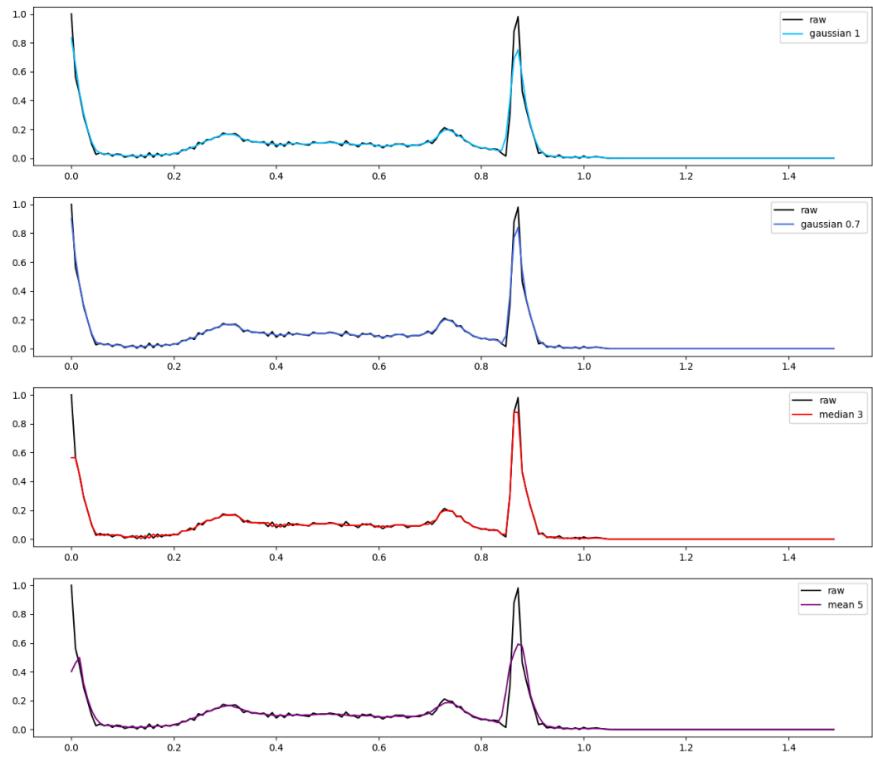


Figure 14: 4 lissages du même signal

Sur un signal légèrement filtré, il est plus facile d'identifier les pics. La fonction `findpeaks` du module `scipy.signal` retourne aussi les largeur et hauteurs de pics. Ci-dessous un exemple de détection de 3 pics sur la classe normale et la classe Fusion de battements ventriculaires et normaux .

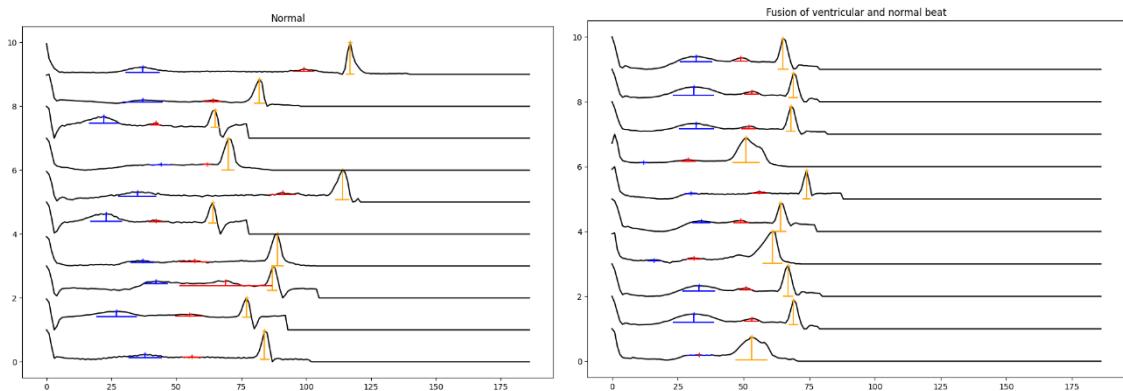


Figure 15: pics identifiés sur 10 signaux de deux classes

La concaténation des attributs des pics et de `catch22` donnent par exemple cette matrice de corrélation.

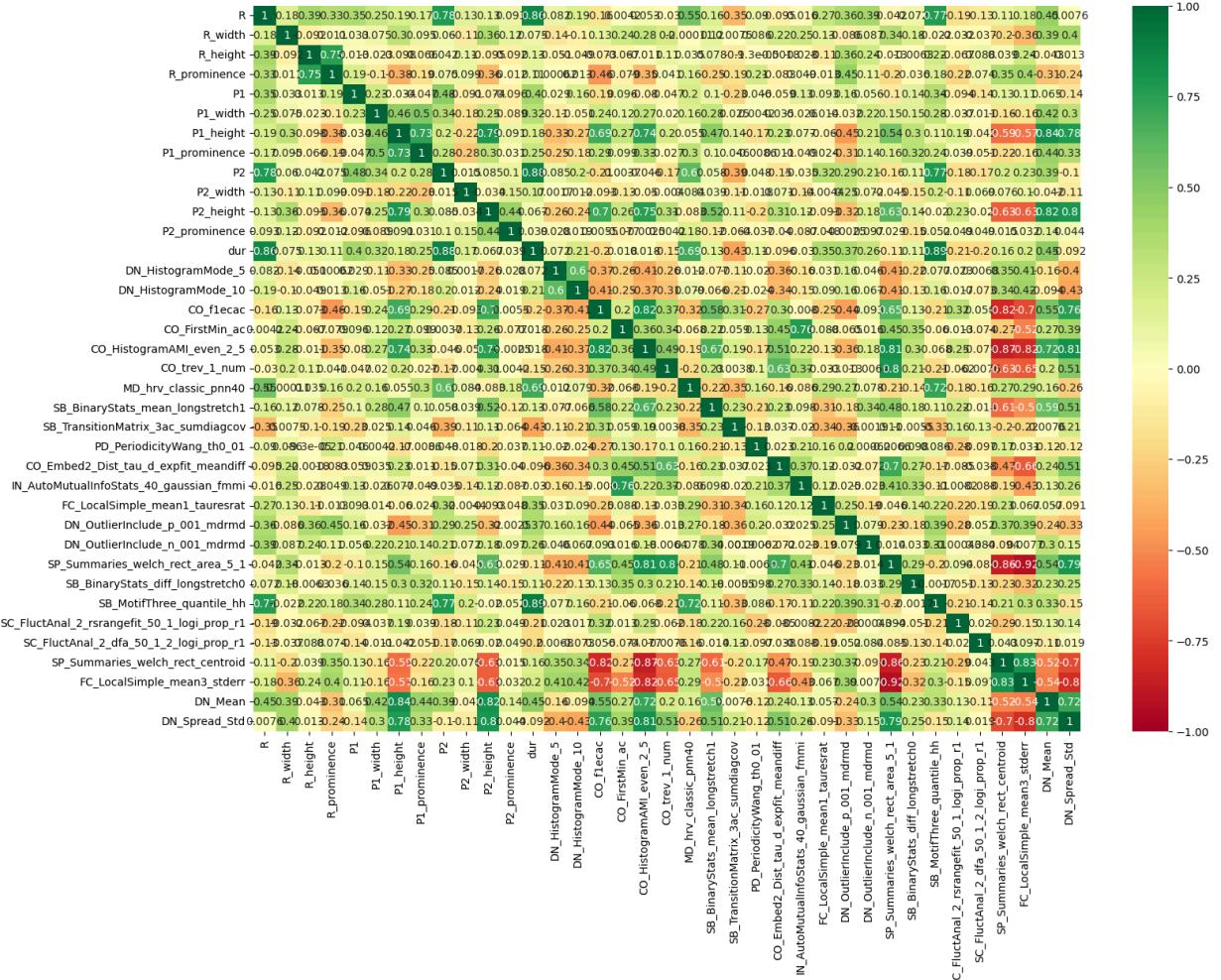


Figure 16 : matrice de corrélation sur les attributs calculés

Nous ajoutons à ce attributs les paramètres calculés sur le début du signal (pente, range, std, médiane) ce qui donne en tout 41 attributs pour un modèle de machine learning.

5.2. Pour un modèle de deep learning

Dans le cas d'un modèle de deep learning, nous ne pensons pas appliquer de prétraitement. Le modèle sera basé sur différentes couches de convolutions 1D qui dans notre cas ne nécessite pas de transformation préalable.

6. Perspectives et Prochaines Étapes

- Implémentation et comparaison de différentes architectures de réseaux de neurones :
 - Nous allons implémenter plusieurs architectures de réseaux de neurones, notamment les CNN 1D, LSTM, et Transformers. Chaque architecture sera testée pour évaluer sa capacité à traiter les signaux ECG et à effectuer une classification précise.
- Évaluation des performances des modèles sur le Dataset 1 :
 - Une évaluation approfondie des performances des modèles sera réalisée en utilisant le Dataset 1. Les métriques de performance, telles que la précision, le rappel et le F1-score, seront analysées pour identifier les modèles les plus efficaces.
- Intégration du Dataset 2 pour améliorer la généralisation :
 - Le Dataset 2 sera intégré dans le processus d'entraînement pour renforcer la capacité des modèles à généraliser sur des données non vues. Cela permettra de vérifier si les modèles peuvent maintenir leur performance sur des ensembles de données plus diversifiés.
- Développement de techniques d'interprétabilité :
 - Des techniques d'interprétabilité seront développées pour expliquer les décisions des modèles. Cela inclut l'analyse des caractéristiques importantes qui influencent les prédictions, afin de mieux comprendre les résultats obtenus.

Chaque modèle sera entraîné à la fois sur un dataset sous échantilloné et un dataset sur-échantilloné. Un tableau de performance pourra alors être utilisé incluant les métriques, les dataset d'entraînement et les différents modèles.

7. Modélisation

Nous entamons ici la phase de modélisation du projet, une étape cruciale et complexe qui implique de nombreux choix nécessitant une analyse approfondie. Cette phase représente le cœur du travail d'un Data Scientist. Nous avons choisi de débuter avec un modèles de classification simple, afin de définir une baseline permettant d'évaluer et de comparer les performances de modèles plus complexes à venir. L'enchaînement des paragraphes suit globalement la chronologie des étapes de modélisation.

7.1. Classification par un classifier SVM

7.1.1. Base don données d'entraînement :

Un modèle SVM a été entraîné sur différentes bases de données :

- La base de données d'entraînement sous échantillonnée. Le nombre de signaux par classe est ramené au plus faible, c'est-à-dire 641.
- La base de données sur-échantillonnée et augmentée construite à partir de la base train contenant 13000 signaux par classe, puis 50 000 signaux par classe.

La modélisation utilise les attributs définis dans le paragraphe 5. Les attributs sont normalisés par retrait de la moyenne et division par la déviation standard.

1.1.1. Résultats de modélisation :

Une première modélisation est réalisée sur la base sous échantillonnée pour tester le classifier, puis un gridsearch a été effectué. Les meilleurs paramètres issus de la base de données sous-échantillonnées sont utilisés pour la modélisation sur les bases avec un nombre élevés de paramètres. Les paramètres du gridseacrh testés sont :

- C : de 0.1 à 10
- Gamma : de 0.001 à 0.6
- Kernel : 'rbf', 'linear', 'poly'

Ce gridsearch a été effectué en 2 étapes afin d'affiner les paramètres C et gamma une fois le kernel déterminé et les meilleurs paramètres sont retenus sur la base du f1 score.

La table suivante présente les résultats sur la base de test. Cette base est aussi déséquilibrée en faveur de la classe normale. Pour évaluer les résultats, nous regardons alors les scores macro qui moyenne les scores sans tenir compte du nombre de signaux par classe. Si l'on regardait les scores généraux, les résultats paraîtraient souvent très bons alors que de nombreuses erreurs existent dans les classes avec peu de signaux.

Tableau 1 : résultats d'évaluation de modèles Machine learning

modèle	Nombre de signaux par classe	gamma	kernel	C	precision macro avg	recall macro avg	f1 score macro avg
SVM initial	641	0,01	poly	1	0,59	0,74	0,52
SVMgridsearh	641	0,1	rbf	10	0,6	0,89	0,67
SVM	13000	0,1	rbf	10	0,78	0,93	0,84
SVM	50000	0,1	rbf	10	0,81	0,84	0,82

Nous voyons que l'augmentation de la base d'entraînement permet d'augmenter la précision. Cependant nous voyons une baisse du recall avec a plus grosse base de données.

prediction	0.0	1.0	2.0	3.0	4.0	prediction	0.0	1.0	2.0	3.0	4.0
target						target					
0.0	17378	389	226	92	33	0.0	17762	202	69	61	24
1.0	63	472	18	2	1	1.0	310	231	10	3	2
2.0	19	5	1395	25	4	2.0	36	0	1376	32	4
3.0	8	1	10	143	0	3.0	7	1	9	145	0
4.0	7	5	27	0	1569	4.0	18	0	10	1	1579

Figure 17: meilleurs résultats sur la base à 13000 signaux par classe (gauche) et 50000 signaux par classe (droite)

Dans le premier cas, 97 personnes présentant une anomalie n'ont pas été détectées, dans l'autre, 371, et 770 personnes présentant un électrocardigramme normal ont été diagnostiquées avec une anomalie contre 356 pour le deuxième modèle. Si nous nous arrêtons là, nous choisirions le premier modèle même si la précision sur le deuxième est meilleure pour éviter d'avoir des anomalies non détectées.

1.1.2. Conclusions

Ce modèle présente des résultats satisfaisant pour les la classification des. Cependant, on note que pour les classe 1 et 3, si le modèle prévoit ce classement, on a juste légèrement plus d'1 chance sur 2 que cela soit véritablement le cas.

Nous avons également évalué le modèle sur une autre base de données en utilisant une classification binaire (normal/anormal). Lors de l'application de cette classification à la base contenant des signaux « anormaux », nous nous attendions à ce que peu de signaux soient classés comme normaux. Cependant, les résultats se sont révélés très insatisfaisants, avec la

majorité des signaux étant incorrectement classés comme normaux. Faute d'informations détaillées sur cette base, il est possible qu'elle contienne des types d'électrocardiogrammes anormaux absents de la base d'entraînement. Nous avons quand même testé d'augmenter la base d'entraînement avec le résultat de la classification appliquée à cette base anormale puisque nous pouvons constater que lorsque le modèle prédit un signal véritablement anormal, sa classification est plutôt bonne. Cependant, ceci a détérioré les résultats, diminuant la précision et le F1-score, principalement sur la classe 1 avec beaucoup plus de signaux normaux classé en 1.

						precision	recall	f1-score	support		
prediction	target	0.0	1.0	2.0	3.0	4.0	0.0	0.98	0.95	0.97	18118
		0.0	1.0	2.0	3.0	4.0	1.0	0.29	0.53	0.37	556
0.0	17231	729	68	70	20	2.0	0.93	0.95	0.94	1448	
1.0	244	295	11	3	3	3.0	0.58	0.87	0.70	162	
2.0	44	2	1370	28	4	accuracy			0.94	21892	
3.0	9	1	11	141	0	macro avg	0.75	0.86	0.79	21892	
4.0	20	1	10	1	1576	weighted avg	0.96	0.94	0.95	21892	

Figure 18 : résultats sur un SVM entraîné sur la base à 50000 signaux augmentés de la classification du modèle précédent sur les signaux anormaux du dataset 2

1.2.Modèles de Deep Learning

Nous nous sommes concentrés sur la modélisation par deep learning. Plusieurs étapes ont été nécessaires dans le process. Cependant les résultats ont été plutôt bons dès la première expérimentation. L'entraînement des modèles a été réalisé avec PyTorch. Nous avons dans un premier temps testé des modèles composés uniquement de réseaux de neurones puis incluant des couches de convolutions 1D. Ces premiers tests ont été réalisés sans normalisation ni dropout. Au fur et à mesure du process et en regardant les modèles existants en traitement d'image, nous avons ajouté des normalisations et du dropout ce qui a légèrement amélioré les résultats. De nombreux tests ont été réalisés en faisant varier les fonctions d'activation et le nombre de neurones. Nous avons du prendre de l'avance sur le programme et avons donc par exemple fait l'erreur d'essayer un gridsearch sur des modèles CNN ce qui en pratique ne se fait pas car le nombre de paramètres entrant en jeu est trop élevé, et nous nous en sommes rendu compte. Pour cette classification nous avons tester différentes approches :

- ANN : réseaux de neurones
- CNN : réseaux de neurones convolutifs
- RNN : réseaux de neurones incluant des cellules récurrentes permettant d'avoir un effet mémoire.

De la même façon, ces modèles ont été appliqués sur les bases sous échantillonnées et sur-échantillonnées. Nous nous sommes vite rendu compte qu'il était nécessaire d'avoir une base conséquente pour entraîner ces modèles. Il a fallu un temps non négligeable pour prendre en main Pytorch et notamment sur les modèles récurrents. Nous n'avons pas pu tester par exemple les couches LSTM. La théorie sur ses couches récurrentes est assez complexe et il a fallu énormément de temps pour comprendre les RNN simples.

La première étape de modélisation a consisté à se familiariser avec Pytorch et à entraîner des modèles ANN et CNN sur la base train et analyser les résultats sur la base test. Dans la deuxième étape, nous avons tenté d'améliorer ces premiers modèles notamment en introduisant des couches de normalisations et de dropout pour essayer de réduire l'overfitting. Nous avons aussi entraîné et analysé les résultats en utilisant de la cross-validation à partir de la base de données d'entraînement. A la toute fin, le modèle est testé sur la base de test qui n'a donc jamais été utilisée à des fins d'entraînement.

Au cours de la troisième partie de modélisation, nous avons entrepris l'implémentation de modèle récurrent ainsi que le test de transfer learning à partir d'un modèle de classification d'images.

1.2.1. Réseaux de neurones simples

Les modèles testés ici sont des réseaux de neurones, avec différents nombres de neurones par couches et différentes couches. De nombreux essais ont été réalisés avec 3 grandes familles de réseaux de neurones :

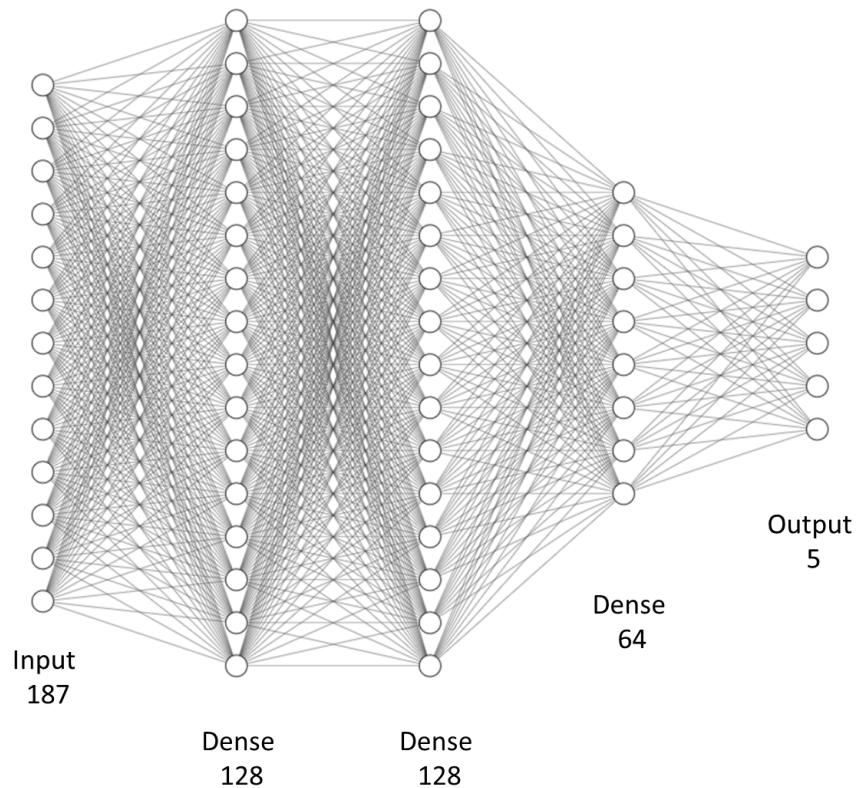
- Un réseau de 3 couches où le nombre de neurones diminue avec les couches
- Un réseau de 3 couches de nombres de neurones égaux
- Un réseau avec 5 couches de nombres de neurones égaux

Des tests de fonction d'activations ont également été effectués. Nous nous sommes contraints à ne pas avoir un modèle avec des centaines de milliers de paramètres. N'ayant qu'à la base environ 20 000 signaux et 187 échantillons, il ne faudrait pas avoir plus de paramètres que de données.

Tableau 2 : résultats d'évaluation de modèles de réseaux de neurones

modèle	Annotation	Nombre de signaux par classe	Nombre de paramètres	precision macro avg	recall macro avg	f1 score macro avg
ANN 1		13000	18,437	0,67	0,91	0,74
ANN 2		13000	20,677	0,74	0,91	0,8
ANN 3		13000	33,157	0,62	0,88	0,68
ANN 2		50000	20,677	0,85	0,91	0,87
ANN 4	Sigmoid	50000	49,157	0,84	0,92	0,87
ANN 4	ReLU	50000	49,157	0,86	0,91	0,88
ANN 4	LeakyRelu	50000	49,157	0,86	0,91	0,88
ANN 5		50000	11,949	0,87	0,91	0,89
ANN 4	Smote seulement	50000	49,157	0,9	0,9	0,9
double entraînement avec réduction du learning rate		50000	49,157	0,9	0,91	0,9

Le modèle retenu ayant les meilleurs scores est composé des couches suivantes :



Les activations sont de ReLU sauf la dernière couche où la classification est réalisée par une couche Softmax. L'optimisation du modèle est réalisée avec l'optimiseur Adam sur 50 epochs, tel que le montre la figure suivante. La base de cross-validation correspond à un 5% de la base à 50 000 signaux par classe.

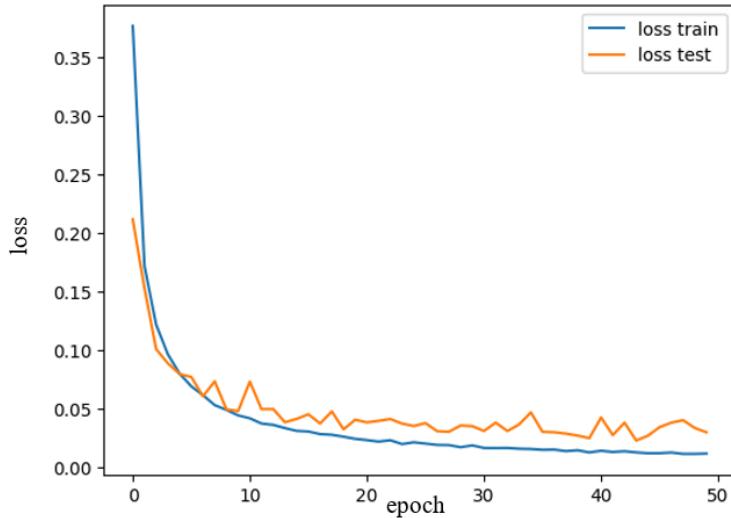


Figure 19: Courbe de loss de l'entraînement du meilleur modèle.

Une deuxième itération a été effectuée en changeant le learning rate. Ceci n'améliore pas le modèle qui over fit.

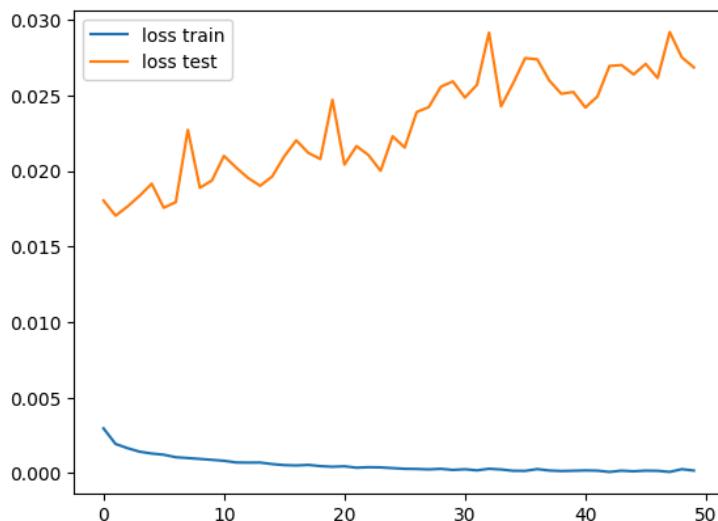


Figure 20: Tentative d'amélioration par diminution du learning rate.

L'interprétabilité des modèles de deep learning est plutôt compliquée. Nous avons utilisé les SHAP values pour voir quels échantillons interviennent dans la classification d'un électrocardiogramme. Les SHAP values sont entraînées sur la base du fichier mit_train.csv puis exploitées sur des signaux de la base de test. Sur les quelques signaux analysés, ce sont quand même les premiers échantillons qui ressortent puis les pics du signal.

Ci-dessous un exemple des meilleurs échantillons pour la classification sur 4 signaux.

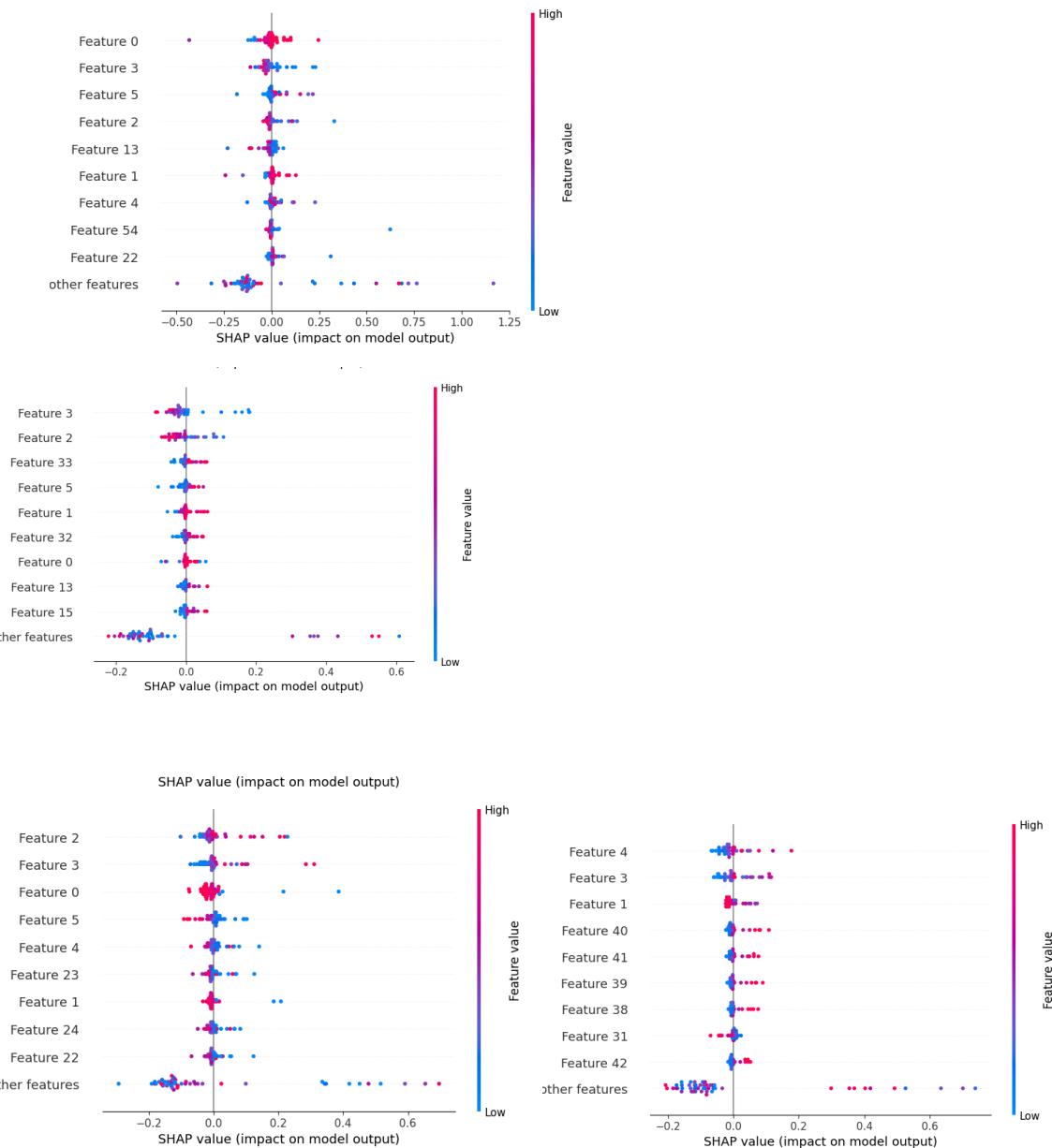


Figure 21: importance des SHAP values

Si l'on regarde un signal de plus près, voici ce que nous donne l'analyse des SHAP values :

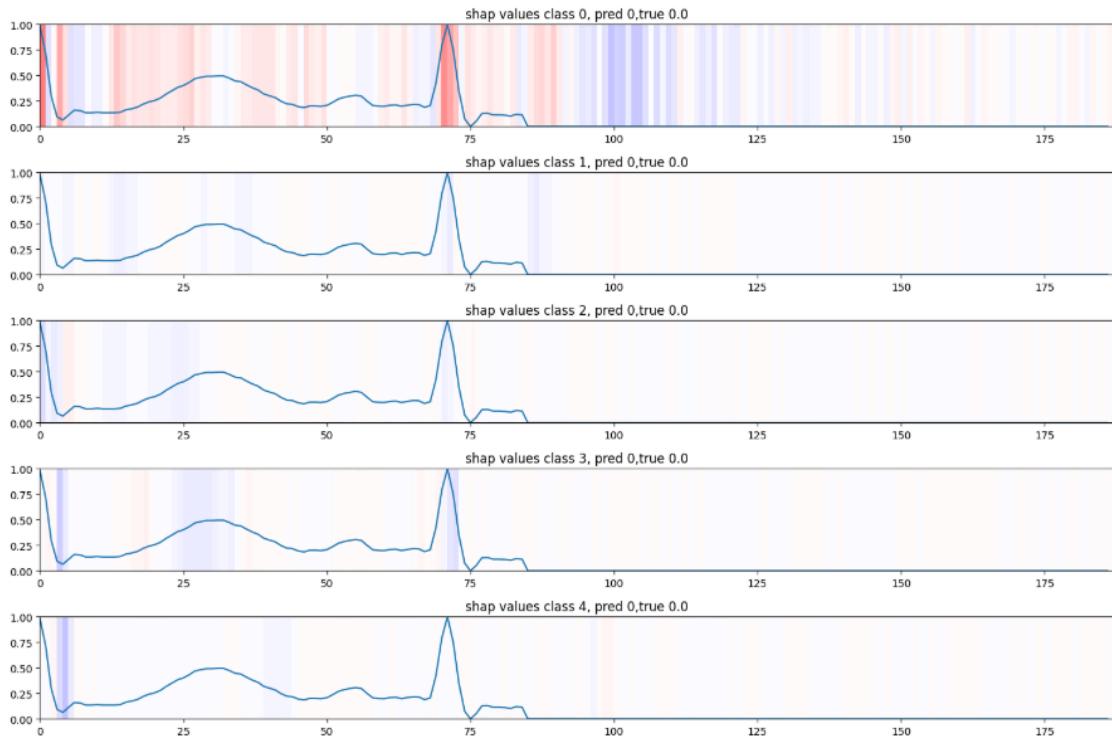


Figure 22: Shap values pour chaque classe d'un signal de classe 0

Ce signal est normal, l'amplitude des SHAP values est affichée en couleur pour chacune des 5 classes. La prédiction est correcte. Les valeurs rouges tirent le résultats vers la classe. Les quelques premiers échantillons et le pic R tirent le classifier vers la classe 0.

Sur cette autre exemple de classe 2, ce sont surtout les premiers échantillons qui permettent au signal d'être bien classé. Les valeurs proches de zéros au début du signal s'opposent à une classification en signal normal.

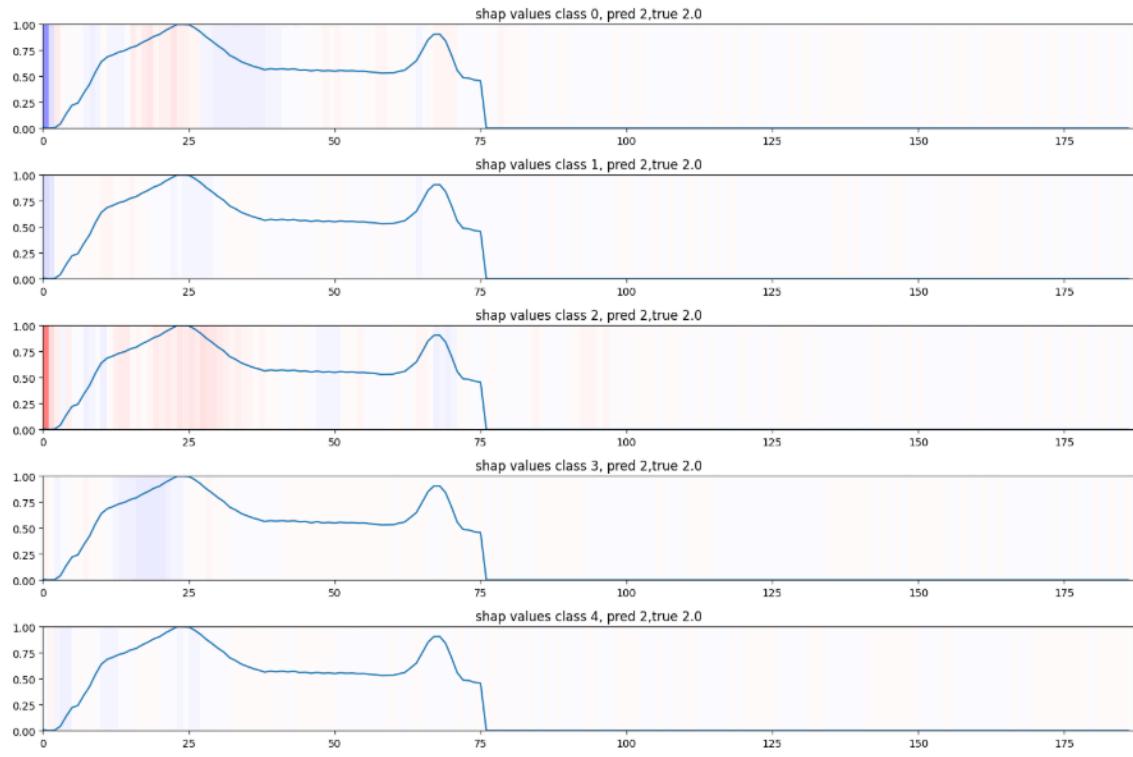


Figure 23: Shap values pour chaque classe d'un signal de classe 2

1.2.2. Réseaux de neurones convolutifs

Les réseaux de neurones convolutifs sont utilisés en computer vision pour détecter des caractéristiques, d'abord des contours, puis des objets, puis des scènes... Dans le cas du traitement du signal, nous aimerais que le modèle apprenne certaines formes : des pics, des plateaux.... De nombreux paramètres entrent en jeu dans ce type de modèle. Les premières tentatives durant la première phase de modélisation n'utilisaient que des couches de convolutions et un classifieur de type réseaux de neurones. La formation avançant, nous avons vu que généralement des couches de dropout et de normalisation étaient incluses entre chaque couche de convolution. D'autres modèles ont été créés incluant ces couches.

Nous avons fixé le premier kernel de convolution à 7 ou à 11 échantillons volontairement pour que le réseau arrive à voir des caractéristiques assez longues. L'ajout de batch normalisation a permis d'améliorer les résultats. Sur la fin, nous avons augmenté le dropout afin de diminuer l'overfitting. Ceci a permis encore d'améliorer les résultats sur la base de tests. Pour tous les essais à partir de la deuxième série de modélisation, nous avons utilisé une base de cross-validation et évaluer le modèle à la fin sur la base de test.

Nous avons aussi testé plusieurs fonctions d'activation.

Pre-entraînement

Nous avons aussi essayé d'utiliser les deux bases de données pour que la modélisation parte d'un modèle disons pré-entraîné. Cependant, nous avons laissé tous les paramètres libres pour la deuxième itération :

- Entrainement sur la base du dataset normal/anormal avec une classification en 2 classes avec un optimizer Adam et un learning rate à 0.001
- Changement de la dernière couche par une classification en 5 classes avec un learning rate de 0.0001 et entraînement sur la base de données 1.

Le but étant de tester si l'on pouvait avoir un modèle fonctionnant pour les deux bases de données en optimisant d'abord avec la base 2 puis en espérant être dans un bon minimum et ne pas trop dégrader le modèle avec la base 1.

Les résultats sur la base 1 sont satisfaisants mais n'ont pas permis d'avoir de bons résultats sur la base 2.

Le modèle overfit largement car quasiment aucune erreur est faite sur la base de cross-validation.

0	1.00	0.99	0.99	2476
1	0.99	1.00	1.00	2485
2	1.00	1.00	1.00	2516
3	1.00	1.00	1.00	2448
4	1.00	1.00	1.00	2575
accuracy		1.00		12500
macro avg		1.00	1.00	12500
weighted avg		1.00	1.00	12500
0	0.99	0.99	0.99	18118
1	0.84	0.84	0.84	556
2	0.96	0.96	0.96	1448
3	0.71	0.85	0.77	162
4	0.99	0.99	0.99	1608
accuracy		0.98		21892
macro avg		0.90	0.93	21892
weighted avg		0.98	0.98	21892

Figure 24 : overfitting manifeste

Tests de la taille des kernels

Dans ce test, nous faisons varier la taille des kernels de convolution tout en maintenant une architecture avec 3 couches convolutives. Les configurations de kernels testées sont les suivantes:

- 11,7,3 avec 80 filtres
- 7,5,3 avec 80 filtres
- 5,3,3 avec 80 filtres
- 5,3,3 avec 32, 64 et 128 filtres

Les résultats obtenus sont excellents et montrent peu de différences entre les configurations. Dans ce contexte, nous choisissons le modèle le plus léger.

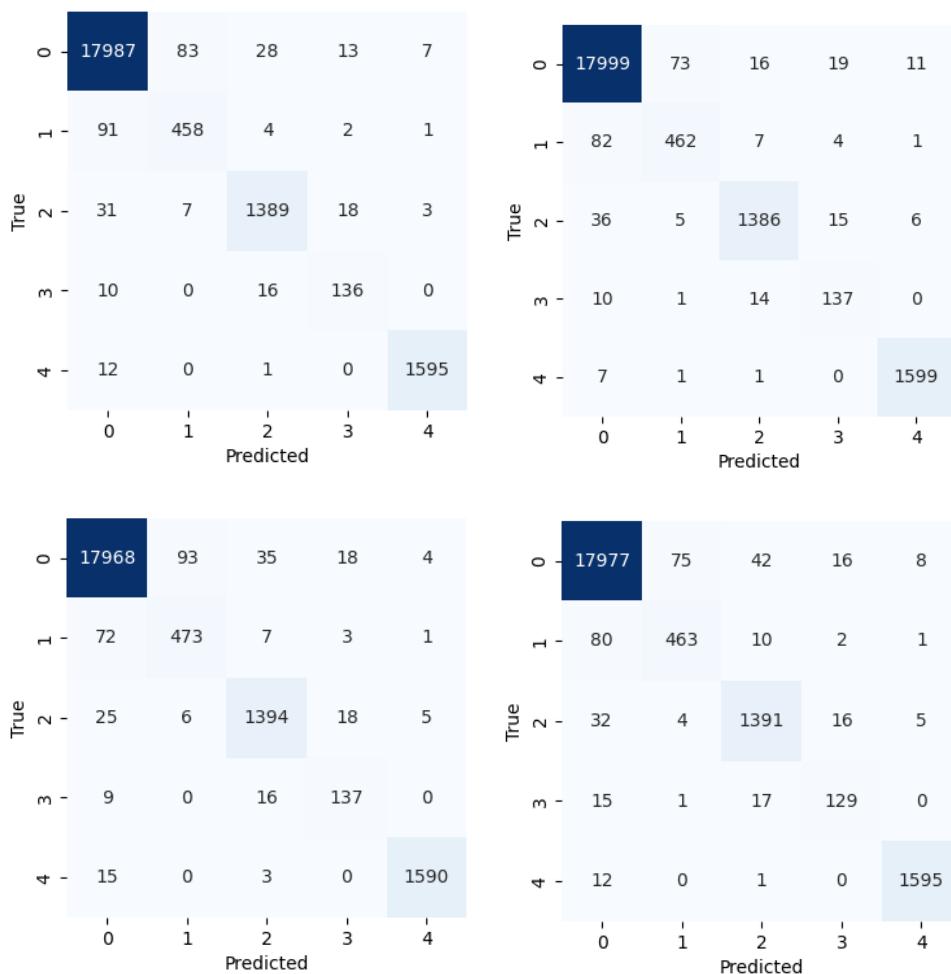


Figure 25 : matrices de confusions. Celle en rouge correspond au modèle retenu.

Le modèle retenu comporte 3 couches de convolution utilisant chacune 80 filtres, avec des kernels de tailles 5 puis 3 puis 3. Les performances sont très similaires à celles du modèle utilisant des kernels de tailles 11, puis 7 puis 3.

Nous avons visualisé les kernels de convolution, mais leur interprétation reste complexe. Nous aurions espéré identifier des kernels présentant, par exemple, une forte inclinaison dans une direction, susceptibles de détecter des variations marquées de pente dans le signal.

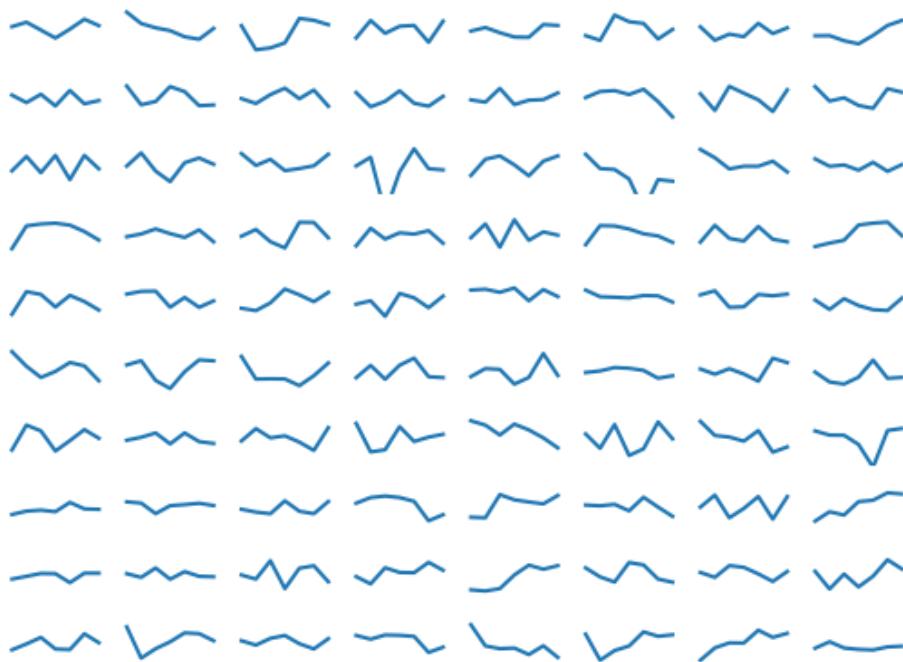


Figure 26: kernels de convolution du modèle retenu

Ci-dessous sont présentés les différents résultats des modèles inversés :

Tableau 3 : résultats d'évaluation de modèles CNN

modèle	Annotation	Nombre de signaux par classe	Nombre de paramètres	precision macro avg	recall macro avg	f1 score macro avg
CNN 1		13000	64389	0,8	0,92	0,85
CNN 2		13000		0,76	0,92	0,82
CNN 1		6000	64389	0,73	0,92	0,8
CNN 1		50000	64389	0,89	0,91	0,9
CNN 1	Smote seulement	50000	64389	0,83	0,91	0,87
CNN 3	gridsearch , dropout, normalisation	50000	89752	0,85	0,93	0,88
CNN4	pre-entraîné dataset2	50000	89752	0,9	0,93	0,89
CNN 5	kernels tests	50000	141905	0,92	0,92	0,92
CNN 6	kernels tests	50000	123825	0,92	0,92	0,92
CNN 7	kernels tests	50000	132785	0,91	0,93	0,92
CNN 8	kernels tests	50000	166257	0,92	0,92	0,92

Description détaillée du meilleur CNN

Le schéma suivant présente l'architecture du modèle donnant les meilleurs résultats

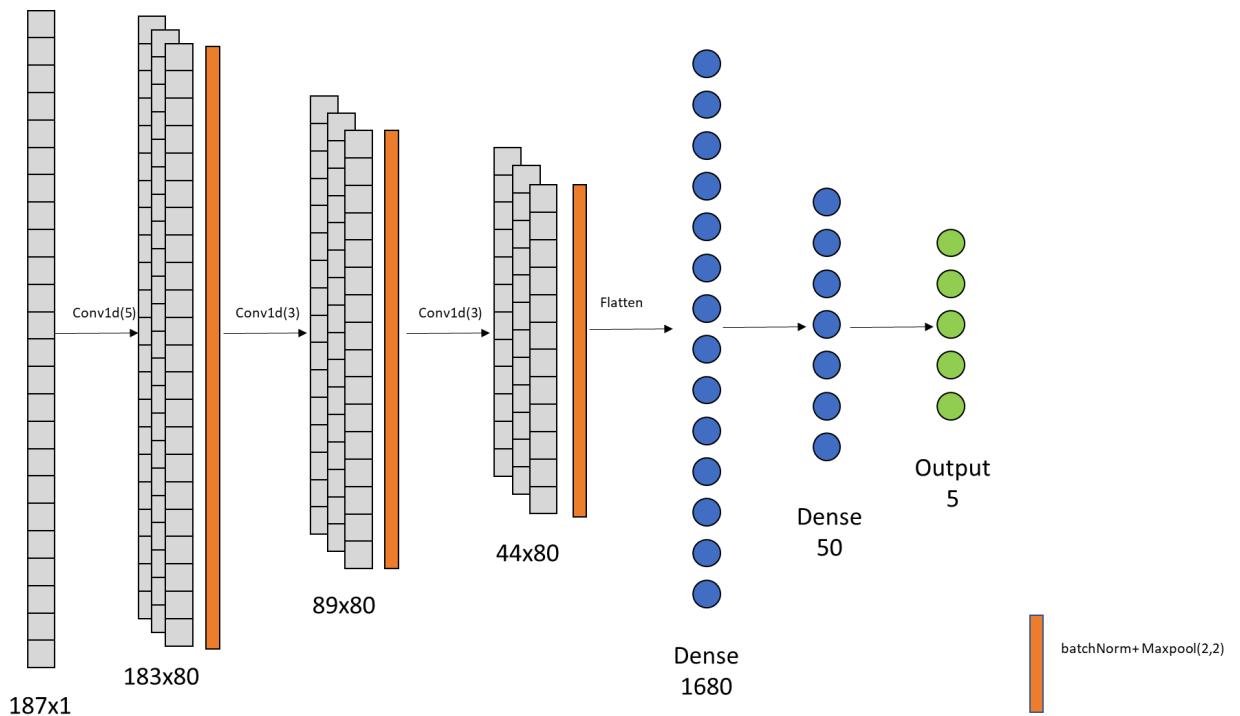


Figure 27: Architecture du modèle CNN retenu

Courbes d'entraînement

Le modèle est testé sur une base de cross-validation correspondant à 15% de la base d'entrée. Les données ayant été suréchantillonnées, la base de cross-validation est très proche de celle d'entraînement en termes de contenu.

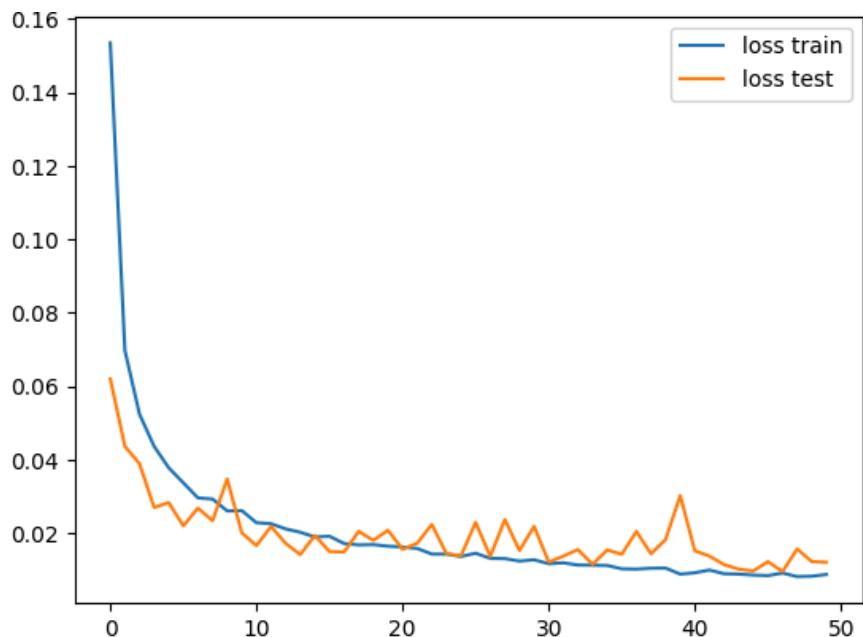


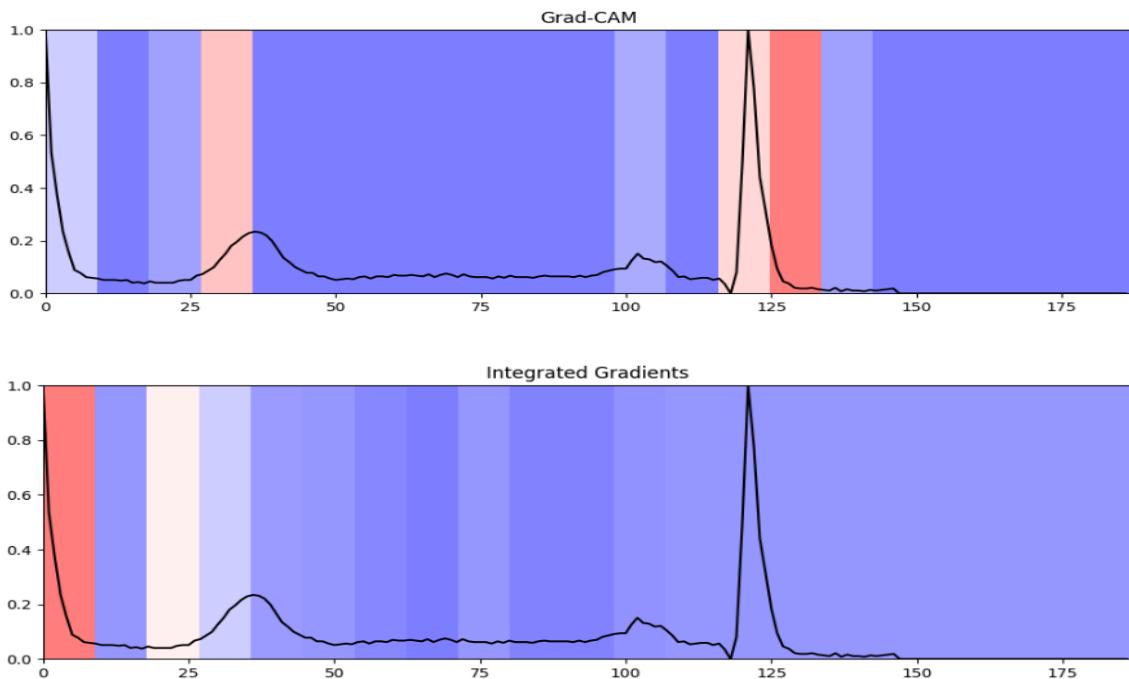
Figure 28: Courbe d'entraînement du meilleur modèle CNN

Difficulté d'analyse des erreurs.

L'analyse des erreurs est compliquée pour la simple raison que nous n'avons pas vraiment de critères permettant de comprendre la classification. Deux signaux que nous trouvons semblables à l'œil peuvent être classés de manière différentes. Il nous manque la connaissance métier qui aurait peut-être pu nous aider à se focaliser sur certaines parties du signal.

Certains modules, notamment dans la bibliothèque Captum, proposent des méthodes d'interprétabilité. Par exemple, le Grad-CAM et le Linear Gradient exploitent la dernière couche du modèle avant la classification pour mettre en évidence les zones activées. Une autre approche, appelée méthode d'occlusion, consiste à masquer progressivement une partie du signal et à effectuer des prédictions. Cela permet de calculer des coefficients reflétant l'importance de la zone masquée dans la prédiction. L'occlusion présente l'avantage d'être plus facile à comprendre et de mieux échantillonner les résultats. En revanche, les deux premières méthodes génèrent un nombre d'échantillons limité correspondant à la taille réduite du signal après les convolutions et les étapes de pooling.

Ci-dessous un exemple des 3 méthodes appliquées à un même signal.



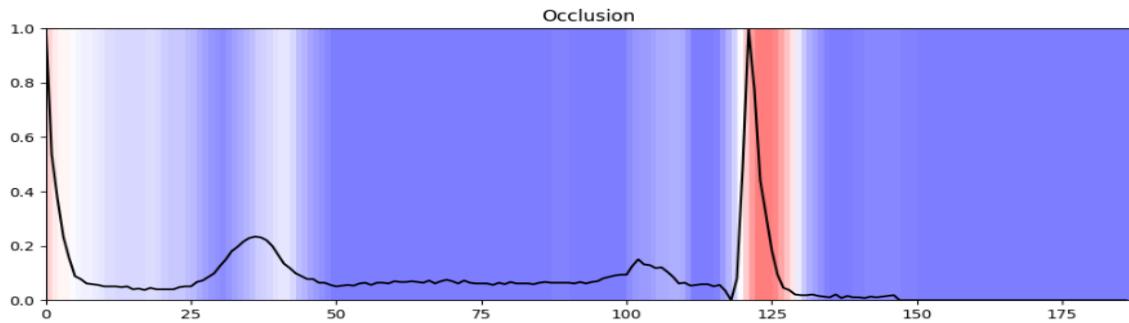


Figure 29: Application de méthodes d'interprétabilité de CNN

Ces résultats sont différents, la méthode des gradients intégrés donne plus de poids au début du signal. L'occlusion fait apparaître les zones mises en évidence par les deux autres méthodes. Nous allons regarder l'occlusion sur les différentes classes. Les figures ci-dessous montrent en rouge l'importance du signal dans la prédiction pour des signaux bien classés. Il semble que la classe 4 est déterminée sur la forme du système QRS, au début ou à sa répétition à la fin du signal.

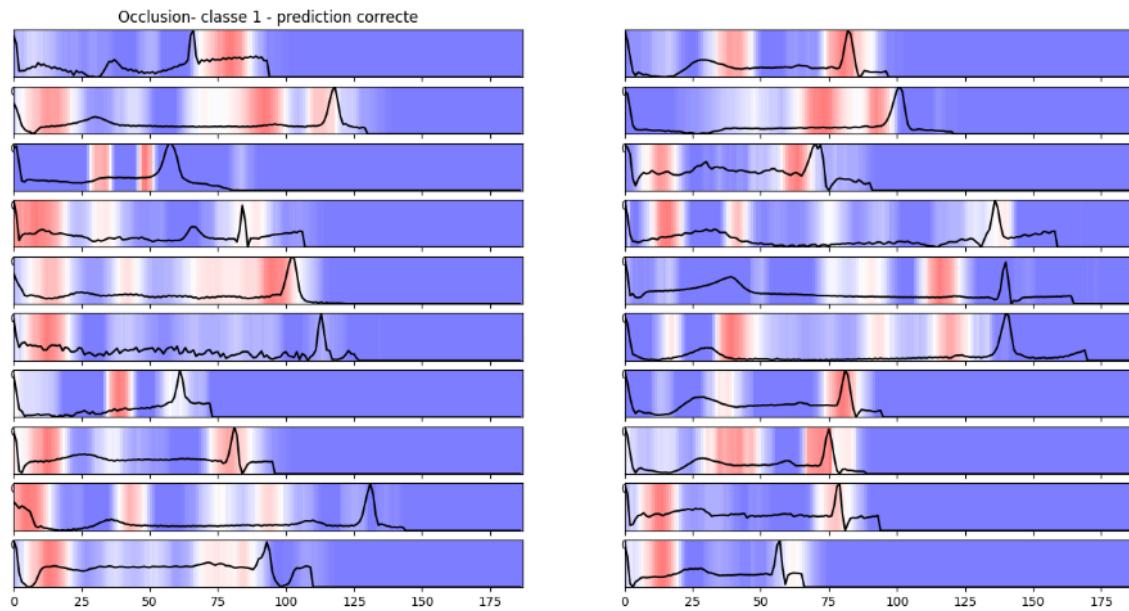


Figure 30: Application de la méthode d'occlusion à des signaux correctement classés en classe 1

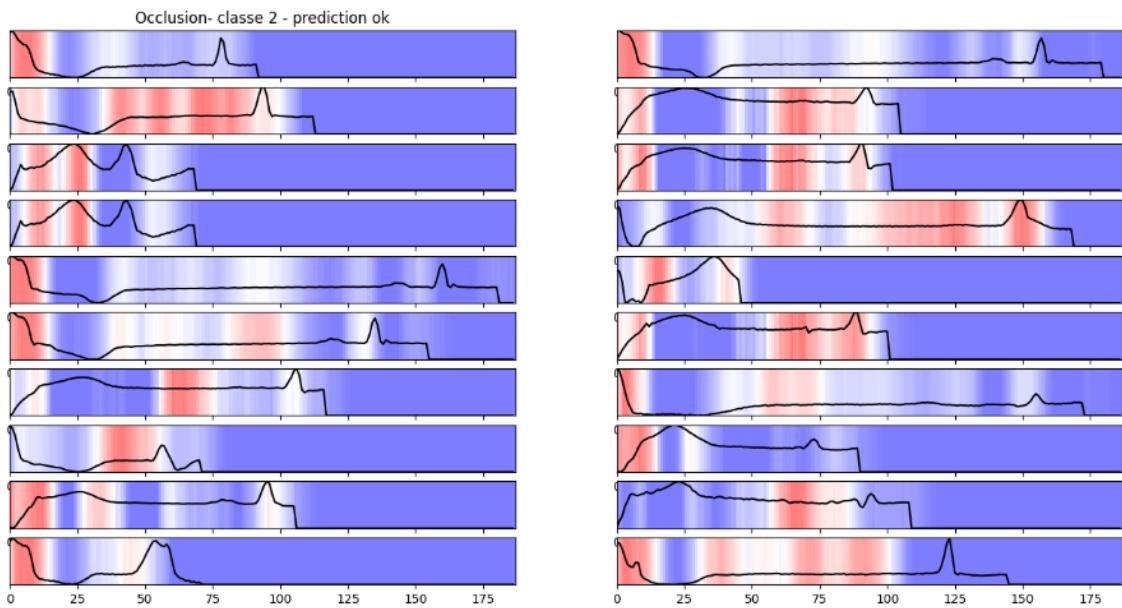


Figure 31: Application de la méthode d'occlusion à des signaux correctement classés en classe 2

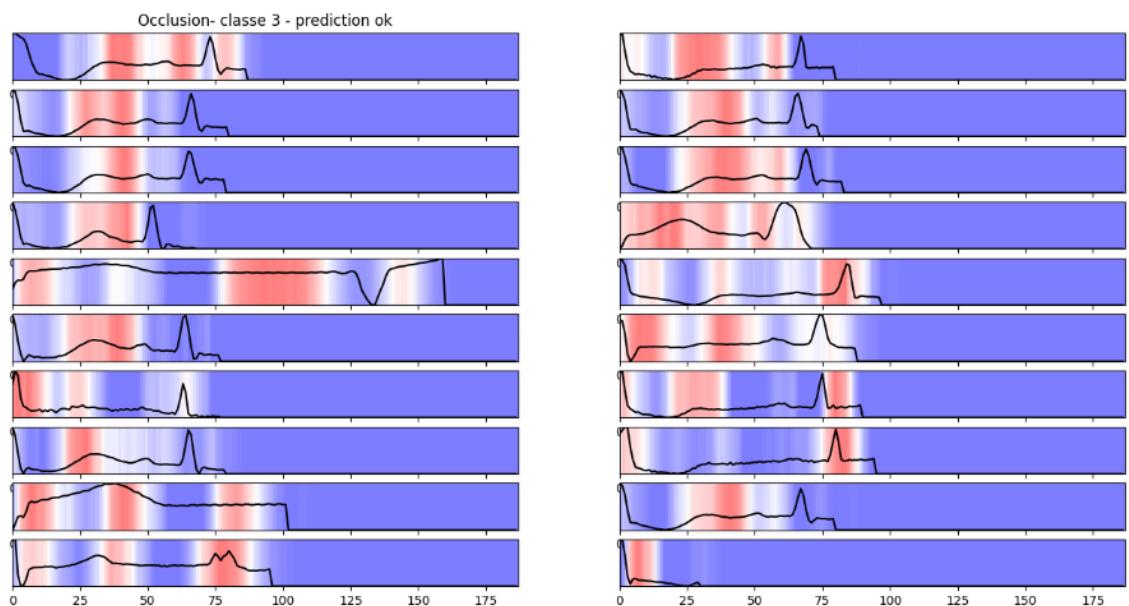


Figure 32: Application de la méthode d'occlusion à des signaux correctement classés en classe 3

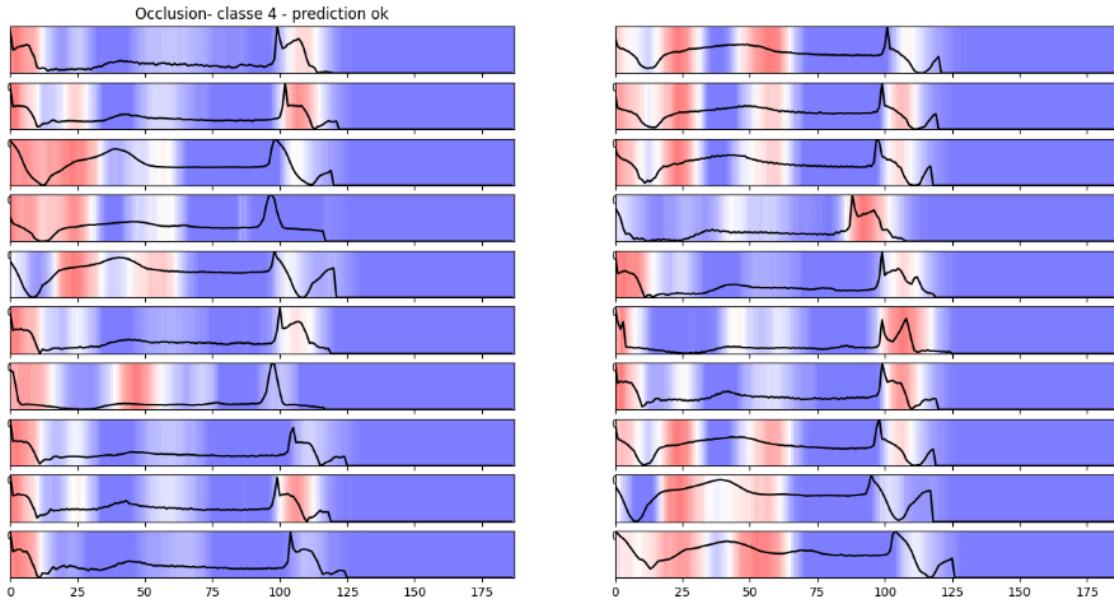


Figure 33: Application de la méthode d'occlusion à des signaux correctement classés en classe 4

Le début du signal semble jouer un rôle déterminant dans la classification, confirmant ainsi les hypothèses formulées lors de l'analyse des données. En effet, certains signaux, bien qu'incomplets et comprenant peu d'échantillons, étaient tout de même classés. Cela indique que la forme du signal associée au relâchement du cœur constitue un élément clé de la classification. De plus, ce signal, souvent observé en fin de données, semble également jouer un rôle important dans le processus. Cependant, le début du signal n'est pas le seul indice de classification. Sur la classe 3, la première bosse apparaît aussi comme élément révélateur.

1.2.3. Réseaux de neurones convolutifs et récurrents

Les réseaux de neurones récurrents (RNN) constituent une classe de modèles d'apprentissage automatique conçus pour traiter des données organisées sous forme de séquences, comme le texte, la parole ou les séries temporelles. Leur architecture repose sur des connexions récurrentes, leur permettant d'exploiter les informations provenant des étapes précédentes pour prendre des décisions en continu. Contrairement aux réseaux neuronaux traditionnels, les RNN se distinguent par leur aptitude à modéliser les relations temporelles dans les données. Cependant, ils présentent des limites lorsqu'il s'agit de capturer des relations à long terme, c'est-à-dire des connexions entre des éléments éloignés dans une séquence. Malgré ces contraintes, les RNN s'avèrent essentiels pour des applications variées telles que la traduction automatique, la synthèse de texte ou encore la reconnaissance vocale, grâce à leur capacité à contextualiser et à traiter efficacement les données séquentielles.

Dans le cadre du projet, plusieurs tests ont été réalisés :

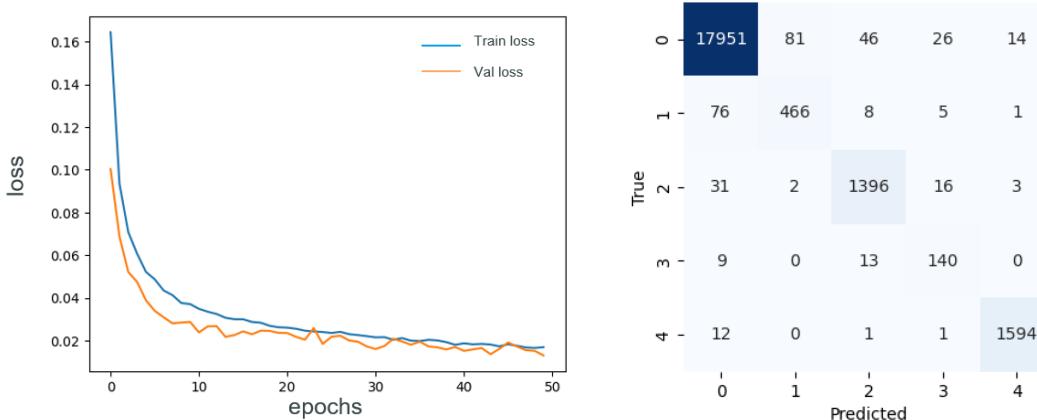
- une couche récurrente avec 32 hidden states et un réseaux de neurones
- une couche de convolution suivie d'une couche récurrente et une couche de neurones
- une couche GRU avec 32 hidden states et un réseaux de neurones
- une couche de convolution suivie d'une couche GRU et une couche de neurones
- 2 couches de convolutions suivies d'une couche GRU bi-directionnelle

Ces modèles adoptent une architecture relativement simple, car il était nécessaire de bien comprendre le fonctionnement des couches récurrentes. Au fil du temps, la complexité des modèles proposés a progressivement augmenté.

Tableau 4 : résultats d'évaluation de modèles Récurrents

modèle	Nombre de signaux par classe	Nombre de paramètres	precision macro avg	recall macro avg	f1 score macro avg
RNN	50000	31045	0,78	0,9	0,83
Conv + RNN	50000	11533	0,78	0,93	0,84
GRU	50000	33285	0,84	0,92	0,87
Conv + GRU	50000	14989	0,9	0,92	0,91
Conv x2 + Bi-dir GRU	50000	59845	0,9	0,93	0,92

Ci-dessous les courbes d'entrainements et la matrice de confusion du meilleur modèle. La courbe de loss est en dessous de la courbe de train car un dropout important a été utilisé.



	precision	recall	f1-score	support
0	0.993	0.991	0.992	18118
1	0.849	0.838	0.843	556
2	0.954	0.964	0.959	1448
3	0.745	0.864	0.800	162
4	0.989	0.991	0.990	1608
accuracy			0.984	21892
macro avg	0.906	0.930	0.917	21892
weighted avg	0.985	0.984	0.984	21892

Figure 34: Courbe d'entraînement et résultats sur la base de test d'un modèle récurrent avec une couche GRU bi-directionnelle

Ce modèle donne de bons résultats avec un nombre de paramètres bien inférieur au modèle CNN.

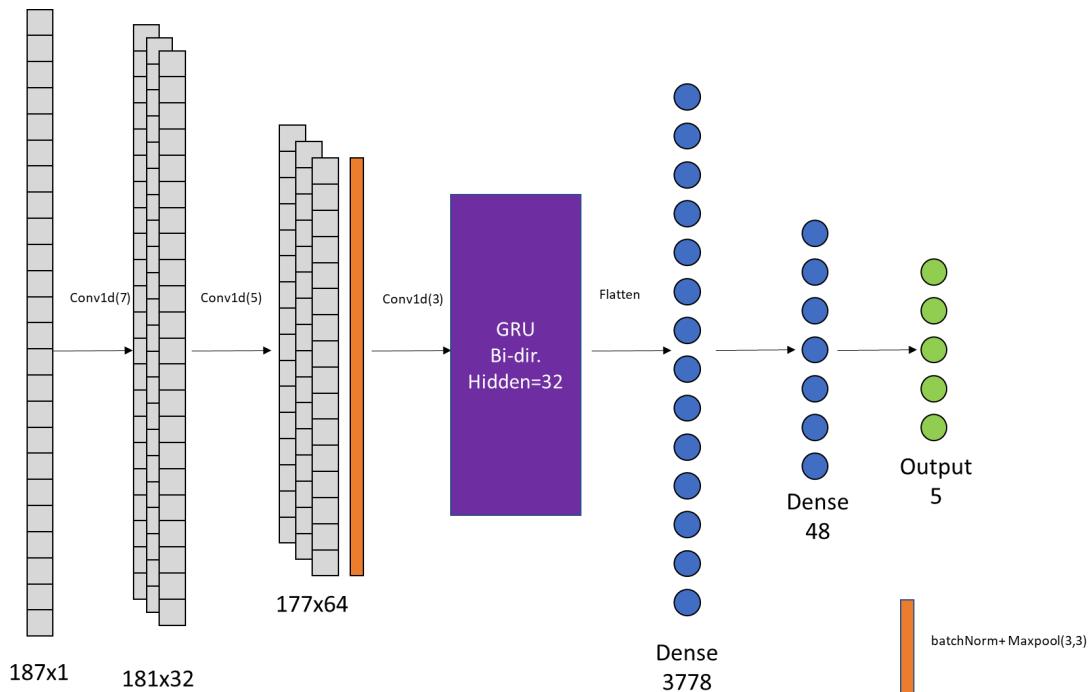


Figure 35: Architecture du meilleur modèle récurrent

1.2.4. Amélioration du modèle CNN

Dans cette section, nous visons à améliorer le modèle CNN sélectionné (Figure 27) en exploitant une base de données de signaux pré-calculés comprenant 50 000 signaux par classe. Pour chaque minibatch, des transformations aléatoires sont appliquées aux signaux afin de renforcer la robustesse du modèle :

- Une partie des signaux est décalée aléatoirement d'un nombre de points inférieur à la taille du padding (le nombre de zéros ajoutés pour uniformiser la taille des signaux à 187 points).
- Une partie des signaux est filtrée à l'aide d'un filtre gaussien avec un écart-type de 0,7.
- Enfin, certains signaux sont également filtrés avec un filtre gaussien ayant un écart-type de 0,5.

Note : l'application du filtre gaussien a un impact important sur les premiers et derniers échantillons du signal puisqu'il est basé sur une convolution. Notamment, le début du signal correspond généralement à une pic descendant. Pour ne pas altérer ce pic, le filtrage n'est appliqué qu'à partir du 5^e échantillon.

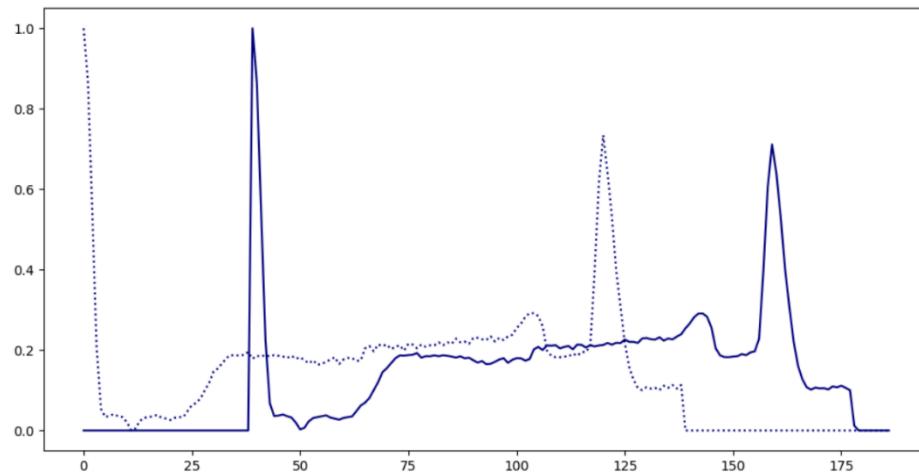


Figure 36 : exemple d'un signal ayant subi une transformation dans le mini batch

La première transformation vise à apprendre au modèle des caractéristiques indépendantes de leur position temporelle, permettant ainsi de mieux détecter des patterns récurrents quelle que soit leur localisation dans le signal. La deuxième transformation, quant à elle, consiste à atténuer les faibles variations, car la classification des signaux semble principalement dépendre de la forme des pics principaux. La taille des filtres gaussiens est volontairement restreinte afin de limiter l'atténuation excessive des hautes fréquences, qui pourrait altérer l'amplitude des pics les plus marqués.

Le modèle est entraîné sur 85 % de la base de signaux d'entraînement et évalué sur les 15 % restants. Une réduction automatique du learning rate a été mise en place, avec un facteur de 0,5 et une patience de 5 epochs.

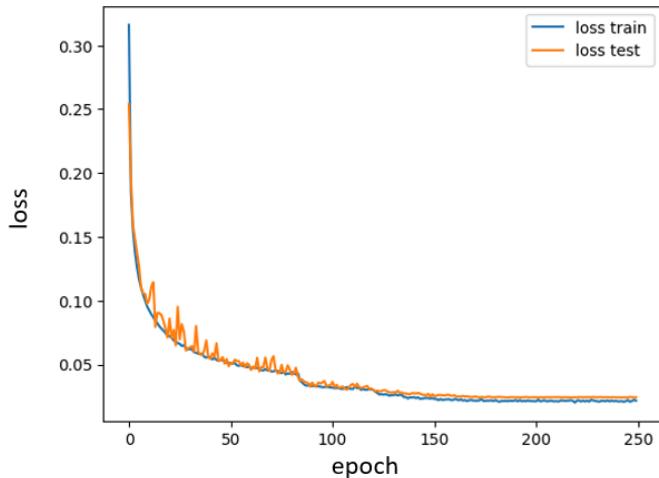


Figure 37: entrainement du modèle CNN avec data augmentation « on the fly »

Ce modèle fournit des résultats équivalents au modèle de départ. On ne constate pas d'amélioration significative.

	precision	recall	f1-score	support
0	0.993	0.991	0.992	18118
1	0.830	0.836	0.833	556
2	0.957	0.960	0.959	1448
3	0.818	0.858	0.837	162
4	0.988	0.991	0.990	1608
accuracy			0.984	21892
macro avg	0.917	0.927	0.922	21892
weighted avg	0.984	0.984	0.984	21892

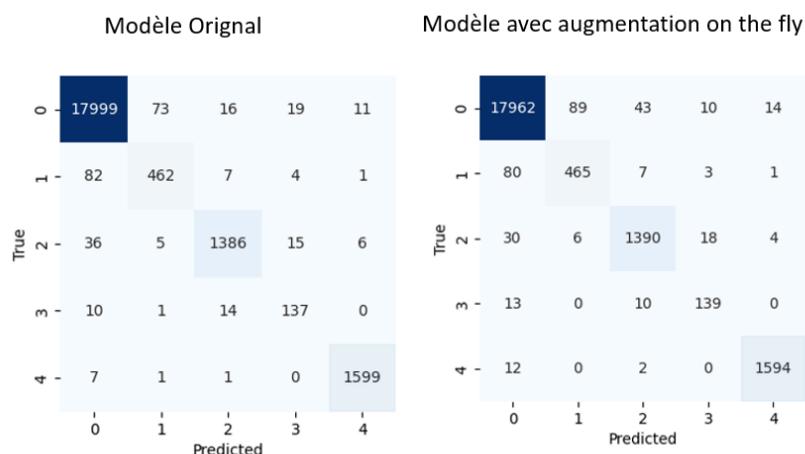


Figure 38: Matrices de confusion pour le modèle de référence (gauche) et le nouveau test (droite)

1.2.5. Transfer learning

Le transfer learning est une technique d'apprentissage automatique qui consiste à réutiliser un modèle préalablement entraîné sur une tâche spécifique pour en résoudre une autre, souvent similaire ou complémentaire. Cette méthode permet de tirer parti des connaissances acquises sur de vastes ensembles de données pour s'attaquer à des problèmes disposant de ressources plus limitées, tout en réduisant les temps d'entraînement et les besoins en puissance de calcul. Les signaux ECG, initialement de nature temporelle, ont été transformés en signaux 2D à l'aide de la transformée en ondelette continue (CWT), une méthode permettant de convertir des données temporelles en images spectrales. Un modèle de classification d'images, préalablement entraîné sur un ensemble de données généralistes, a ensuite été ajusté pour reconnaître et classer ces représentations transformées, exploitant ainsi les caractéristiques complexes inhérentes aux signaux ECG.

Cette approche devait en lumière le potentiel du transfer learning pour aborder des problématiques a priori éloignées, tout en améliorant les performances grâce aux représentations riches et généralisées fournies par des modèles pré-entraînés.

Cependant, les résultats obtenus demeurent insatisfaisants. Une approche plus méthodique et un temps d'investigation supplémentaire auraient été nécessaires, notamment en travaillant sur les données et en testant différentes transformées..

Nous avons opté pour un classifieur 2D plutôt qu'un classifieur 1D. Les modèles 1D existants sont souvent conçus pour des applications telles que le traitement de la voix, où les données sont généralement échantillonnées à des fréquences de plusieurs dizaines de kHz. Dans notre cas, avec une fréquence maximale de 62,5 Hz, notre signal contient relativement peu de points. Tenter de suréchantillonner ce signal aurait conduit à des séquences très courtes ou à un contenu fréquentiel bien inférieur à celui de la voix. Par ailleurs, de nombreux modèles de classification d'images sont optimisés pour des entrées de quelques centaines de pixels, ce qui s'adapte mieux à nos données.

Deux tests ont été effectués :

- L'un avec le modèle efficientNet B0 dont les 3 dimensions de l'image sont identiques correspondant à une transformée en ondelette continue de type Mexican Hat.
 - L'un avec le modèle ResNet50 dont chaque dimension de l'image correspond à une transformée en ondelette continue utilisant les ondelettes Gauss, Mexican Hat et Morlet.
- Ce deuxième test est bien plus lourd en temps de calcul puisqu'il faut réaliser 3 transformées. Les librairies utilisées ne sont pas optimisées pour des Tenseurs.

La base de données utilisée correspond à la base du MIT-BIH transformée par modification du signal comme évoqué au paragraphe 2.6.2 pour obtenir 10000 signaux par classes.

Il a été testé lors de chaque batch un nombre aléatoire de signaux soit augmenté par décalage temporel et/ou application d'un léger filtrage gaussien. Etonnamment ceci n'a pas fonctionné. Ces étapes ont été aussi supprimées pour voir si les performances dépendaient d'une mauvaise augmentation de données.

L'entraînement est réalisé avec Kaggle permettant d'avoir 30 h de GPU par semaine

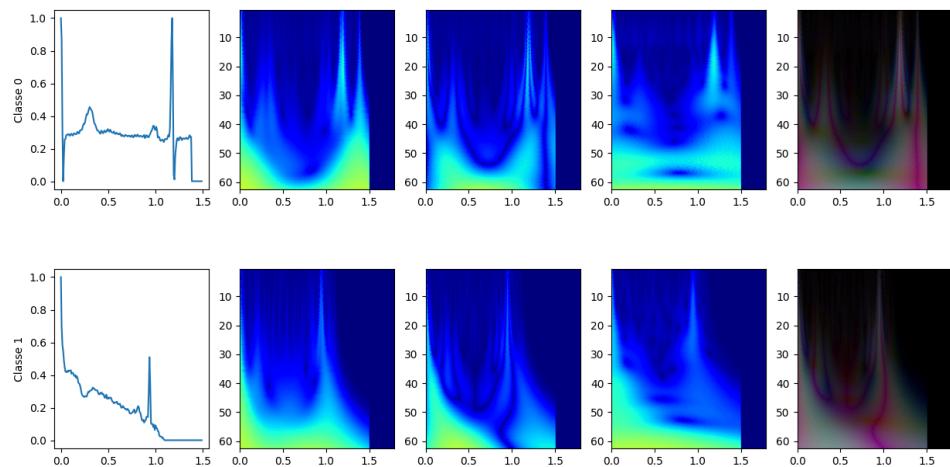
Transformée en ondelettes

La transformée en ondelettes continues nous est familière mais nous ne sommes pas experts en la matière. Cette technique consiste à utiliser une ondelette de référence, puis à calculer le produit scalaire entre le signal et cette ondelette décalée et dilatée à différentes échelles. Cette opération permet de convertir un signal 1D en une représentation 2D sous forme d'image, où les axes correspondent aux coefficients d'échelle et de décalage temporel.

La normalisation des résultats a été effectuée de manière empirique, en divisant les coefficients par le maximum des valeurs obtenues sur la base d'entraînement. Il est probable qu'une formulation analytique plus rigoureuse existe pour cette étape, mais nous ne l'avons pas explorée dans ce travail.

L'application de la transformée en ondelettes continues implique plusieurs choix importants, notamment la sélection des facteurs d'échelle et du type d'ondelette à utiliser. Ces paramètres influencent directement la résolution temporelle et fréquentielle de la transformation.

La figure suivante montre les transformations réalisées pour un signal de chaque classe avec les ondelettes de gauss, mexican hat et de morlet. La dernière image est l'image obtenue en RGB en utilisant les 3 transformées comme un canal de l'image.



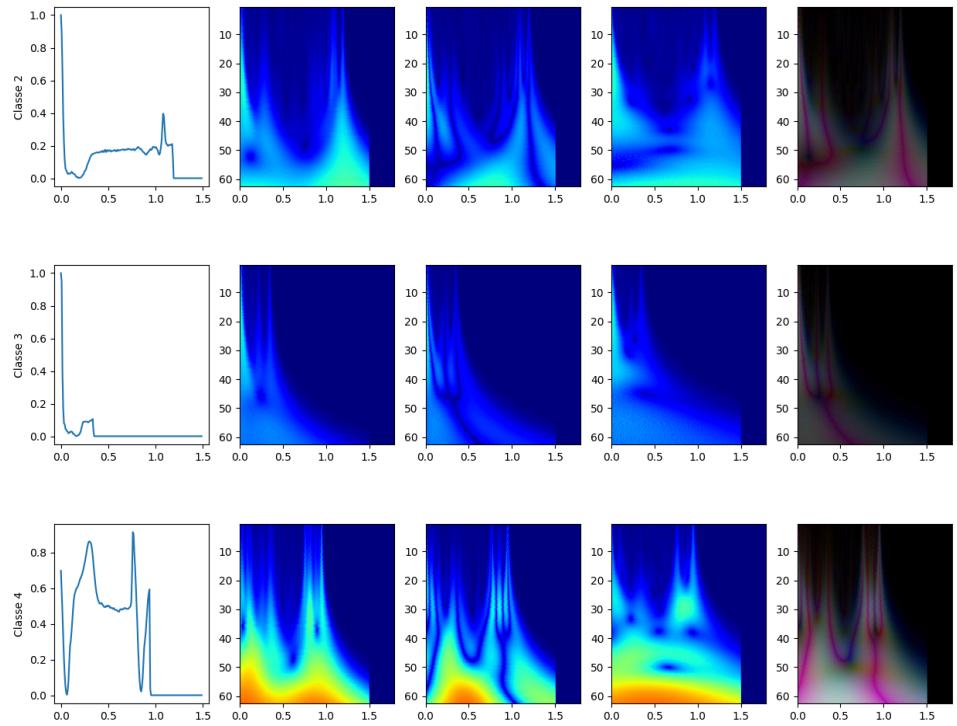


Figure 39: Transformée en ondelettes continue. Les 3 images centrales représentent les transformées du signal à gauche depuis 3 ondelettes différentes. L'image de droite intègre les 3 transformées comme canaux RGB

Modèle efficientNet

La dernière couche du classifier du modèle EfficientNet est remplacée par un classifieur de deux couches de neurones [2048,128] puis [128,5].

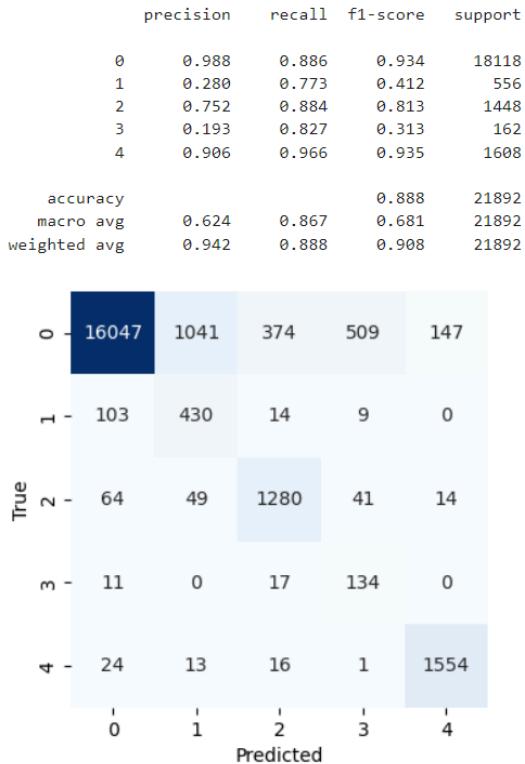


Figure 40: Résultats sur la base de test du modèle EfficientNet

Modèle ResNet50

La dernière couche du classifier du modèle ResNet50 correspondant en un réseau de de taille [2048,1000] permettant d'avoir 1000 classes en sortie est remplacée par un classifieur de deux couches de neurones [2048,128] puis [128,5]. Les résultats ne sont pas meilleurs avec une mauvaise précision.

Transformation en image par couche de convolution.

Un dernier test est réalisé en modifiant un modèle existant : une première couche de convolution est ajoutée avant le modèle ResNet. Cette couche a pour rôle de convertir le signal 1D en un signal 2D de dimensions (3, 224, 224) à l'aide de 224 filtres de taille 3x3, les canaux 2 et 3 sont identiques au canal 1. Bien que les résultats obtenus soient meilleurs, ils restent inférieurs à ceux du modèle entièrement convolutif. L'objectif principal est de permettre au modèle de générer lui-même l'image optimale. Intuitivement, ce type de transformation semble peu adapté à un modèle de classification d'images, qui a été entraîné à reconnaître des formes directement issues de cette transformée. Cependant, l'approche a tout de même été testée par curiosité pour évaluer son impact.

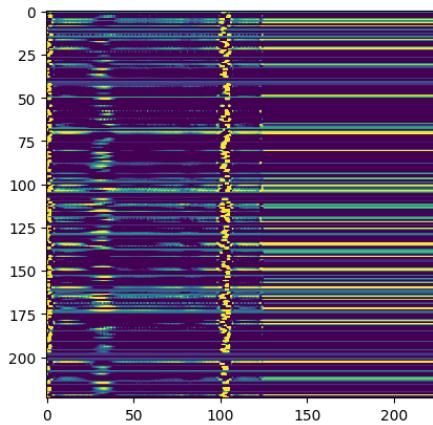


Figure 41: exemple de transformation par 224 convolutions 1D avec des poids entraînés (sortie de la première couche avant l'entrée dans le modèle ResNet)

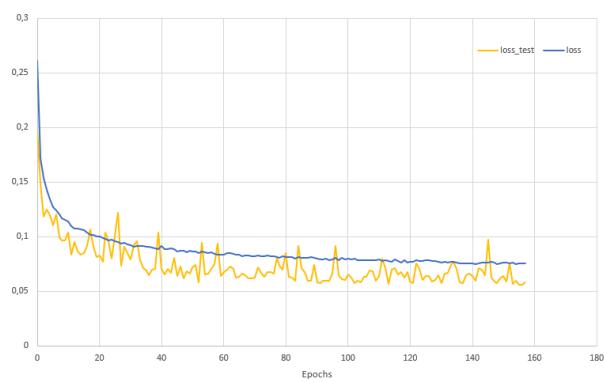


Figure 42: Courbe de loss sur 150 epochs

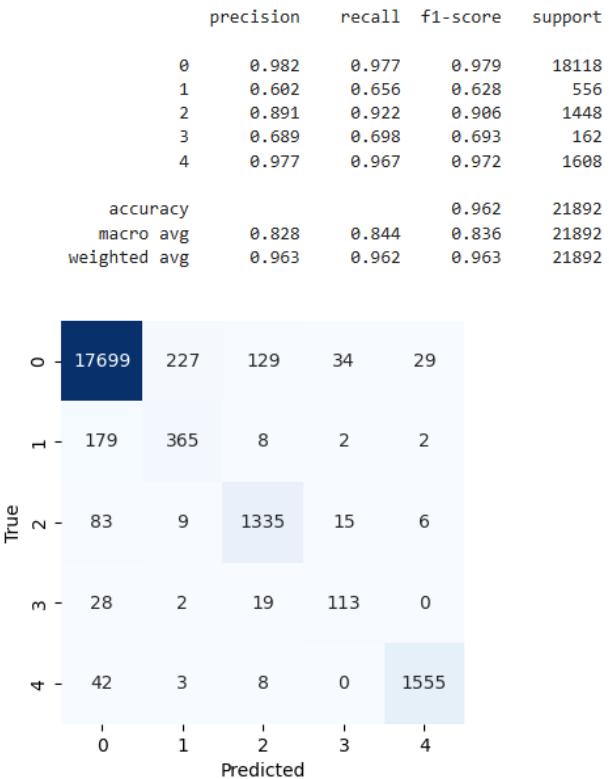


Figure 43: Résultats de transfer learning

1.3. Classification binaire

Dans cette section annexe, nous nous focalisons sur la classification binaire des électrocardiogrammes, visant à distinguer les signaux normaux des signaux anormaux. Pour cela, les deux bases de données ont été fusionnées, et la base multi-classe a été convertie en une classification binaire. Les classes 1, 2, 3 et 4 ont été regroupées pour être considérées comme des signaux anormaux, tandis que les signaux de la classe 0 représentent les cas normaux.

classe	base 1	base 2	total
0	72471	4046	76517
1	15083	10506	25589

Figure 44: nombre de signaux normaux (0) et anormaux (1) pour les 2 bases de données.

La base d'entraînement est constituée de 25 589 signaux anormaux, auxquels s'ajoute un nombre équivalent de signaux normaux sélectionnés de manière aléatoire. La base de test correspond au fichier de test fourni par le MIT, après transformation en classification binaire.

Le processus d'entraînement inclut les modifications des signaux par filtrage gaussien et décalage temporel, conformément aux méthodes décrites au paragraphe 7.2.4. En revanche, les transformations d'amplification des signaux n'ont pas été appliquées, car la taille de la base initiale est jugée suffisamment importante.

Pour référence, le modèle CNN donne sur la base de test transformée :

		0	1
True	0	17999	119
	1	135	3639
		Predicted	

Deux modèles sont testés :

- Le modèle CNN de référence
- Une architecture légèrement différente se rapprochant de l'architecture AlexNet (Figure 45)

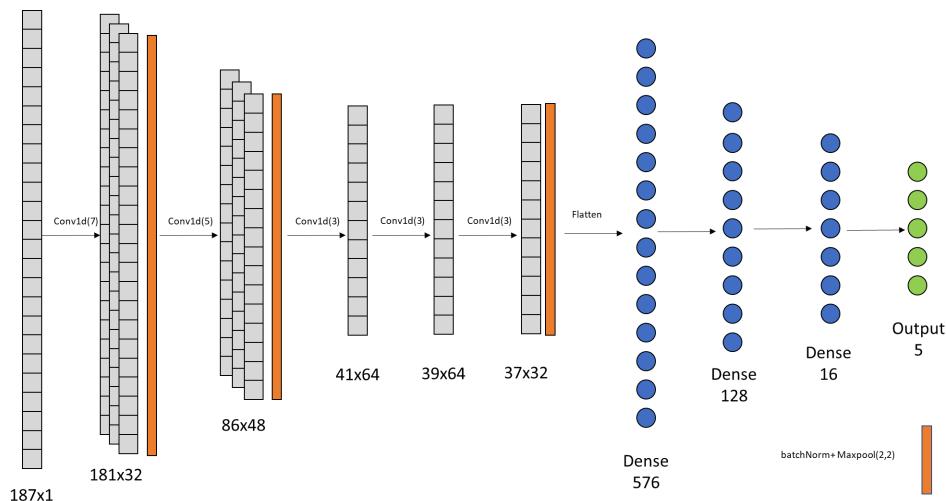
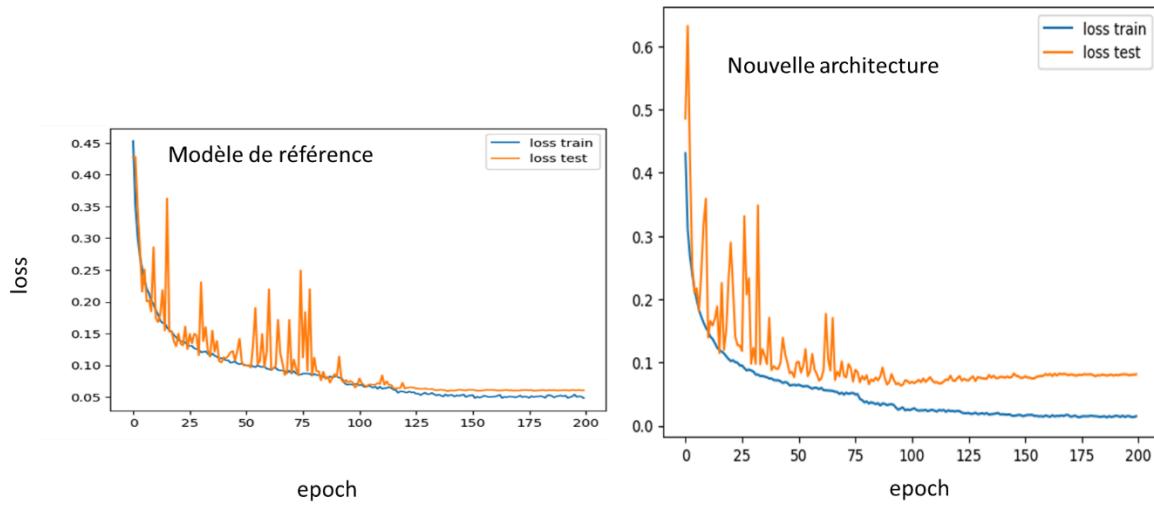
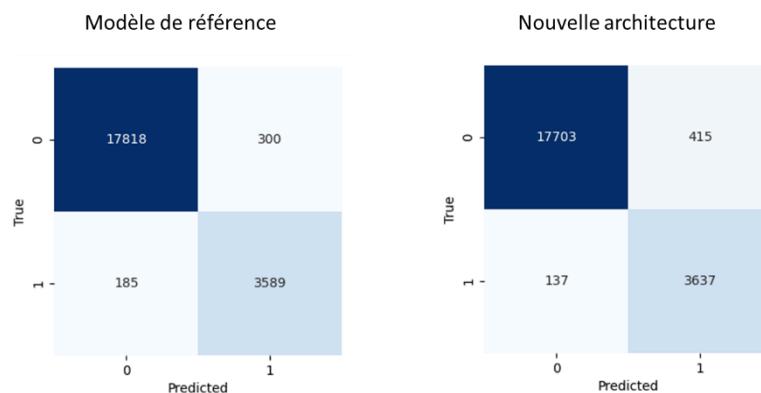


Figure 45: Architecture de modèle pour classification binaire callée sur AlexNet

Ci-dessous sont présentées les courbes d'entraînement sur 200 epochs.



Les résultats obtenus sont équivalents, cependant la nouvelle architecture reconnaît plus de signaux anormaux. Le nombre de faux négatifs (signaux anormaux classés en normaux) est moins élevé, ce qui dans le domaine médical est probablement plus important que des faux positifs (signaux normaux classés en anormaux)



Modèle de référence					Modèle testé				
	precision	recall	f1-score	support		precision	recall	f1-score	support
0	0.990	0.983	0.987	18118	0	0.992	0.977	0.985	18118
1	0.923	0.951	0.937	3774	1	0.898	0.964	0.929	3774
accuracy			0.978	21892	accuracy			0.975	21892
macro avg	0.956	0.967	0.962	21892	macro avg	0.945	0.970	0.957	21892
weighted avg	0.978	0.978	0.978	21892	weighted avg	0.976	0.975	0.975	21892

Cette annexe présente le seul test réalisé sur les bases fusionnées. Cependant, il comporte un certain biais, car la base de test est exclusivement composée de signaux provenant de la base MIT-BIH, ce qui peut limiter la représentativité des résultats obtenus et influencer l'évaluation des performances du modèle.

2. Discussion générale et conclusion

Plusieurs types de modèles ont été évalués, incluant des approches de machine learning utilisant à la fois des attributs du signal et le signal brut. Une observation initiale commune à tous ces modèles est leur faible performance lorsqu'ils sont entraînés sur une base de données avec peu d'échantillons. Pour obtenir des résultats significatifs, il a été nécessaire d'augmenter considérablement la taille de la base. Les meilleures performances ont été obtenues avec un réseau convolutif (CNN).

Les résultats du meilleur modèle sont les suivant :

Tableau 5 : résultats d'évaluation du modèle retenu

Classe	y	Accurac		F1-sc
		Recall	orr	
0	0.993	0.993	0.993	
1	0.852	0.831	0.842	
2	0.973	0.957	0.965	
3	0.783	0.846	0.813	
4	0.989	0.994	0.992	
macro				
avg	0.918	0.924	0.921	
weighted				
avg	0.986	0.986	0.986	

Concernant la construction du modèle, son architecture reste relativement simple et s'inspire des modèles classiques utilisés en classification, en particulier en vision par ordinateur. Cependant, ce modèle a été développé bien avant les cours et les masterclasses de Deep Learning, ce qui a influencé sa construction dû au manque de connaissances. Par exemple, des couches de normalisation et de dropout n'ont pas été intégrées initialement, alors qu'elles auraient sans doute été ajoutées avec un recul plus important, ces modules ayant un impact significatif sur les performances. Un autre exemple concerne un test de grid search réalisé sur des hyperparamètres de deep learning, en s'inspirant des approches couramment utilisées en machine learning classique. Cependant, nous avons appris par la suite que cette méthode est rarement employée dans le contexte du deep learning.

Une deuxième conclusion importante est liée à l'overfitting. Sur la base de test, les meilleurs modèles affichent des taux de précision et de rappel supérieurs à 99,5 %, mais cet

excellent score masque un problème de sur-apprentissage. Cela se manifeste également le second jeu de données. On rappelle que deux stratégies de modélisation étaient possibles :

- Classifier les signaux comme normaux ou anormaux en utilisant les deux bases combinées.
- Classifier le type d'anormalité en se limitant à une seule base.

Les types d'anomalies sont variés, seules trois étaient présentes dans le jeu de données principal, (une quatrième catégorie regroupait les signaux inclassifiables). Les modèles appliqués au jeu de données à classification binaire ne sont pas fiables. L'overfitting peut l'expliquer, mais aussi des types d'anomalie présentes dans le jeu binaire mais non présentes dans jeu d'entraînement. Une utilisation de la première stratégie aurait peut-être masqué cet overfitting tout en simplifiant le problème.

Enfin, l'évaluation sur la base de test a été réalisée en tenant compte des scores moyens par classe en raison du déséquilibre de cette base. Les scores globaux (souvent supérieurs à 99 %) peuvent être trompeurs, car ils occultent des performances faibles pour certaines classes spécifiques.

La combinaison des deux jeux de données représente probablement une piste majeure d'amélioration pour le modèle. Une approche pourrait consister à les traiter avec deux sous-modèles distincts : le premier chargé d'identifier la présence d'une anomalie, et le second, lorsqu'il est possible, de classifier le type d'anomalie. L'utilisation des deux bases devraient réduire l'overfitting. De façon général, extraire d'autres données est un point important pour tenter de réduire l'overfitting.

Et finalement, plus de connaissance métier auraient évidemment aidé à peut-être mettre en évidence certains aspects du signal par un filtrage ou une transformée qui augmenterait les caractéristiques de ce que doit apprendre le modèle. Aucun de nous n'avait la moindre connaissance dans le domaine médical.

Un autre axe d'amélioration concerne l'utilisation de réseaux récurrents. Les tests menés avec ce type de modèles ont montré de très bons résultats, même avec des architectures simples et avec peu de paramètres. En raison de la complexité théorique de ces réseaux, nous avons limité notre exploration à une seule couche récurrente. Cependant, il est fort probable que des réseaux légèrement plus complexes puissent offrir des performances encore meilleures, notamment en utilisant des modèles de type Transformers.

Un troisième axe d'amélioration réside dans l'utilisation du transfer learning. Nous avons testé des modèles de classification d'images. Ces modèles sont nombreux. Nous n'avons pas forcément pris le temps de chercher des modèles basés sur la reconnaissance vocale qui sont peut-être plus appropriés.

De nombreuses approches restent aussi possibles comme les autoencodeurs. Nous n'avons pas pu explorer cette piste.

Il manque aussi dans ce projet toute la partie découpage, ré-échantillonnage et normalisation d'un enregistrement brut qui est nécessaire pour évaluer par exemple sur des signaux enregistrés en temps réel.