Interpretability of Neural Networks applied to Electrocardiograms : Translational Applications in Cardiovascular Diseases

Ahmad Fall

Sorbonne Université

Université Cheikh Anta Diop de Dakar

École Doctorale Informatique, Télécommunications et Electronique

EDITE de Paris - ED130

Unité de Modélisation Mathématique et Informatique des Systèmes Complexes (UMMISCO)

Interpretability of Neural Networks applied to Electrocardiograms: Translational Applications in Cardiovascular Diseases

Par Ahmad Fall

Thèse de Doctorat en Informatique

Dirigée par Pr. Jean-Daniel Zucker et Pr. Alassane Bah

Encadrée par Dr. Edi Prifti et Dr. Mandicou Ba

Présentée et soutenue publiquement le *(29 Novembre 2023)*

Devant un jury présidé par **Pr. Florence D'Alché-Buc** et composé de :

Alassane Bah, *Professeur, UCAD/UMMISCO, Directeur de thèse (Sud)*
Mandicou Ba, *Docteur, UCAD/ESP, Co-Encadrant (Sud)*
Antoine Cornuejols, *Professeur, AgroParisTech, Rapporteur*
Florence D'Alché-Buc, *Professeur, Télécom Paris, Examinateur*
Rémi Dubois, *Professeur, IHU Liryc, Université de Bordeaux, Rapporteur*
Edi Prifti, *Docteur, IRD/SU/UMMISCO, Encadrant (Nord)*
Joe Elie Salem, *MD - Professeur PUPH, APHP, Examinateur*
Jean-Daniel Zucker, *Professeur, IRD/SU/UMMISCO, Directeur de thèse (Nord)*

Acknowledgements

I would first and foremost like to express my deepest gratitude to Pr. Jean-Daniel Zucker and Pr. Alassane Bah, who agreed to supervise and oversee my work and whose guidance, support and expertise have been invaluable throughout this journey. I would also like to sincerely thank my advisors Dr Mandicou Ba and Dr Edi Prifti for providing me with a very close and valuable mentoring as well as significant contribution to my work.

Throughout this PhD, I had the opportunity to be part of the DeepECG4U project and to contribute to it and for that I would like to thank again Edi Prifti and to also thank Joe-Elie Salem. I'm grateful for having the opportunity to contribute to such an important project with significant impact on science. I am excited to pursue this work we started; there is so much to be discovered.

I would like to extend my sincere gratitude to both referees Pr Antoine Cornuejols and Pr Rémi Dubois who have accepted to review this work. I am also grateful to Pr Florence D'Alché-Buc and MD. PUPH Joe-Elie Salem who have accepted to participate in the jury and examine my work. Thank you.

My sincere thanks also go to my colleagues and friends at UMMISCO and IRD generally and Sorbonne Université who contributed and supported this study till this very day.

I also wish to express my appreciation and gratitude to the committee of the *Programme Doctoral International* PDI, Sorbonne Université, Cheikh Anta Diop University of Dakar and the French National Institute of Research for Sustainable Development (IRD) for giving me the opportunity to conduct this PhD and for funding it. Thank you Christophe Cambier and Elisabeth Pereira and my supervisors, advisors, for your administrative implication particularly at the rough beginning of this journey.

I cannot conclude without acknowledging the unfailing support of my family. To my parents, thank you for your belief in my capabilities. Your unwavering faith motivated me in my most challenging moments.

The pursuit of this PhD has been a remarkable journey, and for all the guidance, support, contribution and opportunities I received during this time, I am grateful.

Abstract

Electrocardiograms (ECGs) are non-invasive tools for assessing the electrical activity of the heart, they are widely used to detect cardiac abnormalities. Deep learning algorithms enable automatic detection of complex patterns in ECG data, offering significant potential for improved cardiac diagnosis. However, their adoption is hindered by a low level of trust among medical professionals and a substantial need for data to train the models. Artificial intelligence, particularly deep learning, allows for exploration of hierarchical representations of complex data, leading to a better understanding of internal interactions. Nevertheless, interpretability of the models are crucial to gain specialists' trust and facilitate widespread implementation. This thesis aims to develop a novel interpretability algorithm for neural networks applied to ECG analysis, working in close collaboration with cardiology specialists. Our study focuses on a specific cardiac pathology, Torsades-de-Pointes (TdP). TdP is a life threatening arrhythmia associated with various factors, including medications and congenital mutations. Accurate prediction of this risk can enhance patient care and potentially save lives. We started by designing a neural network algorithm for predicting the risk of TdP using ECG data. Second, we developed a new interpretability algorithm named Evocclusion, that enables a better understanding of the neural network's decision process. This algorithm aims to provide human readable insights into the model's predictions, leading to increased trust among clinicians and specialists. Third, we present two main frameworks developed to improve ECG analysis and the interpretability method. A crucial aspect of ECG analysis is signal quality. Therefore, we propose a new method using a denoising autoencoder to significantly remove noise from the ECG data and partially recover the waveform from alterations. This technique improves the reliability of the input data for subsequent analysis and ensures that the neural networks have access to high quality information. We also developed neural networks to segment the ECG and extract beats, P and T waves, and QRS complexes. These segmentation results enable a deeper understanding of the ECG components and facilitate further analysis. Additionally, we provide a method to assess a quality score vector of the ECG, enabling us to focus on parts of the signal that have a good quality score. This approach ensures that the most reliable information is used for analysis and clinicians which reduces the risk of false positives and negatives. This research seeks to enhance trust in artificial intelligence, leading to better automation of complex tasks in medicine and beyond, ultimately improving patient outcomes.

Keywords: Electrocardiogram · XAI · interpretability · TdP · denoising · segmentation · neural network · deep learning

Résumé

L'électrocardiogramme (ECG) est un outil non invasif permettant d'évaluer l'activité électrique du cœur. Ils sont largement utilisés dans la détection d'anomalies cardiaques. Les algorithmes d'apprentissage profond permettent la détection automatique de schémas complexes dans les données ECG, ce qui offre un potentiel important pour l'amélioration du diagnostic médical. Toutefois, leur adoption est freinée par un faible niveau de confiance des cliniciens et un besoin massif de données pour entrainer les modèles. L'intelligence artificielle, en particulier l'apprentissage profond *(deep learning)*, permet d'explorer des représentations hiérarchiques de données complexes, ce qui permet de mieux comprendre les interactions internes. Néanmoins, l'interprétabilité des modèles est cruciale pour gagner la confiance des spécialistes et permettre une utilisation générale. Ces travaux de thèse, réalisés en étroite collaboration avec des spécialistes en cardiologie, visent à développer un nouvel algorithme d'interprétabilité pour les réseaux de neurones appliqués aux données ECG. Notre étude se concentre sur une pathologie cardiaque spécifique, la Torsades de pointes (TdP). La TdP est ube arythmie mortelle associée à divers facteurs, notamment médicamenteux et/ou des mutations congénitales. Une prédiction précise de ce risque peut améliorer les soins aux patients et potentiellement sauver des vies. Nous avons commencé par concevoir un réseau de neuronnes pour prédire le risque de TdP à l'aide de données ECG. Ensuite, nous avons développé un nouvel algorithme d'interprétabilité baptisé Evocclusion, qui permet de mieux comprendre le processus de décision du réseau de neurones. Cet algorithme vise à fournir des informations lisibles par l'homme sur les prédictions du modèle, afin d'accroître la confiance des cliniciens et des spécialistes. Enfin, nous présentons deux autres méthodes développées pour améliorer l'analyse de l'ECG et la méthode d'interprétabilité. La qualité du signal est un aspect crucial dans l'analyse d'ECGs. Ainsi, nous proposons une nouvelle méthode utilisant un autoencodeur de débruitage pour réduire de manière significative le bruit présent dans les données ECG et reconstruire partiellement le signal. Cette technique améliore la fiabilité des données d'entrée pour des analyses approfondies et garantit que les réseaux de neurones ont accès à des informations de haute qualité. Nous avons également développé des réseaux supplémentaires pour segmenter l'ECG et extraire les battements, les ondes P et T et complexes QRS. Cette segmentation permet une compréhension plus approfondie des composants de l'ECG et ouvre la voie à de nouvelles analyses sur des composantes spécifiques du signal. En outre, nous fournissons une méthode pour évaluer un vecteur score de qualité ECG, ce qui nous permet de nous concentrer sur les parties du signal qui ont un bon score de qualité. Cette approche garantit que les informations les plus fiables sont utilisées pour l'analyse et les cliniciens, ce qui réduit le risque de faux positifs et négatifs. Cette recherche vise à renforcer la confiance dans l'utilisation de réseau de neurones, ce qui permettra d'améliorer l'automatisation des tâches complexes en médecine et ailleurs, et, en fin, d'améliorer le traitement des patients.

Mots clés: Electrocardiogramme · XAI · interpretabilité · TdP · débruitage · segmentation · réseau de neurones · deep learning

Contents

Index of Figures

Index of Tables

Chapter 1

Introduction and Generalities

Chapter Summary

1.1 Context

1.1.1 The Electrocardiogram and heart diseases

Cardiovascular diseases are one of the leading causes of death worldwide [7], they refer to a group of disorders that affect the cardiovascular system and metabolic function. These diseases can cause a variety of health problems, including heart attack, stroke, and diabetes. Early detection and diagnosis of cardiometabolic diseases are crucial for effective treatment and prevention of complications. One important tool used for detecting these diseases is the eletrocardiogram (ECG). The ECG is a non-invasive diagnostic tool that quantifies the eletrical activity of the heart (Figure 1). It records the electrical impulses generated by the heart cells as they beats. This overall summed signal can provide valuable information about the heart's rhythm and function state.

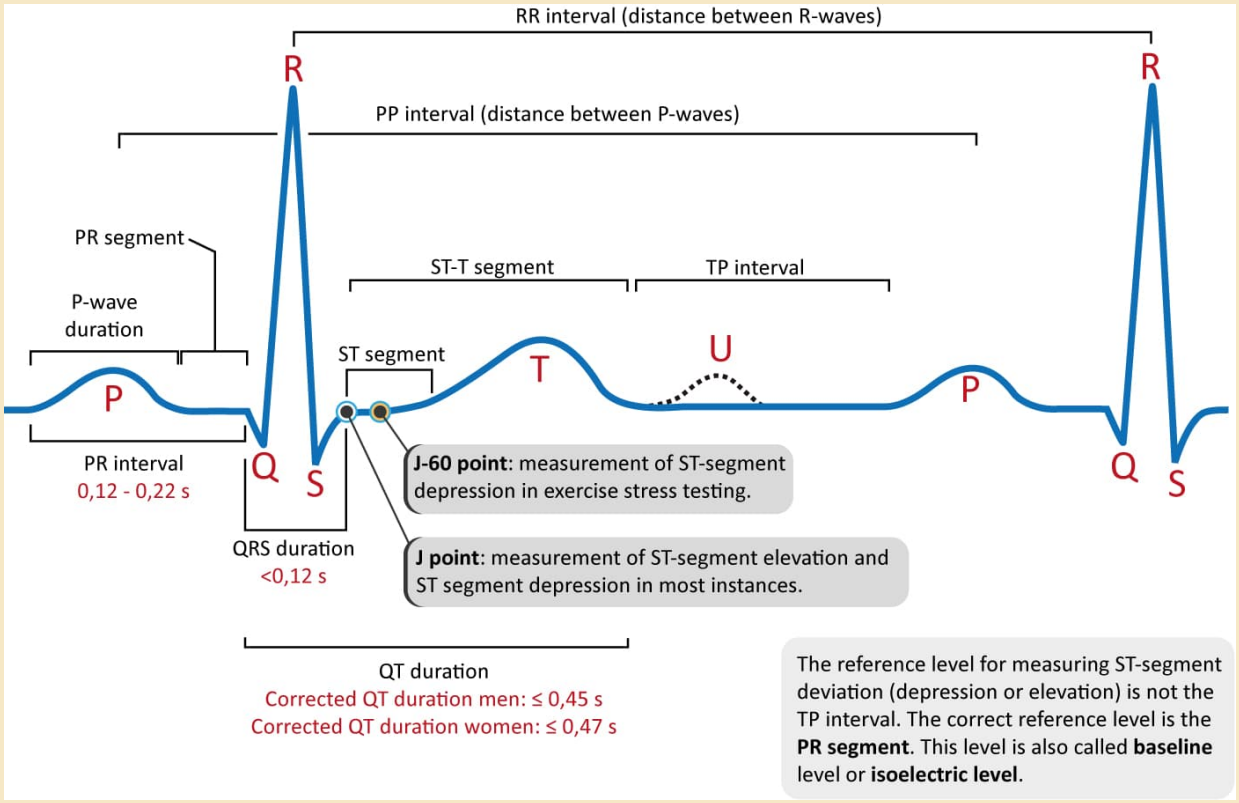


Figure 1: ***ECG Heartbeat Representation From The Clinical ECG Interpretation Book (Kusumoto, 2020).***

ECGs are recorded using electrodes, which are placed on different parts of the chest as shown in Figure 2. Each electrode records an electrical signal called lead, most ECGs are recorded on 12 leads using 10 electrodes. The leads are described as follows:

- Lead I: records the electrical signal difference between the left arm and right arm
- Lead II: records the electrical signal difference between the left leg and right arm
- Lead III: records the electrical signal difference between the left leg and left arm
- Lead aVR: captures a higher electrical signal voltage coming from the right arm
- Lead aVL: captures a higher electrical signal voltage coming from the left arm
- Lead aVF: captures a higher electrical signal voltage coming from the left leg
- Leads V1-V6: record electrical signals from the 6 electrodes placed on the chest

These leads provide a spatial and temporal representation of the heart's electrical activity. When considering spatial representation, each lead presents how the electrical impulses travel through the heart chambers (frontal, horizontal and sagittal planes) as shown in Figure 4 where each lead has a typical waveform. As for temporal representation, each lead shows how long it takes for each part of the heart to depolarize/contract and repolarize/relax during each heartbeat.

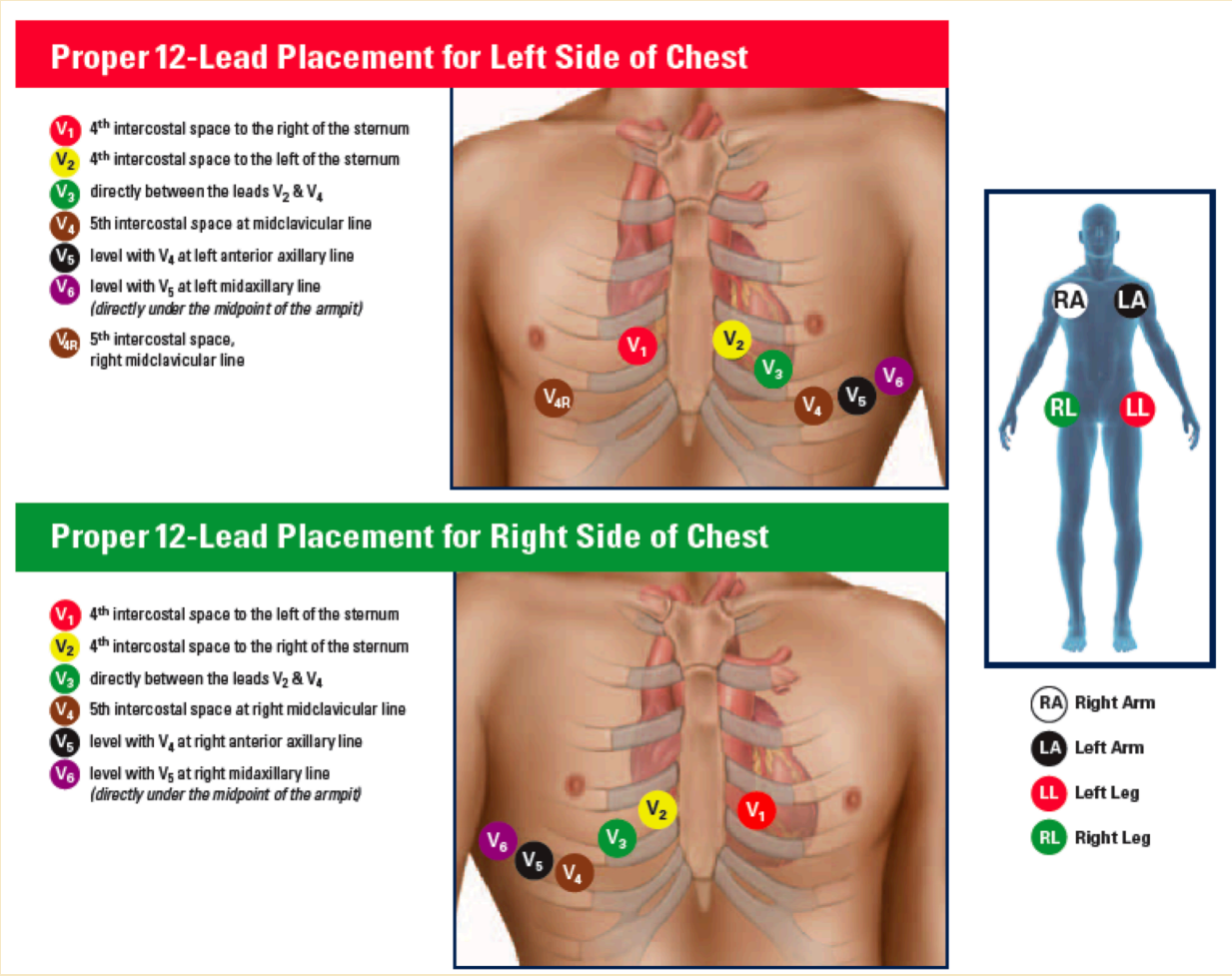
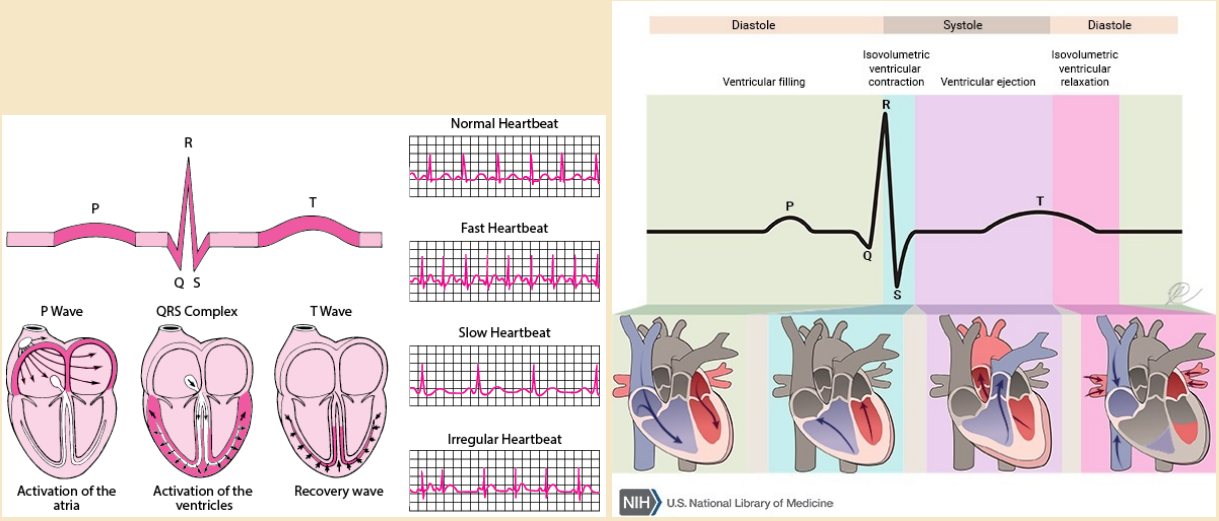


Figure 2: ***Electrodes position on the chest for recording an ECG****: 6 electrodes are placed on the chest, each one records eletrical impulses which form the ECG periodic signal. The leads records the same eletrical activity at different timesteps, the physical position of the eletrodes also provides a spatial view of the heart eletrical activity.*

ECGs are widely used by clinicians to detect cardiac abnormalities by focusing on specific regions or segments of the ECG. The most used ones as displayed in Figure 1 are the P wave, the QRS complex and the T wave. The right and left atria or upper chambers generate the first wave: P wave, followed by a flat line when the electrical impulse goes to the bottom chambers. The right and left bottom chambers or ventricles generate the next wave called the QRS complex when they contract strongly to pump the blood towards the periphery of the organ. The final wave or T wave represents the electrical recovery or return to a resting state of the ventricles [8].

Abnormalities can be spotted on the ECG and help diagnosing severe conditions such as arrhythmia. Arrhythmia is an abnormal heart rhythm that can occur due to a variety of reasons, such as heart disease, medication side effects, or hormonal imbalances. Arrhythmias can be detected on ECG by measuring the time between heartbeats and identifying irregularities in the rhythm. Early detection of arrhythmias is important because they can increase the risk of other cardiometabolic diseases, such as heart failure or stroke [8].

Some of the most common cardiometabolic diseases that can be detected from ECG are heart attacks or myocardial infarction, atrial fibrillation (AF), pulmonary embolism, chronic lung disease, hypertrophic cardiomyopathy, sick sinus syndrome, prolonged QT interval or Torsades de Pointes (TdP) [9][8][10][11][12]. Heart attacks, or myocardial infarctions (MI)[13][14][15], occur when blood flow to a portion of the heart muscle is obstructed, leading to ischemia and eventual tissue death. MIs are classified into two primary types: ST-segment elevation myocardial infarctions (STEMIs) and non-ST-segment elevation myocardial infarctions (NSTEMIs). STEMIs are characterized by the elevation



(a) **(b)**

Figure 3: ***ECG waves and their relation to heart nodes****: Illustration of the different ECG waves (P, Q, R, S, and T) and their corresponding relationship to the electrical activity in the heart's sinoatrial (SA) and atrioventricular (AV) nodes. The P wave represents atrial depolarization initiated by the SA node, while the QRS complex corresponds to ventricular depolarization and the activation of the AV node. Lastly, the T wave signifies ventricular repolarization. Understanding these ECG waveforms and their association with heart nodes is crucial for diagnosing and managing various cardiac conditions.*

of the ST segment on an ECG, which is indicative of complete blockage of a coronary artery. In contrast, NSTEMIs do not exhibit ST-segment elevation and are typically associated with partial blockage of a coronary artery. Both types of MIs can result in significant damage to the heart muscle and have potentially life-threatening consequences. The pathophysiology of MIs involves a complex interplay of factors, including atherosclerosis, plaque rupture, thrombosis, and vasoconstriction. Atherosclerosis is the progressive narrowing of coronary arteries due to the buildup of fatty deposits called plaques. Plaque rupture can expose the underlying thrombogenic material, leading to the formation of a blood clot that obstructs blood flow to the heart muscle. Additionally, vasoconstriction and vasospasm can further compromise blood flow, exacerbating ischemia and ultimately leading to myocardial necrosis.

MIs can be diagnosed through ECGs[16][17]. In fact, they can provide valuable information regarding the extent of ischemia, the affected coronary artery, and the severity of the infarction. The characteristic ECG changes associated with MIs include ST-segment elevation or depression, T-wave inversion, and the development of pathological Q-waves. In STEMIs, the ST-segment elevation observed on the ECG typically reflects transmural ischemia, which involves the entire thickness of the heart muscle. This suggests that a major coronary artery is completely blocked, necessitating rapid reperfusion therapy to restore blood flow and minimize myocardial damage. In contrast, NSTEMIs typically exhibit ST-segment depression or T-wave inversion, indicating subendocardial ischemia or injury, which affects only a portion of the heart muscle wall. The localization of an MI can also be determined based on the specific ECG leads that demonstrate abnormalities [17]. For example, anterior MIs are often associated with ST-segment elevation in leads V1 to V4, while inferior MIs show ST-segment elevation in leads II, III, and aVF. Identifying the affected coronary artery is crucial for selecting the appropriate reperfusion strategy, such as percutaneous coronary intervention (PCI) or coronary artery bypass graft (CABG) surgery.

In addition to MIs diagnosis, ECGs play a significant role in the management of myocardial infarctions. Indeed, they can be used to monitor the response to treatment, detect complications, and guide decision-making regarding further intervention. For instance, the resolution of ST-segment elevation following reperfusion therapy may indicate successful restoration of blood flow, while the persistence of ST-segment elevation may warrant additional intervention. Another critical cardiac

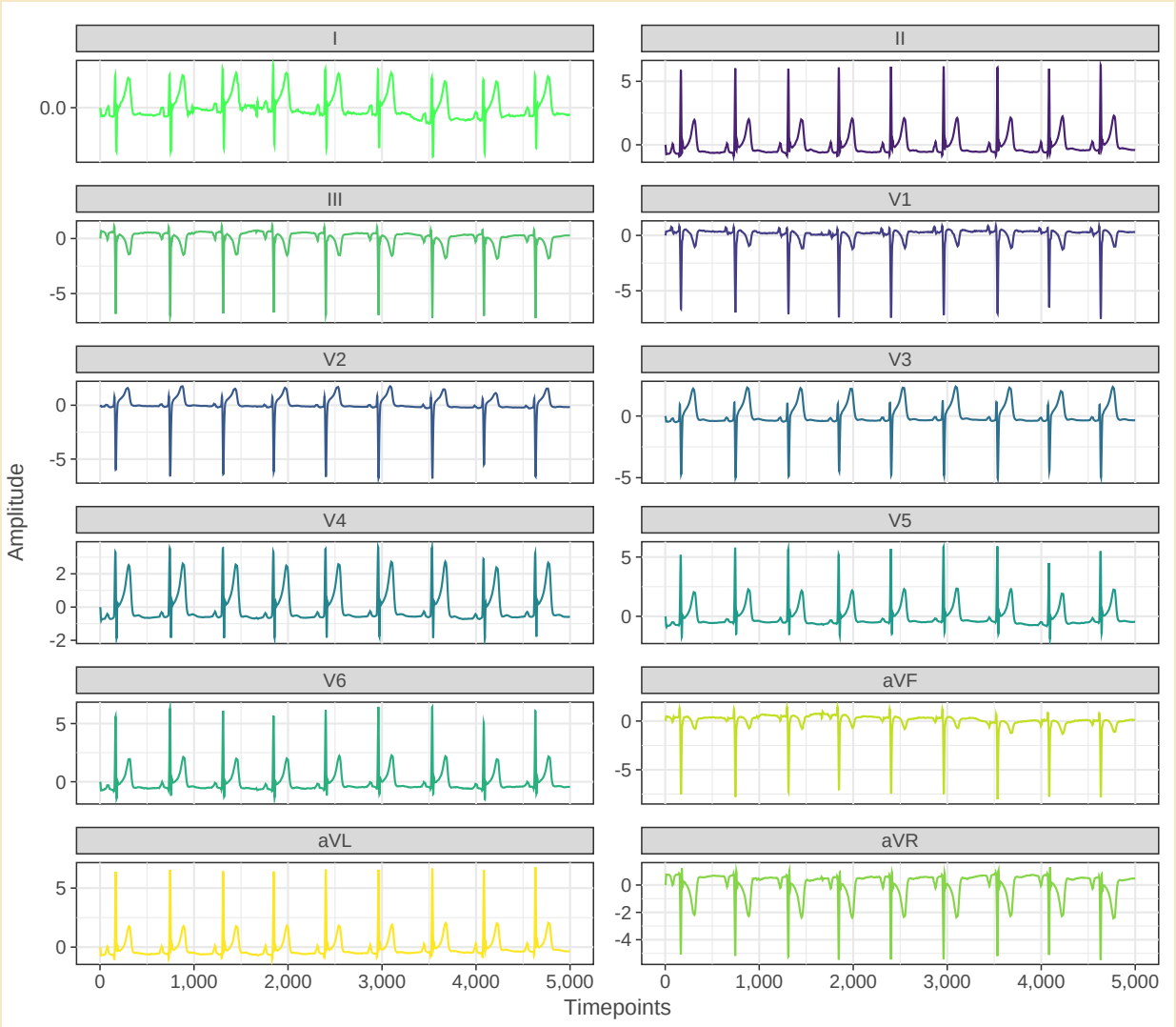


Figure 4: ***Mapping the Electrical Activity of the Heart: A 12-Lead ECG Visualization****: The ECG is obtained by placing electrodes on the chest and limbs of the patient. The ECG consists of 12 leads, each of which provides a different view of the heart's electrical activity. The leads are named using a combination of letters and numbers.*

condition that can be detected and monitored through ECG is atrial fibrillation (AF), an arrhythmia characterized by irregular and rapid heart rates due to chaotic electrical activity in the atria. Although AF is not a direct consequence of myocardial infarctions, there is a significant overlap in the risk factors for both conditions, including hypertension, diabetes, obesity, and advanced age. Furthermore, the presence of AF can complicate the management of patients with a history of MI, increasing the risk of stroke, heart failure, and other adverse outcomes.

Atrial fibrillation can be diagnosed using a 12-lead ECG [18][19], which typically reveals an irregular rhythm, absent P waves, and variable ventricular response. The ECG findings in AF are distinct from those observed in MIs; however, it is crucial to recognize that patients with myocardial infarctions may also develop atrial fibrillation as a secondary complication, particularly in the setting of acute myocardial ischemia, heart failure, or significant electrolyte imbalances. The management of AF in the context of myocardial infarctions involves addressing both the arrhythmia itself and the underlying ischemic heart disease. The primary goals of AF treatment [18] include rate control, rhythm control, and anticoagulation to prevent thromboembolic complications such as stroke. Rate control strategies, which aim to slow the ventricular response, may include the use of beta-blockers, calcium channel blockers, or digoxin. Rhythm control methods, on the other hand, seek to restore and maintain normal sinus rhythm and can involve the use of antiarrhythmic drugs, electrical cardioversion, or

catheter ablation procedures.

Another cardiac condition that warrants attention is sick sinus syndrome (SSS) [20]. SSS, also known as sinus node dysfunction, is a group of disorders characterized by abnormal functioning of the sinoatrial (SA) node, the natural pacemaker of the heart [21]. The SA node is responsible for generating regular electrical impulses that initiate the cardiac cycle and maintain a normal heart rate. In patients with SSS, the SA node fails to perform its function effectively, leading to various arrhythmia and symptoms, such as dizziness, palpitations, syncope, or fatigue. SSS encompasses a wide range of arrhythmia, including sinus bradycardia, sinus pauses or arrest, sinoatrial exit block, and tachy-brady syndrome. Sinus bradycardia is defined as a slow heart rate, typically less than 60 beats per minute (bpm), resulting from a decreased SA node firing rate. Sinus pauses or arrest occur when the SA node fails to generate an impulse, leading to a temporary cessation of atrial activity. Sinoatrial exit block refers to the impaired conduction of impulses from the SA node to the surrounding atrial tissue, while tachy-brady syndrome is characterized by alternating episodes of abnormally fast and slow heart rates, such as atrial fibrillation with slow ventricular response followed by sinus bradycardia or pauses. The pathophysiology of SSS is multi-factorial and can involve intrinsic and extrinsic factors. Intrinsic factors include age related degeneration of the SA node and surrounding atrial tissue, fibrosis, or infiltration by inflammatory or neoplastic processes. Extrinsic factors encompass the effects of medications, such as beta-blockers, calcium channel blockers, or anti-arrhythmic drugs, and autonomic nervous system influences, particularly increased vagal tone. Additionally, SSS can be associated with underlying cardiac conditions, such as ischemic heart disease, myocarditis, or congenital heart defects.

ECGs play a vital role in the diagnosis of sick sinus syndrome [22], providing valuable information regarding the underlying arrhythmia and guiding subsequent management strategies. Characteristic ECG findings in SSS include:

- Sinus bradycardia: A regular rhythm with a rate less than 60 bpm and normal P wave morphology, followed by a QRS complex.
- Sinus pause or arrest: An absence of P waves and QRS complexes for a duration that is not an exact multiple of the normal P-P interval, eventually followed by the resumption of atrial activity.
- Sinoatrial exit block: A regular rhythm with normal P wave morphology, followed by a QRS complex, but with a sudden interruption in the P-P interval, which is an exact multiple of the normal P-P interval.
- Tachy-brady syndrome: A combination of tachyarrhythmias, such as atrial fibrillation or atrial flutter, interspersed with episodes of bradycardia, sinus pauses, or arrest.

The management of SSS primarily focuses on addressing the underlying cause [22], treating associated symptoms, and preventing complications. In cases where medication-induced bradycardia is suspected, dose reduction or discontinuation of the offending drug may alleviate symptoms. However, the definitive treatment for symptomatic SSS is the implantation of a permanent pacemaker, which helps maintain an appropriate heart rate and prevents syncope or in patients with tachy-brady syndrome. Performance of the pacemaker is monitored through ECGs and help detecting any potential complications or device malfunctions. Regular ECG assessments are necessary to ensure adequate pacing, sensing, and capturing of the atrial and ventricular leads. ECG findings [23] that suggest pacemaker malfunction or failure to pace include the absence of pacing spikes, loss of atrial or ventricular capture, or inappropriate sensing of intrinsic cardiac activity.

Other cardiac electrophysiological disorders can affect different aspects of the heart's electrical activity [24]. Indeed, it is important to consider other conditions with distinct underlying mechanisms and clinical implications. One such condition is the prolonged QT interval, which, similar to SSS, affects the heart's electrical activity but in a different manner. Instead of the heart's ability to generate

and regulate its rhythm being primarily affected, as seen in SSS, the prolonged QT interval is characterized by an extended duration of ventricular depolarization and repolarization, represented by a longer QT interval on an ECG (Figure 5) [11].

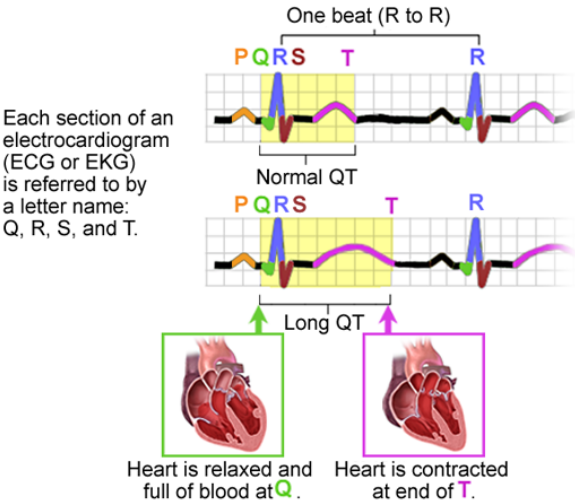


Figure 5: ***Illustration of prolonged QT interval****: The QT interval is measured from the beginning of the QRS complex to the end of the T wave, representing ventricular depolarization and repolarization, respectively. In this example, the QT interval is prolonged, indicated by the increased distance between the QRS complex and T wave.*

This condition increases the risk of life-threatening arrhythmia, such as TdP [25], and sudden cardiac death. The causes of prolonged QT interval can be congenital or acquired, with acquired causes often being more common and including factors such as certain medications, heart rate, autonomic tone, electrolyte imbalances, and underlying medical conditions. The QT interval represents the time from the onset of ventricular depolarization (the Q wave) to the end of ventricular repolarization (the T wave) and is an essential measure of the electrical activity within the heart. The normal QT interval duration varies depending on age, sex, and heart rate [25]. It can be measured using different methods, including calipers and tangent. The first one is performed by using a pair of calipers to measure the distance between the start of the QRS complex and the end of the T wave. The start of the QRS complex is typically defined as the point where the QRS complex begins to deviate from the baseline, and the end of the T wave is defined as the point where the T wave returns to the baseline. The caliper method is a simple and straightforward technique but can be prone to measurement errors due to variations in the placement of the calipers. The tangent method involves drawing a tangent line from the steepest slope of the R wave to the end of the T wave and measuring the distance from the intersection of the tangent line with the baseline to the end of the T wave. This method takes into account the slope of the T wave and can provide a more accurate measurement of the QT interval, especially in cases where the T wave is not well-defined or has multiple peaks. Once the QT interval is measured, it is typically corrected by the heart rate using the Bazett or Fredericia's formula [26], which divides the QT interval by the square root of the R-R interval (the time between successive R waves) or by the cubic root of the R-R interval for Fredericia formulae. However, the Bazzet correction method has limitations, especially at high and low heart rates, Fridericia's formula, may be more appropriate in some cases [26][27]. The corrected QT is denoted as QTc. A prolonged QTc is generally defined as >440 ms in men and >460 ms in women. There are two primary categories of prolonged QT interval: congenital and acquired.

- Congenital Prolonged QT Syndrome (LQTS) is an inherited disorder caused by mutations in genes responsible for encoding cardiac ion channels. These mutations disrupt the normal balance of ions entering and exiting the cardiomyocytes, leading to delayed ventricular repolarization. LQTS can be further classified into various subtypes, depending on the specific gene affected.

- Acquired Prolonged QT Syndrome is more common than congenital and is often due to external factors. Common causes include medications (e.g., antiarrhythmics, antipsychotics, and certain antibiotics), electrolyte imbalances (especially hypokalemia, hypomagnesemia, and hypocalcemia), myocardial ischemia, and other medical conditions (e.g., liver disease, hypothyroidism).

Prolonged QT interval increases the risk of developing potentially fatal arrhythmia, such as TdP [25][10][11], which may degenerate into ventricular fibrillation and sudden cardiac death. Patients with congenital LQTS are at a higher risk for these events, but even those with acquired causes are at an increased risk. Symptoms of prolonged QT interval may include syncope, seizures, palpitations, or sudden cardiac death. The diagnosis of a prolonged QT interval is primarily based on ECG analysis. However, thorough clinical evaluation, including a detailed family and medical history, should be performed to identify potential causes and risk factors. Management strategies of LQTS involve: (i) discontinuing QT-prolonging medications, correcting electrolyte imbalances, and treating underlying medical conditions. (ii) Risk stratification: patients should be assessed for their risk of developing life-threatening arrhythmia. High-risk patients include those with a history of syncope, a family history of SCD, or a markedly prolonged QTc. (iii) Patients should be educated on avoiding triggers such as strenuous exercise, emotional stress, and exposure to QT-prolonging medications. LQTS have several clinical implications and potential complications. One of the most severe and life-threatening arrhythmias associated with LQTS is TdP, a unique polymorphic ventricular tachycardia characterized by its distinctive appearance on an ECG: rapid heart rate, usually around 200-250 beats per minute, distinct pattern showing QRS complexes that appear to twist around the isoelectric line, hence the name "torsades de pointes," (in french) which translates to "twisting of the points." as shown in Figure 6. This twisting is due to the varying amplitudes and morphology of the QRS complexes. If left untreated, TdP can lead to ventricular fibrillation and sudden cardiac death. Prolonged QT interval is a critical risk factor for the development of TdP and is a marker of delayed ventricular repolarization.

The underlying mechanism of TdP is primarily related to abnormal ventricular repolarization. The QT interval represents the time taken for the ventricular myocardium to recover from excitation and prepare for the subsequent cycle. This period is dependent on the balance between inward and outward ionic currents, which determines the duration of the action potential. Key players in this process are the rapid (I_Kr) and slow (I_Ks) components of the delayed rectifier potassium currents, the L-type calcium current (I_CaL), and the transient outward potassium current (I_to).

Prolongation of the QT interval occurs when there is an imbalance between these currents, resulting in a prolonged action potential duration. This can be due to reduced outward potassium currents, increased inward calcium currents, or a combination of both. The repolarization abnormalities lead to the development of early after depolarizations (EADs), which are oscillations in the membrane potential occurring before complete repolarization. EADs can trigger TdP if they reach a threshold to initiate an action potential in neighboring cells, causing the twisted QRS pattern seen on the ECG [10][11].

There are numerous factors that can predispose an individual to develop TdP [11][24], including congenital and acquired causes. One of the congenital causes of Congenital long QT syndrome (LQTS) caused by genetic mutations in genes encoding ion channel sub-units or associated proteins. There are at least 17 sub-types of LQTS, with the most common being LQT1, LQT2, and LQT3, which involve mutations in the KCNQ1, KCNH2, and SCN5A genes, respectively. These mutations lead to abnormal ion channel function, resulting in a prolonged action potential duration and increased susceptibility to TdP. Acquired causes of TdP are more common and can result from a variety of factors, such as electrolyte imbalances, medications, and structural heart disease. Hypokalemia, hypomagnesemia, and hypocalcemia can all prolong the QT interval and predispose an individual to TdP. A wide range of medications, including anti-arrhythmic agents, antibiotics, anti-psychotics, and anti-depressants, can also cause acquired LQTS and TdP. Drug-induced TdP is typically a result of these medications blocking the I_Kr current, prolonging the action potential duration. Certain pathological conditions and structural heart diseases can also lead to a prolonged QT interval and TdP. Myocardial ischemia, heart failure, and cardiomyopathies can result in repolarization abnormalities and increased susceptibility to arrhythmia. Additionally, patients with a history of ventricular arrhythmia, syncope, or sudden cardiac death in the family are at an increased risk for developing TdP. The diagnosis of TdP is primarily based on the characteristic ECG findings [11]. ECG is essential in identifying risk factors and precipitating causes of TdP. As men-

tioned earlier, a prolonged QT interval is a major risk factor for TdP, and its identification can help guide further investigation into potential underlying causes. Congenital long QT syndrome (LQTS) and acquired causes, such as electrolyte imbalances or medications, can lead to a prolonged QT interval. A detailed analysis of the ECG, along with a thorough patient history and physical examination, can help distinguish between these different etiologies. In patients with suspected congenital LQTS, the ECG may demonstrate specific patterns associated with the various sub-types. For example, patients with LQT1 may exhibit broad-based T waves, while those with LQT2 may have low-amplitude, bifid T waves. Additionally, the ECG may reveal other findings suggestive of structural heart disease, such as myocardial ischemia or cardiomyopathies, which can predispose to TdP. Continuous ECG monitoring is crucial in managing patients with TdP, as it allows for the prompt detection of arrhythmia and assessment of the effectiveness of treatment strategies. In the acute setting, ECG monitoring can help guide the administration of intravenous magnesium sulfate or the use of temporary cardiac pacing or electrical cardioversion to terminate the arrhythmia. Furthermore, ECG monitoring can help assess the QT interval and the potential pro-arrhythmic effects of medications, allowing for the optimization of pharmacotherapy and minimizing the risk of recurrent TdP. In patients with a history of TdP or those at high risk, long-term ECG monitoring using Holter monitors or implantable loop recorders may be employed to detect episodes of TdP or other arrhythmia that may require further intervention. This information can also aid in determining the need for more aggressive interventions, such as implantable cardioverter-defibrillators (ICDs), particularly in patients with congenital LQTS or those with a history of recurrent TdP.

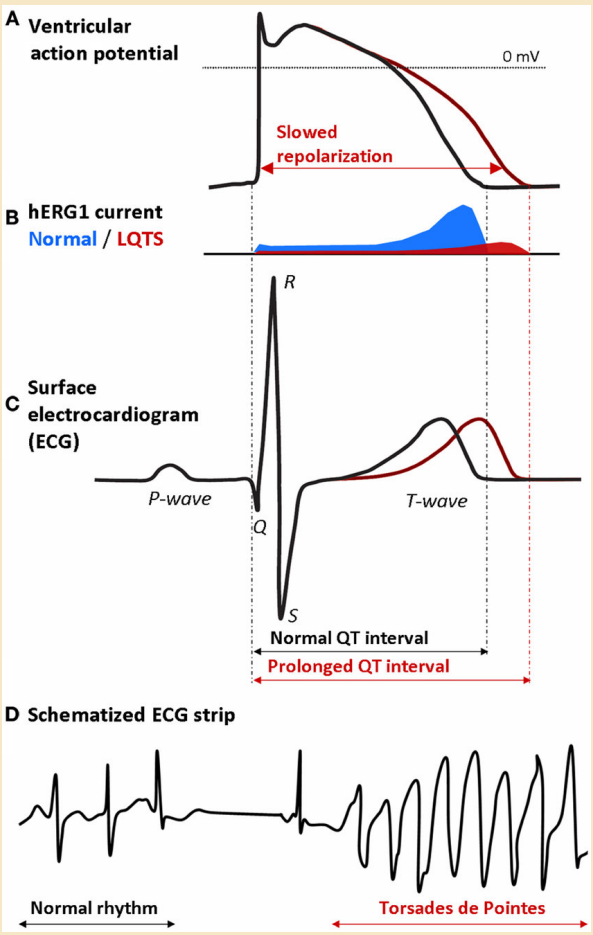


Figure 6: ***Illustration of prolonged QT interval leading to Torsade de Pointes (TdP) arrhythmia****: TdP signal is characterized by a unique twisting pattern of the QRS complex. This can be caused among others by a prolonged QT interval. TdP can lead to syncope, ventricular fibrillation, and sudden cardiac.*

1.1.2 ECG computer analyses

ECGs are crucial for detecting cardio-metabolic pathologies and the methods have evolved significantly over the past few decades. In the early days of ECG interpretation, experts manually examined the ECG waveform to detect abnormalities, such as changes in the ST segment, T wave, or QRS complex as previously stated. The interpretation relied on the knowledge and experience of the expert, making it subjective and prone to inter-observer variability. This approach was time-consuming and limited in its ability to detect subtle changes in the ECG waveform. In the 1980s and 1990s, there was a growing interest in automating the interpretation of ECGs using computer algorithms. Rule-based systems were developed to achieve this goal. These systems used a set of predefined rules to detect abnormalities in the ECG waveform. The rules were based on the knowledge and experience of experts in the field, who identified the most important features of the ECG waveform and defined rules for their interpretation (for example PR, QT, ST

intervals, amplitude of P, T waves...). The rule-based systems were designed to be faster and more consistent than manual interpretation, as they could process large volumes of ECG data quickly and consistently. However, these systems had limitations. They were designed to operate within a fixed set of rules and were not able to adapt to new data or update their rules automatically. Therefore, the rule-based systems required significant domain expertise to develop and maintain the rule sets. Also they couldn't detect more complex abnormalities that could not be captured by a set of predefined rules. For example, subtle changes in the ECG waveform such as a very light prolongation of the P wave combined with a proportional decrease of its amplitude that may be coupled with QT prolongation indicating early signs of arrhythmia may not be detected by a rule-based system. This meant that these systems were less sensitive than manual interpretation, and there was a risk of missing important information that could affect patient outcomes. Most of these systems required a preliminary diagnosis. Despite these limitations, rule-based systems played an important role in the development of ECG analysis. They provided a foundation for the development of more advanced machine learning algorithms and enabled the automation of basic ECG analysis tasks.

In the late 1990s and early 2000s, Artificial Intelligence (AI) approaches, specifically machine learning algorithms were first introduced to ECG analysis [28]. These algorithms used statistical methods to learn patterns in the ECG waveform and detect abnormalities. The algorithms were designed to identify relevant features in the ECG waveform that could be used to differentiate between normal and abnormal signals. These features included the amplitude, duration, and shape of various components of the ECG waveform, such as the QRS complex, T wave, and P wave. The advantage of machine learning algorithms over rule-based systems was their ability to adapt to new data without the need for manual intervention. Machine learning algorithms could analyze large datasets and learn patterns in the data that were not initially apparent to human experts. This approach was more flexible than rule-based systems, which relied on predefined rules that could not be easily modified or adapted to new data. However, machine learning algorithms were still limited by the quality and quantity of the training data. To train these algorithms, large datasets of labeled ECG signals were needed. The quality of the training data was critical to the accuracy of the algorithm, and errors or biases in the training data could affect the performance of the algorithm on new data. Additionally, machine learning algorithms often required significant feature engineering to extract meaningful information from the ECG waveform. This process involved selecting relevant features from the ECG waveform, preprocessing the data, and transforming the data into a format that could be analyzed by the algorithm. Despite these limitations, machine learning algorithms were a significant improvement over previous approaches to ECG analysis. They enabled automated analysis of large datasets and provided new insights into the patterns and features of ECG signals. In recent years, machine learning algorithms paved the way for the development of more advanced techniques, such as deep learning (more specifically neural networks), which eliminated the need for manual feature engineering and improved the accuracy and efficiency of ECG analysis.

These algorithms, neural networks, have emerged as the state-of-the-art approach to ECG analysis due to their ability to automatically learn complex patterns in the ECG waveform. They use a layered network of artificial neurons that can detect patterns and relationships between different aspects of the ECG waveform, which enables more accurate and efficient detection of abnormalities, the features are learned directly from the raw data. Neural networks algorithms have been shown to be effective in detecting subtle changes in the ECG waveform that were previously undetectable with manual or rule-based approaches. For example, deep learning algorithms have been used to detect early signs of heart disease and predict the risk of TdP arrhythmia from among others, prolonged QT intervals.They can also be used to identify atrial fibrillation, which can be difficult to detect with traditional approaches.

1.1.3 Concepts in Deep Learning

Neural networks are the foundation of deep learning, a sub-field of artificial intelligence that has revolutionized various domains such as computer vision, natural language processing, and reinforcement learning. In the past few years, there has been rapid and significant improvements in the field,

enabling machines to learn intricate patterns and representations from large-scale data. The core concept of neural networks is inspired by the human brain, with interconnected nodes called neurons that process and transmit information.

A neural network consists of multiple layers of interconnected neurons, with each layer transforming the input data to produce more abstract or complex representations [29]. The three main types of layers are:

- **Input Layer:** This layer receives raw data from external sources and is responsible for preprocessing and normalizing the data.
- **Hidden Layers:** These layers are between the input and output layers and consist of multiple neurons that process the input data. The hidden layers perform non-linear transformations on the input data, enabling the neural network to learn complex patterns and features.
- **Output Layer:** The final layer in the neural network produces the output, which could be a classification, a regression, or a probability distribution.

The architecture of a neural network can vary in terms of the number of layers, the number of neurons in each layer, and the connectivity pattern between layers. Three common types of neural networks are:

- **Feedforward Neural Networks:** In these networks, information flows in one direction, from the input layer through the hidden layers to the output layer. There are no loops or cycles in the connections, and each layer is connected only to the next layer.
- **Recurrent Neural Networks (RNNs):** These networks have connections that form loops, allowing them to maintain a hidden state that can capture information from previous time steps. RNNs are particularly useful for processing sequential data.
- **Transformer Neural Networks:** In 2017, *Google Brain* introduced a new type of neural network architecture calledTransformer networks. Transformers have revolutionized the field of natural language processing (NLP) for instance. These networks are designed to process sequential data such as text, speech, and time series data. They have been shown to outperform traditional recurrent neural networks (RNNs) in many NLP tasks. One of the key innovations in transformer networks is the use of an attention mechanism. This mechanism allows the network to weigh the importance of different parts of the input sequence when making predictions. For example, when translating a sentence from one language to another, the attention mechanism allows the network to focus on specific words in the source sentence that are relevant to predicting a word in the target sentence.

Artificial neurons or perceptrons, or nodes, are the fundamental building blocks of neural networks, inspired by the biological neurons found in the human brain. They receive input signals, process the information, and output a transformed signal, which is then passed to other neurons within the network. The basic structure of an artificial neuron consists of the following components:

- Inputs: The inputs, denoted as $x_1, x_2, ..., x_n$, are the values that the neuron receives from other neurons, or from external sources in the case of input neurons. Each input is associated with a weight, $w_1, w_2, ..., w_n$, which determines the strength of the connection between the input and the neuron.
- Activation Function: The activation function, f, is a nonlinear function that transforms the sum of the weighted inputs. It introduces non-linearity into the neural network, allowing it to learn complex patterns and model non-linear relationships between inputs and outputs.
- Output: The output of the neuron, y, is the result of applying the activation function to the sum of the weighted inputs.

The output of the neuron is represented as:

$$y = f(\sum(w_i * x_i) + b)$$

where:

- x_i: the input vector components
- w_i: the corresponding weight vector components
- b: the bias term
- $f()$; the activation function

Activation functions are an essential component of neural networks, they serve to transform input values and modulate neuron outputs. Their ability to capture non-linear relationships between inputs and outputs renders them indispensable in deep learning applications. These mathematical operations, utilized in conjunction with the weighted sum of inputs (neuron operation), serve as the basis for neuron activation and ultimately determine the network's ability to learn complex patterns and make accurate predictions. Activation functions are broadly categorized into two groups: linear and non-linear. Linear activation functions, such as the identity function, maintain the proportionality of the input and output. However, they have limited use due to their inability to learn complex patterns or model non-linear relationships. Conversely, non-linear activation functions are capable of modeling complex, non-linear relationships between inputs and outputs. This flexibility makes them well-suited for various applications in deep learning and neural networks. Some common activation functions are:

– **Sigmoid Activation Function** also known as the logistic function, is mathematically expressed as:

$$f(x) = \frac{1}{1 + e^{-x}}$$

This S-shaped curve maps input values to the range (0, 1), making it suitable for binary classification tasks. However, the sigmoid function is prone to the vanishing gradient problem, wherein gradients become increasingly small during backpropagation, impeding the learning process. This issue is exacerbated as the neural network grows deeper.

– **Hyperbolic Tangent (tanh)** function is similar to the sigmoid function but transforms input values into a range between -1 and 1. This results in a function with a steeper slope, allowing for faster learning. However, like the sigmoid function, tanh is also prone to the vanishing gradient problem, limiting its effectiveness in deep neural networks. It is defined as:

$$f(x) = \frac{e^{2x} - 1}{e^{2x} + 1} = \frac{sinh(x)}{cosh(x)}$$

– **Rectified Linear Unit (ReLU)** is a popular activation function that addresses the vanishing gradient problem. It is defined as

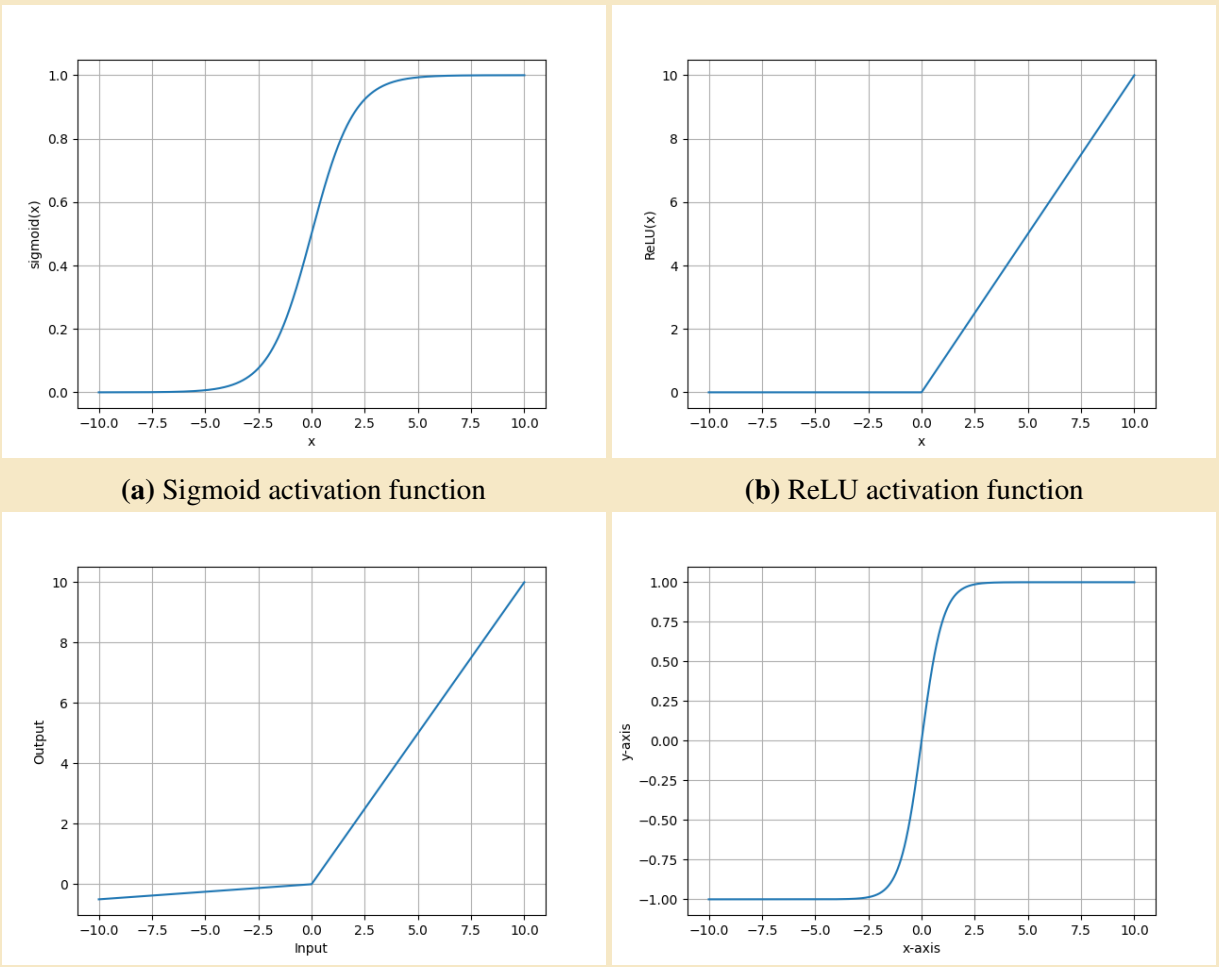
$$f(x) = max(0, x)$$

, meaning that if the input is positive, the function returns the input value, while if it is negative, the function returns zero. ReLU is computationally efficient and encourages sparse activation in neural networks, which can lead to more effective learning. However, ReLU is not without drawbacks. It is susceptible to the dying ReLU problem, where neurons may become inactive during training and never contribute to learning again.

– **Leaky ReLU and Parametric ReLU (PReLU)** were introduced to mitigate the dying ReLU problem. Leaky ReLU modifies the ReLU function by introducing a small, non-zero slope for negative input values, ensuring that neurons maintain some level of activity. PReLU takes this a step further by making the slope for negative input values a learnable parameter, allowing the network to adapt during training. Both equations are defined as:

$$f(x) = max(\alpha * x, x)$$

where α is a learnable parameter in case of PReLU.



(a) Sigmoid activation function

(b) ReLU activation function

(c) Leaky ReLU activation function

(d) Tanh activation function

Figure 7: A 2x2 grid of subfigures

Selecting the appropriate activation function depends on the specific problem and the architecture of the neural network. One must consider the properties and limitations of various activation functions while experimenting with different combinations to achieve optimal performance.

Loss Functions also known as cost functions or objective functions, are crucial components for training and optimizing neural network models. They quantify the discrepancy between the predicted label and the actual or true label, thus providing a measure of the performance of the model. Mathematically, a loss function is represented as:

$$L(y, \hat{y}) = L(y, f(x))$$

Where L is the loss function, y is the true output, $\hat{y}$ is the predicted output, and $f(x)$ is the model that maps input x to output $\hat{y}$. The primary goal of a learning algorithm is to minimize the loss function, as

this results in improved model performance. The optimization process adjusts the model parameters using techniques such as gradient descent, an optimizer algorithm, which iteratively minimizes the loss function by updating the parameters in the direction of the negative gradient. Loss functions can be broadly categorized into two classes: regression loss functions and classification loss functions. Regression loss functions are employed for continuous target variables, while classification loss functions are used for discrete target variables, such as binary or multi-class problems. Most common losses are:

Regression losses

– Mean Squared Error (MSE): is the most commonly used regression loss function. It calculates the average of the squared differences between the predicted and true values:

$$MSE(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}i)^2$$

Where n is the number of samples, y_i is the true value, and $\hat{y}_i$ is the predicted value.

The MSE is sensitive to outliers due to the squaring operation, which magnifies the effect of large errors. It is also differentiable, making it suitable for gradient-based optimization algorithms.

– Mean Absolute Error (MAE): computes the average of the absolute differences between the predicted and true values:

$$MAE(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Unlike the MSE, the MAE is less sensitive to outliers, as it does not involve squaring the differences. However, the MAE is not differentiable at zero, which can pose challenges for some optimization algorithms.

– Huber Loss: is a hybrid of the MSE and the MAE. It is less sensitive to outliers than the MSE and is differentiable:

$$Huber(y, \hat{y}, \delta) = \begin{cases} (\frac{1}{2})(y_i - \hat{y}i)^2, & \text{if } |y_i - \hat{y}i| \leq \delta \\ \delta|y_i - \hat{y}_i| - \frac{1}{2}\delta^2, & \text{otherwise} \end{cases} \tag{1.1}$$

Where δ is a user-defined parameter controlling the transition between the squared-error (first part) and the absolute-error (second part) region.

Classification losses

– Binary CrossEntropy Loss (Log Loss): also known as Log Loss, is used for binary classification problems. It measures the dissimilarity between the true probability distribution (y) and the predicted probability distribution $(\hat{y})$:

$$BinaryCrossEntropy(y, \hat{y}) = -[y\log(\hat{y}) + (1-y)log(1-\hat{y})]$$

The Binary CrossEntropy Loss penalizes incorrect predictions with high confidence, thus encouraging the model to produce well-calibrated probabilities.

– Categorical CrossEntropy Loss: is an extension of the Binary CrossEntropy Loss for multiclass classification problems. It measures the dissimilarity between the true probability distribution (y) and the predicted probability distribution (ŷ) for each class:

$$CategoricalCrossEntropy(y, \hat{y}) = -\sum_{i}^{n}(y_i * \log(\hat{y}_i))$$

Where i iterates over all classes with n the number of classes, y_i is the true probability of class i, and $\hat{y}_i$ is the predicted probability of class i.

Like the Binary CrossEntropy Loss, the Categorical CrossEntropy Loss penalizes incorrect predictions with high confidence and encourages well-calibrated probabilities.

In some cases, predefined loss functions may not be suitable for a particular problem or dataset. In such scenarios, custom loss functions can be designed to meet specific objectives. When designing a custom loss function, it is essential to consider the following aspects:

- Differentiability: The loss function should ideally be differentiable for compatibility with gradient-based optimization algorithms.
- Robustness: The loss function should be robust to noise and outliers, especially in datasets with a high degree of variability or imperfections.
- Interpretability: The loss function should have a clear interpretation, making it easier to understand the model's performance and diagnose potential issues.

Optimization algorithms and backpropagation Backpropagation is a fundamental optimization algorithm used in training artificial neural networks. It is a supervised learning algorithm that minimizes the error between predicted and actual outputs by adjusting the weights and biases of the neural network. The core concept of backpropagation is the application of the chain rule from calculus to compute gradients of the loss function with respect to each weight. The backpropagation algorithm can be broken down into the following steps:

- Forward pass
- Compute the loss
- Backward pass (compute gradients)
- Update the weights

The forward pass involves computing the output of the neural network given the input data. The output is then used to compute the loss, The backpropagation algorithm is then applied to compute the gradients of the loss with respect to the weights, and these gradients are used to update the weights in a manner that minimizes the loss. Mathematically, the backpropagation algorithm can be represented as follows. Let L be the loss function, and let W be the set of all weights in the network. The gradient of the loss function with respect to a weight w in the set W can be computed using the chain rule:

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial y} * \frac{\partial y}{\partial w}$$

where y is the output of the neuron directly connected to the weight w. The gradients are then used to update the weights:

$$w = w - \eta * \frac{\partial L}{\partial w}$$

where η is the learning rate, a hyperparameter that controls the step size of the weight updates.

Efficient gradient computing during backpropagation is made possible using auto-differentiation. It is a set of techniques to compute derivatives of functions symbolically and automatically. It is essential in modern deep learning frameworks as it simplifies the implementation of backpropagation, making it possible to compute gradients for complex models without manual differentiation. There are two primary forms of auto-differentiation: forward mode and reverse mode. In the context of deep learning, reverse mode auto-differentiation is more commonly used as it is more efficient for computing gradients of scalar-valued functions with respect to many inputs, such as the loss function in neural networks. Reverse mode auto-differentiation computes gradients in a backward pass through the computational graph of the function. It starts from the output (the loss) and proceeds to the input, computing the gradients of the loss with respect to each intermediate variable and weight in the graph using the chain rule.

As backpropagation is used to efficiently compute the gradients of the loss function with respect to the weights and biases, optimization algorithms are the methods used to update the weights and biases of the neural network based on the gradients computed by the backpropagation algorithm. Optimization algorithms define how the model parameters should be updated to minimize the loss function over time. These functions use the gradients provided by backpropagation to make informed updates to the weights and biases, ideally leading to better model performance. Some common optimization functions include Gradient Descent, Stochastic Gradient Descent (SGD), and adaptive gradient methods like AdaGrad, RMSprop, and Adam. These optimization algorithms differ in how they update the model parameters based on the computed gradients, and each has its advantages and disadvantages in terms of convergence speed, stability, and computational complexity.

– Gradient Descent (GD): is a first-order optimization algorithm that aims to find the minimum of a function by iteratively moving in the direction of the steepest decrease in the function's value. In the context of deep learning, the function to be minimized is the loss function. The update rule for Gradient Descent is:

$$w = w - \eta * \nabla L(w)$$

where w is the weight, η is the learning rate, and $\nabla L(w)$ is the gradient of the loss function with respect to the weight w.

– Stochastic Gradient Descent (SGD): is a variant of Gradient Descent that computes the gradient using a randomly selected subset of the dataset (also called a minibatch) at each iteration instead of the entire dataset. This random sampling introduces noise into the optimization process, which can help the algorithm escape local minima and converge faster. The update rule for Stochastic Gradient Descent is:

$$w = w - \eta * \nabla L(w, X)$$

where X is a randomly selected subset of the dataset.

– Momentum: is a technique used to improve the convergence of optimization algorithms like SGD by incorporating a velocity term, which is a moving average of the gradients, into the update rule. This results in smoother and faster convergence as the velocity term helps the algorithm to overcome local minima and saddle points. The update rule for SGD with momentum is:

$$v = \beta * v + \eta * \nabla L(w, X)$$

$$w = w - v$$

where v is the velocity, and β is the momentum coefficient (typically between 0.5 and 0.9).

Adaptive Gradient Methods Adaptive gradient methods are a family of optimization algorithms that adapt the learning rate for each weight individually, based on the history of gradients. This can help to speed up convergence and improve the stability of the optimization process. Some popular adaptive gradient methods include AdaGrad, RMSprop, and Adam.

– Adagrad[30]: is an adaptive learning rate method that adjusts the learning rate for each parameter based on the historical gradients. The primary motivation behind Adagrad is to improve the convergence rate by adapting the learning rate for each parameter independently. The update rule for Adagrad is as follows:

$$w_{t+1} = w_t - \eta * (G_t + \epsilon)^{-1/2} * g_t$$

Here, w_t is the weight at time step t, η is the global learning rate, G_t is the sum of the squared gradients up to time step t,ϵ is a small constant to prevent division by zero (usually set to 1e-8), and g_t is the gradient at time step t. Adagrad accumulates the squared gradients in a diagonal matrix G, which is used to scale the learning rate element-wise. The primary advantage of Adagrad is that it can handle sparse data efficiently, as it will adapt the learning rates for the infrequently updated parameters more aggressively.

However, Adagrad has a significant drawback: the accumulation of squared gradients in the denominator can cause the learning rate to become too small, which slows down the learning process or stops it altogether.

– RMSprop (Root Mean Square Propagation)[31]: was proposed as a modification to Adagrad to address the diminishing learning rate issue. RMSprop introduces an exponential decay factor (γ) to maintain a moving average of the squared gradients instead of accumulating them. The update rule for RMSprop is:

$$E[g^2]t = \gamma * E[g^2](t-1) + (1-\gamma) * g_t^2$$
$$w_{t+1} = w_t - \eta * (E[g^2]_t + \epsilon)^{-1/2} * g_t$$

Here, $E[g^2]_t$ is the moving average of the squared gradients at time step t, and γ is the decay factor (typically set to 0.9).

By using the moving average of squared gradients, RMSprop prevents the learning rate from becoming too small, thus avoiding the issue faced by Adagrad.

– Adam (Adaptive Moment Estimation)[32]: combines the ideas from both Adagrad and RMSprop, along with a momentum term. The algorithm computes the first moment (mean) and the second moment (uncentered variance) of the gradients, and it employs bias correction to account for the initialization of the moments. The update rule for Adam is:

$$m_t = \beta_1 * m_{t-1} + (1-\beta_1) * g_t$$
$$v_t = \beta_2 * v_{t-1} + (1-\beta_2) * g_t^2$$
$$\hat{m}_t = m_t/(1-\beta_1^t)$$
$$\hat{v}_t = v_t/(1-\beta_2^t)$$
$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} * \hat{m}_t$$

Here, m_t and v_t are the first and second moment estimates, respectively, β_1 and β_2 are the decay rates for the first and second moments (typically set to 0.9 and 0.999, respectively), and $\hat{m}_t$ and $\hat{v}_t$ are the bias corrected first and second moment estimates. Adam has several advantages over other adaptive gradient optimization algorithms. First, it combines the benefits of both adaptive learning rate methods (Adagrad and RMSprop) and momentum-based optimization. Second,

the algorithm is computationally efficient and requires minimal memory. Finally, Adam's default hyperparameters often work well in practice, making it suitable for a wide range of tasks. However, Adam is not without its limitations. For instance, it has been observed that Adam can sometimes lead to poor generalization performance compared to other optimization algorithms like stochastic gradient descent (SGD) with momentum. To address this issue, modifications like AMSGrad[33] and AdaBound[34] have been proposed, which incorporate second order information and dynamic bounds on the learning rates, respectively.

Neural network architectures

Convolutional networks A convolutional neural network (CNN) is a type of neural network that uses convolutional layers to extract features from input data, such as images or 1D signals (time series, speech signals, sequences) [35][36]. Convolutional layers consist of a set of filters or kernels that slide over the input data and produce feature maps, which are matrices that represent the presence of a certain feature in a certain location. Convolutional layers can be followed by other types of layers, a typical CNN consists of the following layers:

- Input layer
- Convolution layer(s) which extracts features through different data representations
- Activation function(s)
- Pooling layer(s) which reduce the size and complexity of the feature maps
- Fully connected layer(s) which perform classification or regression tasks on the extracted features
- Output layer which produces the final decision output of the network

One of the main advantages of CNNs is that they can learn features automatically from data, without requiring manual feature engineering. This makes them suitable for complex and high-dimensional data, such as images or natural language. Another advantage of CNNs is that they are translation-equivariant, meaning that if the input data is shifted by some amount, the feature maps will also be shifted by the same amount. This property allows CNNs to handle variations in the position or orientation of objects in images.

The mathematical operation behind convolutional layers is called convolution, which is a way of combining two functions to produce a third function. In the context of CNNs, one function is the input data (such as an image), and the other function is the filter or kernel (such as a 3x3 matrix). The convolution operation involves sliding the filter over the input data and multiplying each element of the filter with the corresponding element of the input data, and then summing up the results. The output of this operation is a single value, which is stored in a feature map. The convolution operation can be repeated with different filters to produce different feature maps. The convolution operation can be expressed mathematically as follows:

$$f(x,y) * g(x,y) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m,n) g(x-m, y-n)$$

Where $f(x,y)$ is the input data, $g(x,y)$ is the filter or kernel, and $*$ denotes convolution. The output of this operation is a function $h(x,y)$, which is the feature map [36].

The convolution operation can also be generalized to multiple dimensions, such as 3D for color images or 4D for video data. In this case, the input data and the filter have more than two dimensions, and the convolution operation involves multiplying and summing over all dimensions except

for one. For example, for a 3D input data with dimensions (H, W, D) and a 3D filter with dimensions (F_H, F_W, F_D), where H is height, W is width, and D is depth (or channels), the convolution operation can be expressed as follows:

$$f(i,j,k) * g(i,j,k) = \sum_{m=0}^{F_H-1} \sum_{n=0}^{F_W-1} \sum_{o=0}^{F_D-1} f(i+m, j+n, k+o) g(m,n,o)$$

Where $f(i,j,k)$ is the input data, $g(i,j,k)$ is the filter or kernel, and $*$ denotes convolution. The output of this operation is a function $h(i,j)$, which is a 2D feature map. The convolution operation can be repeated with different filters to produce different feature maps. Figure 8 illustrates a convolution operation.

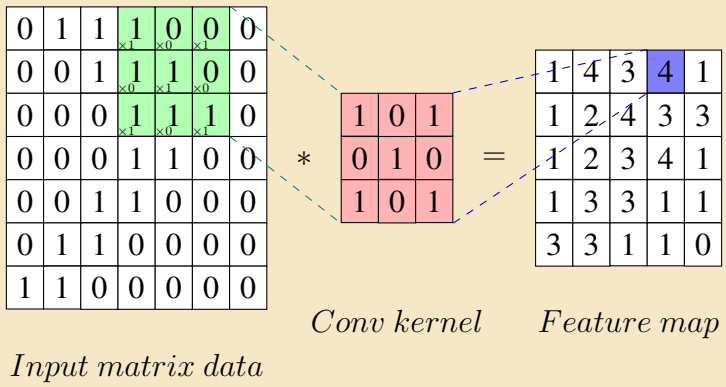


Figure 8: ***Convolution operation****: a mathematical operation that combines two functions, an input signal (or N dimensions matrix) and a kernel, to produce a modified output signal (feature map). It involves element-wise multiplication and summation of overlapping regions between the input and kernel. In image processing and deep learning, convolution is used to extract features by sliding the kernel across the input.*

Pooling layers are used to reduce the spatial dimensions of the feature maps, which in turn helps to reduce the number of parameters and computational complexity of the network. The two most common types of pooling are max pooling and averagepooling. Max-pooling returns the maximum value from a defined sub-region of the input feature map, while average pooling computes the average value in the same region. Mathematically, max-pooling can be represented as:

$$output_{ij} = max_{m,n \in P}(input_{(i+m)(j+n)})$$

Where P is the pooling region, and input and output are the input and output feature maps, respectively.

Fully connected layers are used to combine the features learned in the convolutional and pooling layers to make predictions or classifications. In a fully connected layer, every neuron is connected to every neuron in the previous and subsequent layers. If we represent the weights of the fully connected layer as a matrix W and the input feature map as a vector x, the output of the fully connected layer can be calculated as:

$$output = Wx + b$$

Where b is the bias vector.

Convolutional neural networks are composed of multiple convolutional layers, each with its own set of filters or kernels. The output feature maps of one layer serve as the input data for the next layer. By stacking multiple convolutional layers, CNNs can learn hierarchical features from data, where lower-level features (such as edges or colors) are combined to form higher-level features (such as shapes or objects). The final layer of a CNN is usually a fully-connected layer, which performs

classification or regression tasks on the extracted features. CNNs have been used in various applications, including computer vision, natural language processing, and speech recognition. Some notable applications include:

- Image classification: CNNs have shown exceptional performance in classifying images into different categories, such as recognizing objects, animals, or scenes. ImageNet, a large-scale image dataset, has been used extensively for training and evaluating image classification models, with the AlexNet architecture as a pioneering example.
- Object detection: CNNs have been used to detect and locate objects within images. Architectures such as Faster R-CNN, YOLO (You Only Look Once), and SSD (Single Shot MultiBox Detector) have been successful in detecting multiple objects and their positions in images.
- Image segmentation: CNNs can be used to segment images, which involves classifying each pixel in an image according to its associated object or class. U-Net, an architecture for biomedical image segmentation, and Mask R-CNN, an extension of Faster R-CNN, are examples of successful image segmentation models.
- Natural language processing (NLP): Although recurrent neural networks (RNNs) and transformers have been more popular for NLP tasks, CNNs have also shown success in tasks such as sentiment analysis, text classification, and named entity recognition.
- Reinforcement learning: CNNs have been combined with reinforcement learning algorithms, such as Deep Q-Network (DQN), to train agents that can learn to play games or perform other tasks directly from raw visual inputs.
- Style transfer: CNNs have been employed in neural style transfer, a technique that transfers the artistic style of one image onto the content of another image. This is achieved by leveraging the feature representations learned by CNNs during training.
- Medical image analysis: CNNs have been applied to various medical image/bio-signals analysis tasks, such as detecting diseases in medical scans (e.g., X-rays, CT scans, MRI images, ECGs, EEGs), segmenting organs or tissues, and predicting treatment outcomes.

These applications demonstrate the versatility and effectiveness of convolutional neural networks in various domains, making them an essential tool in modern artificial intelligence and machine learning.

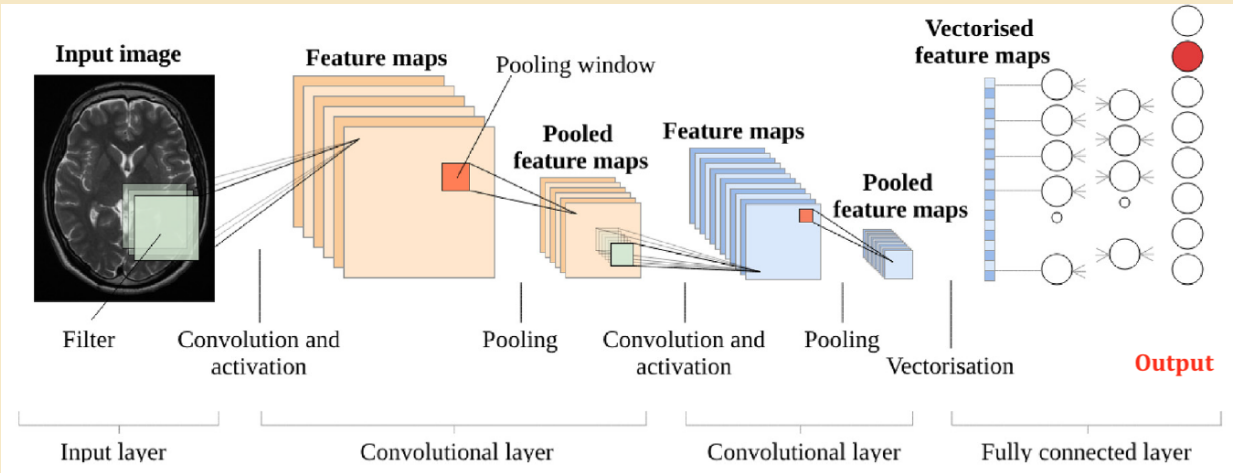


Figure 9: ***A convolution neural network architecture*** *for MRI classification. Convolution layers extract features from the input image and accross layers. Feature maps are progressively pooled then vectorized and fed to a fully conencted based classifier to output a final decision. - Extracted from [1]*

Recurrent networks Recurrent neural networks (RNNs) are another type of artificial neural networks that can process sequential data or time series data [37][38]. Unlike feedforward neural networks, which assume that inputs and outputs are independent of each other, RNNs can use their internal state (memory) to capture the temporal dependencies among the elements of a sequence. This makes them suitable for tasks such as natural language processing, speech recognition, machine translation, and image captioning. The main distinction between RNNs and traditional feedforward neural networks is the presence of feedback connections. In RNNs, the output of a neuron can be fed back to itself or other neurons in the network, effectively forming directed cycles. This architecture enables RNNs to maintain a "memory" of previous inputs, which can be used to influence future computations.

The basic structure of an RNN consists of a recurrent layer that receives an input vector x_t at each time step t and produces an output vector y_t and a hidden state vector h_t. The hidden state vector h_t is computed as a function of the previous hidden state h_{t-1} and the current input x_t, using a shared weight matrix W_{hh} connecting the previous hidden state h_{t-1} to the current hidden state h_t and a bias vector b_h. W_{xh} is the weight matrix connecting the input x_t to the actual hidden state h_t:

$$h_t = f(W_{hh} * h_{t-1} + W_{xh} * x_t + b_h)$$

The output vector y_t is computed as a function of the hidden state h_t, using another weight matrix W_{hy} and a bias vector b_y. More specifically W_{hy} is the weight matrix connecting the current hidden state h_t to the output y_t, and b_y is the output bias term:

$$y_t = g(W_{hy} * h_t + b_y)$$

where f and g are activation functions, such as sigmoid, tanh, or ReLU. The recurrent layer can be unrolled into a chain like structure that shows how the network operates over a sequence of inputs $x_1, x_2, ..., x_T$ and produces a sequence of outputs $y_1, y_2, ..., y_T$.

While the basic RNN architecture described above can capture information from previous time steps, it struggles to maintain long term dependencies in the data due to issues such as vanishing or exploding gradients during training. To address these limitations, several variants of RNNs have been proposed, including Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs).

LSTM networks introduce memory cells and gating mechanisms to better capture long-term dependencies. An LSTM cell maintains a cell state c_t and updates it through input, forget, and output gates:

- Input gate: that decides how much of the new input x_t should be added to the memory cell

$$i_t = sigmoid(W_{hi} * h_{t-1} + W_{xi} * x_t + b_i)$$

- Forget gate: that decides how much of the previous memory cell c_{t-1} should be retained

$$f_t = sigmoid(W_{hf} * h_{t-1} + W_{xf} * x_t + b_f)$$

- Output gate: that decides how much of the current memory cell c_t should be used to compute the output y_t and the hidden state h_t

$$o_t = sigmoid(W_{ho} * h_{t-1} + W_{xo} * x_t + b_o)$$

The cell state is then updated as:

$$c_t = f_t * c_{t-1} + i_t * tanh(W_{hc} * h_{t-1} + W_{xc} * x_t + b_c)$$

Finally, the hidden state and output are respectively computed as:

$$h_t = o_t * tanh(c_t)$$

$$y_t = g(W_{hy} * h_t + b_y)$$

In precedent equations W_{hi}, W_{xi}, W_{hf}, W_{xf}, W_{ho}, W_{xo}, W_{hc}, W_{xc} are weight matrices and b_i, b_f, b_o, b_c are bias vectors.

GRUs, on the other hand, simplify the LSTM architecture by merging the cell state and hidden state and using only two gates, update and reset:

- Update gate: $z_t = sigmoid(W_{xz} * x_t + W_{hz} * h_{t-1} + b_z)$
- Reset gate: $r_t = sigmoid(W_{xr} * x_t + W_{hr} * h_{t-1} + b_r)$

The hidden state is then updated as: $h_t = (1 - z_t) * h_{t-1} + z_t * tanh(W_{xh} * x_t + W_{hh} * (r_t * h_{t-1}) + b_h)$

Training RNNs involves optimizing the network's weights to minimize a loss function, which quantifies the difference between the network's predictions and the true target values. The most common optimization algorithm used for training neural networks is gradient descent, which requires calculating the gradient of the loss function with respect to the network's weights. For RNNs, this involves a process called backpropagation through time (BPTT). BPTT unfolds the RNN over time, converting it into a deep feedforward network with shared weights across time steps. The gradients are then computed using the chain rule and accumulated over time steps before updating the weights. However, BPTT can be computationally expensive due to the need to store and process information over many time steps. To address this issue, truncated backpropagation through time (TBPTT) has been proposed, which limits the number of time steps considered during backpropagation. This reduces the computational complexity at the cost of potentially less accurate gradient estimates.

Transformers networks In 2017, Vaswani et al[2], *Google Brain*, introduced a new type of neural network architecture in their paper "Attention is All You Need": Transformers. The key innovation of the transformer architecture is the self-attention mechanism, which replaces traditional recurrent neural networks (RNNs) or convolutional neural networks (CNNs) in modeling sequences. Transformers have since become a predominant architecture in natural language processing (NLP) and have shown remarkable performance on various tasks, such as machine translation, text summarizing, and sentiment analysis. The basic architecture of a transformer consists of an encoder and a decoder. The encoder takes an input sequence of vectors (such as word embeddings) and transforms it into a high dimensional representation called an encoding. The decoder takes the encoding and generates an output sequence of vectors (such as words or pixels), using a mechanism called masked self-attention to prevent it from seeing the future tokens. Both the encoder and the decoder are composed of multiple layers of sub-modules, each consisting of three main components: self-attention, feed-forward network, and layer normalization.

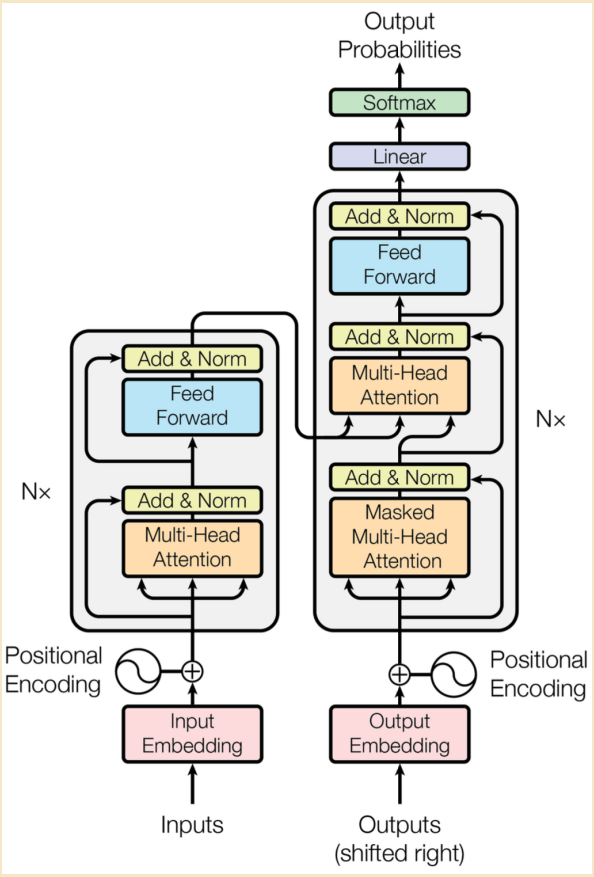


Figure 10: ***The Transformer - model architecture****: Diagram of the Transformer neural network architecture, illustrating the encoder and decoder layers with self-attention mechanisms and the encoder-decoder attention connections. Extracted from Vaswani et al, "Attention is All You Need"[2]*

Self-attention mechanism Self-attention is the core component of transformers. It is a technique that allows a neural network to learn how to focus on the most relevant parts of the input sequence for each output token. Self-attention computes a weighted sum of all the input vectors, where the weights are determined by a function that measures how similar each pair of input vectors are. The function is usually implemented by computing the dot product of two vectors, followed by a scaling factor and a softmax operation. The result is a matrix of attention scores that indicate how much each input vector contributes to each output vector. Mathematically, the self-attention mechanism can be described as follows:

1. Given a sequence of input vectors $x_1, x_2, ..., x_n$, the model first computes a set of queries (Q), keys (K), and values (V) by linearly projecting the input vectors using learned weight matrices W_Q, W_K, and W_V:

$$Q = X * W_Q$$

$$K = X * W_K$$

$$V = X * W_V$$

2. The attention scores are calculated as the dot product between queries and keys, scaled by the square root of the key dimension (d_k):

$$S = softmax(\frac{Q * K^T}{\sqrt{d_k}})$$

3. The output of the self-attention mechanism is the weighted sum of values, using the attention scores as weights:

$$Z = S * V$$

Self-attention can be further divided into three types: encoder self-attention, decoder self-attention, and encoder-decoder attention. Encoder self-attention is applied within each layer of the encoder, allowing each encoding vector to attend to all the other encoding vectors in the same layer. Decoder self-attention is applied within each layer of the decoder, allowing each decoding vector to attend to all the previous decoding vectors in the same layer. This is achieved by masking out the future tokens with zeros in the attention matrix. Encoder-decoder attention is applied between each layer of the decoder and the final layer of the encoder, allowing each decoding vector to attend to all the encoding vectors. This helps the decoder to generate outputs that are aligned with the inputs.

Multi-head attention mechanism In a multi-head attention mechanism, the model processes the input sequence multiple times with different sets of learned parameters. This allows the model to capture different aspects of the input sequence, thereby leading to more expressive representations. The multi-head attention can be formulated as:

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h) * W^O$$

where each head is a self-attention mechanism:

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

and W_i^Q, W_i^K, W_i^V, and W^O are learnable weight matrices.

Position-wise Feed-Forward networks In addition to the multi-head attention mechanism, transformers employ position-wise feed-forward networks (FFNs) to process input elements independently. The FFNs consist of two linear layers with a ReLU activation function in between, it is applied after attention layers in each sub-module of the transformer, acting as a pointwise transformation that enhances the representation power of the network:

$$FFN(x) = max(0, xW_1 + b_1) * W_2 + b_2$$

where W_1, W_2, b_1, and b_2 are learnable weight and bias parameters. The FFN is applied independently to each position in the input sequence, allowing the model to learn non-linear interactions between the input elements.

Positional encoding As transformers lack the inherent ability to consider the position of input elements, positional encoding is introduced to inject positional information into the input vectors before they are fed into the self-attention mechanism. The most common positional encoding scheme employs sine and cosine functions with different frequencies:

$$PE(pos, 2i) = sin(\frac{pos}{10000^{\frac{2i}{d}}}) \quad PE(pos, 2i+1) = cos(\frac{pos}{10000^{\frac{2i}{d}}})$$

where pos represents the position of the input element in the sequence, i is the dimension index, and d is the embedding size.

Residual Connections Residual connections are another important aspect of the transformer architecture. They help mitigate the vanishing gradient problem and facilitate the training of deep networks. In transformers, residual connections are used between the sub-layers in both the encoder and decoder. The output of each sub-layer is added to its input before being passed to the next sub-layer:

$$y = x + F(x)$$

Where x is the input, $F(x)$ is the output of the sub layer *(e.g., self-attention or position-wise feed-forward)*, and y is the combined output.

Principal architecture The transformer architecture is composed of a stack of identical layers, with each layer having two main sub layers: a multi-head self-attention mechanism and a position-wise feed-forward network. Additionally, residual connections and layer normalization are applied around each sub layer. The overall architecture can be represented as: Layer

$$\text{Layer}_i(x) = \text{LayerNorm}(x + \text{MultiHead}(x, x, x))$$

$$\text{Layer}_{i+1}(x) = \text{LayerNorm}(\text{Layer}_i(x) + \text{FFN}(\text{Layer}_i(x))$$

The original transformer architecture is divided in two independent parts: the encoder and the decoder. The encoder processes the input sequence, while the decoder generates the output sequence. However, it's critical to note that not all tasks require both an encoder and a decoder. For example, text classification tasks, such as sentiment analysis, typically use the encoder part of the Transformer. The input text is processed through the encoder, and the final hidden state is used to predict the class. Both components encoder and decoder are composed of a stack of identical layers as described above.

1. The encoder processes the input sequence by applying a series of self-attention and feed-forward layers. Each encoder layer consists of a multi-head self-attention mechanism followed by a position-wise feed-forward network, with residual connections and layer normalization applied around each sub-layer.

2. The decoder generates the output sequence by attending to both the input sequence and its own previously generated elements. The decoder has a similar architecture to the encoder but includes an additional multi-head attention mechanism that attends to the output of the encoder. The decoder layers can be represented as:

$$\text{DecorderLayer}_i(x, z) = \text{LayerNorm}(x + \text{MultiHead}(x, x, x))$$

$$\begin{aligned}\text{DecorderLayer}_{i+1}(x, z) = \text{LayerNorm}(&\text{DecorderLayer}_i(x, z) + \\ &\text{MultiHead}(\text{DecorderLayer}_i(x, z), z, z))\end{aligned} \tag{1.2}$$

$$\begin{aligned}\text{DecorderLayer}_{i+2}(x, z) = \text{LayerNorm}(&\text{DecorderLayer}_{i+1}(x, z) + \\ &\text{FFN}(\text{DecorderLayer}_{i+1}(x, z)))\end{aligned} \tag{1.3}$$

One of the main advantages of transformers over RNNs is that they can handle long-range dependencies more effectively, since they do not suffer from vanishing or exploding gradients. Another advantage is that they can leverage parallel computation more efficiently, since they do not have sequential dependencies between tokens. However, transformers also have some drawbacks, such as requiring more memory and computation resources than RNNs, and being more prone to generating repetitive or nonsensical outputs due to their autoregressive nature. Transformers have become one of the most popular and powerful neural network architectures for natural language processing and

computer vision in recent years. They have achieved state-of-the-art results on various benchmarks and tasks, such as machine translation (Vaswani et al., 2017), natural language understanding (Devlin et al., 2019), text generation (Radford et al., 2019), image classification (Dosovitskiy et al., 2020), image generation (Parmar et al., 2018), and image captioning (Zhou et al., 2020). Transformers have also inspired many variants and extensions, such as universal transformers (Dehghani et al., 2019), convolutional transformers (Bello et al., 2019), graph transformers (Battaglia et al., 2018), and vision transformers (Dosovitskiy et al., 2020), which have further pushed the boundaries of performance and applicability in various domains. Various improvements and modifications have been proposed to enhance its performance and efficiency. Some notable examples include:

1. BERT (Bidirectional Encoder Representations from Transformers) – Developed by Devlin et al., BERT is a pre-trained encoder transformer model that uses a masked language modeling task to learn bidirectional representations, enabling it to achieve state-of-the-art performance on a wide range of NLP tasks.

2. GPT (at it's current version GPT4) (Generative Pre-trained Transformer) – Introduced by OpenAI, GPT is a decoder transformer model that employs a unidirectional architecture and leverages unsupervised pre-training for various NLP tasks.

1.1.4 Deep Learning applied to electrocardiograms

Deep learning is a powerful technique for analyzing large and complex datasets, such as electrocardiograms (ECGs). ECGs are widely used for screening and diagnosing cardiovascular diseases, which are the leading cause of death worldwide. However, ECG interpretation is challenging and requires expert knowledge and experience. In the recent few years, deep learning approaches such as neural network have been shown to be very effective in automating detection and diagnosis of cardiovascular diseases through ECGs.

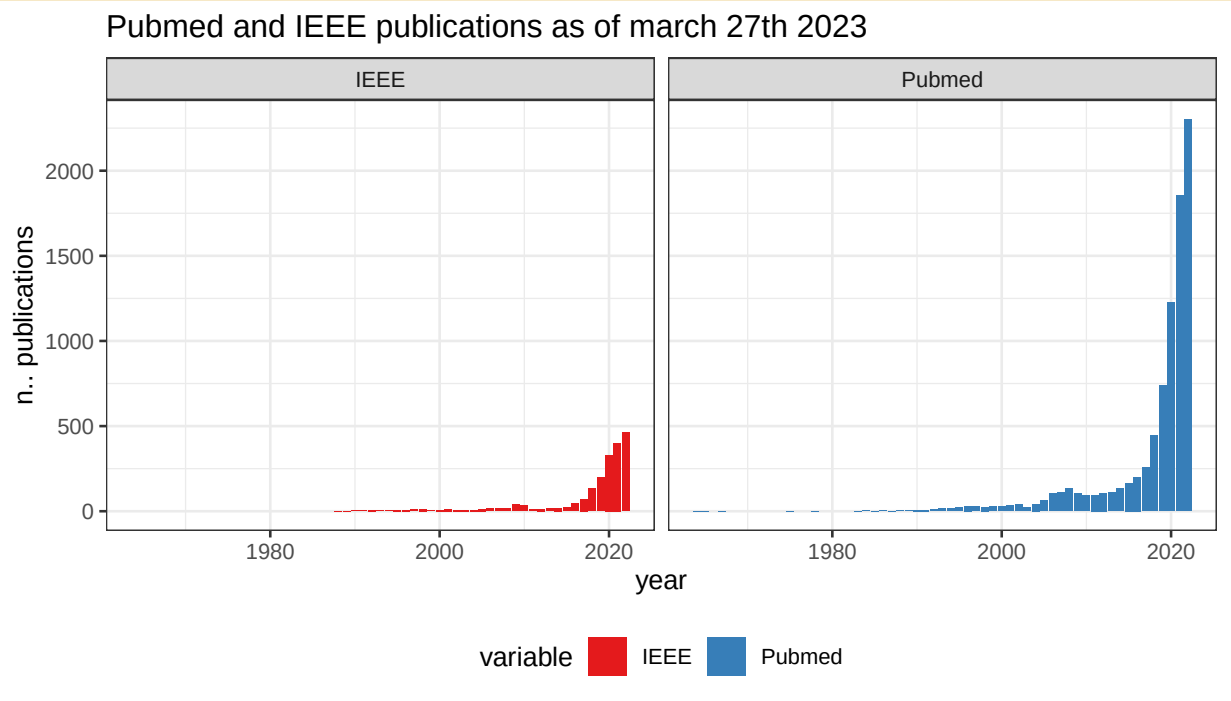


Figure 11: ***Publication trends on ECGs with Deep Learning****: using keywords ((machine OR deep AND learning) OR (artificial AND intelligence) OR (neural AND network)) AND ((cardiovascular AND disease) OR (ECG)).*

With the increasing prevalence of CVDs, automated analysis of ECGs using deep learning techniques has become an active area of research to improve diagnostic accuracy, reduce time for diagnosis, and aid clinicians in making informed decisions.

Deep learning (DL) have been very effective in performing cardiovascular risk prediction and stratification through ECGs, infect deep learning models can leverage the rich information contained in ECG signals and combine it with other clinical or demographic data to estimate the risk of adverse cardiac events, such as stroke, heart failure, or sudden cardiac death. Besides detection and diagnosis, DL have been used to preprocess ECG signals to counter noise interference, segment waveforms to focus on specific areas like P, T waves and QRS complex, to estimate the signal quality of the ECG in order to filter out poor quality signals, or to compute key concepts of the ECG such as the QT, QTcF, RR or PR distances.

ECG noise removal Most ECG recordings are prone to noise interference including baseline wander, recording noise, and artifacts, can hinder the accurate interpretation of ECGs. These disturbances may result in misdiagnosis or delayed diagnosis, leading to poor patient outcomes. In recent years, neural networks have emerged as a promising solution for denoising ECGs, with the potential to improve disease detection and diagnosis. Typical noise sources are:

- Baseline Wander: Baseline wander is a low frequency (typically below 0.5 Hz) drift in the ECG baseline, often caused by patient movement or respiration. This noise can obscure important features in the ECG, such as the ST segment, and may lead to misinterpretation.
- Recording Noise: Recording noise arises from various sources, including electrical interference from nearby equipment, powerline noise, poor electrode contact, and cable motion. This type of noise can interfere with the ECG signal and complicate its interpretation.
- Artifacts: Artifacts are unwanted signals generated by non-cardiac sources, such as muscle contractions, electromagnetic interference, or poor electrode placement. These artifacts can mimic or obscure genuine ECG features, resulting in misdiagnosis or delayed diagnosis.

Neural networks, particularly deep learning models, have shown great promise in denoising ECGs. By learning from large datasets of noisy and clean ECGs, these models can identify and remove various types of noise interference while preserving relevant features. Common neural network architectures involved in ECGs denoising are denoising autoencoders (DAE), CNNs or RNNs. Autoencoders are a type of neural network that can learn to encode and decode ECG signals. By training autoencoders on clean ECG data, they can learn to remove noise from ECGs by reconstructing the clean signal from the noisy input. CNNs can be used for denoising ECGs by learning to recognize and remove noise patterns from ECG signals. By applying convolutional filters, CNNs can isolate and suppress noise components while preserving genuine ECG features. RNNs can model temporal dependencies in ECG signals and are well suited for denoising tasks. By learning the inherent structure of clean ECGs, RNNs can predict and correct noise corrupted segments in ECG signals. Although these approaches can successfully remove noise from the signals, one critical drawback is the implicit removal of important features. Infact, by cleaning the ECG, some important features are likely to be removed thus tempering the signal information. Preserving the relevant features while removing the uncessary noise remains a challenge required to enhance disease detection and diagnosis in several ways: (i) denoised ECGs provide clearer and more accurate representations of cardiac activity, allowing for more reliable detection of cardiac abnormalities; (ii) neural network based denoising can improve the performance of automated ECG analysis algorithms, leading to better disease detection and classification; (iii) by removing noise interference, denoised ECGs can reduce the occurrence of false positives and false negatives in disease detection and diagnosis, leading to more accurate and timely clinical decision-making; (iiii) clean ECGs can improve the quality of remotely collected ECG data *(embedded devices, smart watches, portable medical devices...)*, enabling more reliable remote monitoring and management of patients with cardiovascular diseases.

ECG segmentation ECG analysis involves the identification and delineation of the various components of an ECG waveform, including the P wave, QRS complex, and T wave. Accurate ECG segmentation is essential for the diagnosis of various cardiac disorders and the evaluation of cardiac function. However, manual segmentation is time consuming, labor intensive, and prone to interobserver variability. Consequently, automated ECG segmentation methods can be developed to improve the efficiency and accuracy of this process. Among these, neural networks have emerged as promising tools for ECG segmentation due to their ability to learn complex patterns from large datasets. Deep learning techniques, such as CNNs and RNNs, have demonstrated remarkable success in ECG segmentation tasks. CNNs, in particular, are well-suited for ECG segmentation due to their ability to capture local and global features from raw ECG signals. By convoluting the input signal through multiple layers, CNNs can extract hierarchical features, enabling them to recognize and distinguish between different ECG components. The use of neural networks in ECG segmentation has several implications for clinical practice. First and foremost, automated segmentation can reduce the workload of clinicians, allowing them to focus on more complex diagnostic tasks and patient care. Moreover, by minimizing human errors and interobserver variability, neural network based methods can improve the accuracy and consistency of ECG segmentation, leading to more reliable diagnoses. Furthermore, the integration of neural networks into ECG analysis workflows can streamline the decision making process for clinicians. For example, neural network-based ECG segmentation can be coupled with other deep learning models for disease detection or risk stratification, providing comprehensive diagnostic information in a timely manner. This can enable clinicians to make better informed decisions regarding patient management and treatment. The advancements in ECG segmentation using neural networks also have implications for the development of wearable and remote monitoring devices. Accurate and automated ECG segmentation is essential for the real-time analysis of cardiac signals in ambulatory settings. By incorporating neural networks into these devices, it is possible to improve the quality of remote cardiac monitoring and facilitate timely interventions for patients at risk. Despite the promising results of neural networks in ECG segmentation, several challenges remain. One key issue is the robustness of these models to noise, artifacts, and variability in ECG signals.

ECG data genereration through Generative Adversarial Networks Generative Adversarial Networks (GANs) is a type of neural network algorithm proposed by Ian Goodfellow et al. in 2014 [39]. GANs consist of two neural networks, the generator and the discriminator, which are trained together in a competitive setting. The generator's goal is to generate data samples that are similar to the real data distribution, while the discriminator's objective is to distinguish between real and generated samples. The generator learns to produce increasingly realistic samples as the training progresses, while the discriminator becomes more skilled at discerning the real samples from the generated ones. The application of GANs to ECG data has garnered considerable interest in recent years due to the potential benefits of generating realistic ECG signals for various purposes, such as data augmentation, missing data imputation, and ECG signal analysis. The application of GANs to ECG data has garnered considerable interest in recent years due to the potential benefits of generating realistic ECG signals for various purposes, such as data augmentation, missing data imputation, and ECG signal analysis. One of the primary applications of GANs in the ECG domain is data augmentation. This technique aims to enhance the available dataset by generating additional, realistic ECG signals. Data augmentation is particularly crucial for training deep learning models, which often require large amounts of data to achieve optimal performance. Furthermore the application of GANs for generating ECGs related to rare diseases is a valuable area of research with significant potential impact on diagnosis, treatment, and understanding of these conditions. Rare diseases, by their nature, have a limited number of patients and, consequently, limited available ECG data for study. This scarcity of data hinders the development and evaluation of diagnostic algorithms and clinical decision-making tools. GANs can help address this challenge by generating realistic ECG signals that mimic the characteristics of ECGs from patients with rare diseases.

Another field of interest is missing data imputation. In real world scenarios, ECG signals can be

corrupted due to various factors, such as noise, artifacts, or lost data segments. This can negatively impact the performance of algorithms that rely on clean, continuous ECG signals for analysis. GANs can be employed to predict the missing segments of ECG signals and produce more complete, continuous data. By training the generator network to learn the underlying structure and patterns of ECG signals, it can generate realistic imputations for the missing segments, allowing for more accurate analysis and interpretation of the data. GANs can also be utilized to autocomplete ECG signals. In some cases, ECG recordings may be prematurely terminated or not long enough to obtain a comprehensive view of the cardiac activity. GANs can be trained to generate realistic extensions of the available ECG data, providing additional context and information for analysis.

Several GAN architectures have been proposed [40] to address the challenges associated with ECG data generation and manipulation. Some of the most notable ones include:

- Conditional GANs (cGANs) [40]: cGANs incorporate additional information, such as class labels or other conditioning variables, to guide the generation process. By providing the generator with specific information, such as the desired cardiac rhythm, cGANs can generate ECG signals tailored to the specified condition. This is particularly useful for generating ECG samples representing rare or underrepresented classes, which can then be used for data augmentation.

- Wasserstein GANs (WGANs) [41]: WGANs address the issue of mode collapse, which occurs when the generator produces a limited variety of samples. This is especially problematic when generating ECG signals, as the resulting data may not adequately represent the diversity of the real dataset. WGANs utilize the Wasserstein distance metric to provide a more stable training process and encourage the generator to produce a broader range of samples.

- Recurrent GANs (R-GANs) [42]: R-GANs employ RNNs such as LSTMs or GRUs in the generator and/or discriminator architecture. R-GANs are particularly suitable for generating ECG signals, as they are capable of capturing and modeling the temporal dependencies inherent in timeseries data. By leveraging the memory capabilities of RNNs, R-GANs can generate more realistic and coherent ECG signals over extended periods.

- Temporal Convolutional GANs (TC-GANs) [43]: TC-GANs incorporate temporal convolutional layers in both the generator and discriminator networks, allowing for the efficient modeling of timeseries data. These layers can capture local and global temporal patterns in ECG signals, resulting in more accurate and realistic ECG generation. TC-GANs have been shown to perform well in generating ECG signals with specific morphological and temporal characteristics.

The evaluation of the quality and realism of GAN generated ECG signals is a crucial aspect of assessing the performance of the models. Several quantitative and qualitative metrics have been proposed for this purpose, including:

1. Fidelity and diversity: fidelity refers to the similarity between the generated samples and the real data distribution. High fidelity indicates that the generated samples closely resemble real ECG signals. Diversity, on the other hand, measures the variety of the generated samples. High diversity ensures that the generated ECG signals represent a wide range of morphologies and patterns.

   These metrics can be assessed using various techniques, such as the Fréchet Inception Distance (FID), which compares the feature distributions of real and generated samples, or the Inception Score (IS), which evaluates the quality and diversity of generated samples.

2. Clinical interpretability: Clinical interpretability measures the extent to which the generated ECG signals can be interpreted and analyzed by clinicians or automated algorithms. This can be evaluated by assessing the presence of relevant ECG features, such as P waves, QRS complexes,

and T waves, as well as the accuracy of automated classification algorithms applied to the generated data.

3. Temporal coherence: Temporal coherence refers to the consistency of the generated ECG signals over time. A high degree of temporal coherence ensures that the generated samples maintain a realistic temporal structure and do not exhibit abrupt or unrealistic changes. Temporal coherence can be assessed using techniques such as autocorrelation or cross correlation analysis.

1.2 Our Problematic: the black box challenge

Recent advancements in AI, particularly in the field of deep learning with CNN, now allow us to explore and discover hierarchical representations of complex data and reveal hidden patterns/features. They also enable us to better understand the data and their internal interactions - this is where the interpretability of deep learning models become crucial. Understanding and being able to interpret the outputs of neural networks remains a critical issue, especially in the field of medicine.

Interpretability, in the context of DL, refers to the ability to understand and explain the reasoning behind the predictions made by an algorithm [44][45]. This is essential in the clinical setting, as clinicians need to trust the decision making process of AI based tools to integrate them into their daily practice. Moreover, regulatory authorities demand transparent and explainable AI to ensure the safety and effectiveness of these tools for patient care. To address the interpretability challenge, various techniques have been developed to make neural network models more understandable to clinicians. These methods can be broadly categorized into two groups: post-hoc interpretation techniques and inherently interpretable models.

A. Post-hoc Interpretation Techniques

   a) Feature Importance and Visualization: these techniques can be used to assign importance scores to individual input features. These scores help identify which parts of the ECG signal contributed most to the final prediction. This information can then be visualized in the form of activation maps, which highlight the significant regions in the ECG waveform.

   b) Attention Mechanisms: can be integrated into neural network architectures to focus on specific parts of the ECG signal during processing. These mechanisms provide insights into which segments of the ECG waveform the model considers most relevant for its prediction, offering a level of transparency that can be useful for clinicians.

B. Inherently Interpretable Models

   a) Modular Neural Networks: by designing modular neural networks, researchers can create models that explicitly represent domain knowledge. In the context of ECG analysis, this can mean incorporating known ECG features (also known as concepts such as QRS complex, P wave, T wave) into the network architecture. These models can facilitate interpretability by allowing clinicians to assess the contribution of each known feature to the final prediction.

   b) Decision Trees and Rule Extraction: decision trees and rule extraction techniques can be used to convert the knowledge learned by neural networks into human readable rules. These rules can provide a comprehensible representation of the decision making process, which clinicians can then use to evaluate and understand the model's predictions.

Interpretable neural networks have several implications for clinical practice, as they can assist in various aspects of patient care:

1. Enhanced Trust and Confidence: by providing insights into the decision making process, interpretable models can help clinicians trust and feel more confident in the AI-based tools they use. This can lead to a better acceptance of AI and ML technologies in clinical settings, ultimately improving patient care.

2. Improved Decision making: interpretable models can complement the expertise of clinicians, offering additional insights into the ECG data and helping them make more informed decisions. This can be especially beneficial in complex cases, where multiple factors need to be considered. The interpretability results will encompass not only relevant features but their interactions and how they relate to biological concepts.

3. Accelerated Model Validation and Regulatory Approval: transparent and explainable AI models are more likely to gain regulatory approval, as they allow for a thorough evaluation of their safety and effectiveness. Interpretable models can also facilitate the validation process by making it easier for researchers and clinicians to identify potential biases or errors in the model's predictions.

4. Personalized Medicine: interpretability can enable a more personalized approach to patient care by providing insights into the specific factors that contribute to a patient's ECG classification. This can help clinicians tailor treatment plans according to individual patient needs, ultimately leading to better patient outcomes. Furthermore it can helps focusing on geographical specificities. It, geographical specificities, refer to the differences in ECG patterns and characteristics observed among populations from different countries or regions. These variations can be attributed to several factors, such as genetic backgrounds, environmental influences, and lifestyle differences, which can collectively impact cardiac electrophysiology. For example, certain ethnic populations may exhibit unique ECG features, such as a higher prevalence of early repolarization patterns or Brugada phenocopies. Understanding and accounting for these geographical specificities in ECG interpretation is crucial for the accurate diagnosis and management of cardiovascular diseases across diverse populations. This consideration is particularly important when developing and deploying automated ECG analysis algorithms, which must be trained and validated on diverse datasets to ensure their performance generalizes well across different geographical and ethnic contexts.

Although neural networks can achieve high levels of accuracy, often surpassing that of experts, the level of confidence remains relatively low for specialists. This low level of confidence can be justified by the fact that despite the performance of neural networks, no tool is provided to justify the results of these algorithms and to correlate these results with the initial data. **My hypothesis is that with valid tools to interpret and explain the results of neural networks, it would be possible (i) to better assess the accuracy and robustness of the models, (ii) to better understand the data and provide humanly intelligible interpretations, and (iii) to potentially discover hidden information in the data that would advance the pathophysiology of the disease.** Thus, a high level of confidence in AI algorithms could lead to better automation of complex tasks, particularly in the field of medicine. Today, various techniques are available for interpreting neural networks. These algorithms are capable of determining regions or sets of features in the data that are responsible for the output of the AI algorithm, but they are not accurate enough. Despite the performance of these methods, we still lack precision and granularity in interpretation. In addition to identifying relevant features in the data, it is equally important to make them understandable to humans and discover the relationships between them. Most cutting edge methods focus on discovering significant features that influence classification. On the other hand, explainability, which is a relatively young field, focuses on the internal mechanics of neural networks. The goal is to understand the internal process by which the network makes a prediction. Meanwhile, interpretability algorithms seek cause and effect relationships on the input data. Efficient resource wise interpretability and explainability algorithms for neural networks are key to trust and translational applications.

1.3 Objectives

In this thesis, our primary focus is on the development of novel and improved interpretability algorithms for neural networks, specifically within the realm of ECG analysis. The research aims to enhance the understanding of deep learning models employed in the detection and diagnosis of cardiac pathologies. To accomplish this objective, we will base our work on ECG biosignals, utilizing data from the DeepECG4U project, which is funded by the ANR (Agence Nationale de la Recherche). Initially, we concentrate our efforts on the development of deep learning algorithms to identify a particular cardiac pathology through the analysis of ECG signals: the risk of developing Torsades-de-Pointes (TdP) events. TdP which is a life threatening form of ventricular arrhythmia associated with the use of specific medications, congenital mutations, and electrolyte imbalances; can potentially lead to sudden cardiac death, emphasizing the importance of accurate risk prediction. In the next phase of the research, we propose a new method for interpretability as an evolution of state-of-the-art algorithms. This method will be applied to the TdP risk prediction model, providing valuable insights into the decision making process of the neural network. We will collaborate closely with cardiology experts throughout the development and validation stages of our research, ensuring that our approaches and results are both clinically relevant and scientifically robust. The specific objectives of this thesis include:

1. Design and validate a neural network algorithm for predicting the risk of TdP arrhythmia using ECG data. This will involve the exploration of various architectures, optimization techniques, and feature extraction methods to achieve the highest possible accuracy in risk prediction.

2. Develop and implement an original interpretability algorithm that can provide meaningful insights into the neural network's decision process. This will facilitate the understanding of the model's predictions, increasing trust and acceptance by healthcare professionals. We are also going to correlate the results with clinical insights to evaluate the method.

3. Propose and develop additional tools for improved ECG signal analysis: denoising ECG signals to remove noise interference while preserving important features and segmenting the signal waveform to focus on specific regions of interest. Based on the denoising and segmentation methods we propose a new method for evaluating the quality of ECGs. These methods will be used in the interpretability framework.

Our principal goal is to contribute significantly to the growing field of research on interpretability in deep learning models, particularly within the context of ECG analysis and cardiac pathology prediction. After discussing our results, their impact, and future prospects, we present a translational application project that we initiated during the thesis. This software project enables the application of the various methods we have developed through a web platform dedicated to clinicians and researchers.

Chapter 2

Predicting the Risk of Torsades de Pointes from Electrocardiograms

Chapter Summary

Introduction

Congenital and drug-induced long-QT syndromes (cLQTS and diLQTS) are significant risk factors for torsade de pointes (TdP), a life threatening ventricular arrhythmia. Current strategies for identifying high risk drugs and individuals rely on the measurement of the QT interval corrected by the heart rate (QTc) on the electrocardiogram (ECG). However, this method has limited predictive value and is prone to errors. In recent years, the emergence of artificial intelligence (AI) and deep learning techniques has opened new possibilities for improving the prediction of TdP and diagnosis of cLQTS. In this chapter we explore the potential of convolutional neural network (CNN) models as a promising tool to enhance TdP risk prediction using ECG data and we provide an insight on salient features leading to the model predictions.

We designed a CNN to predict score from 0 to 1 of the risk of TdP occurrence. However, datasets with real cases of TdP are rare, therefore we used a surrogate approach. QTc prolongation has been shown to be associated with TdP and is currently used in clinical practice as a surrogate for evaluating the risk of TdP. Here, we propose a new approach to improve TdP risk prediction. We hypothesized that it would be possible to use cutting edge artificial intelligence models to learn the footprint of drugs, more specifically Sotalol, at the high risk of TdP in healthy volunteers [46]. Sotalol, induces biological effects on the heart (can be seen on ECGs) similar to those observed from LQTS subjects. Those effects, can then be used to mimic the genetic anomaly and thus train the network. We used either the eight leads concomitantly (LI, LII, V1-6; which we named the *multilead approach*) or each of the eight leads independently (*unilead approach*) to train a CNN model to predict **Sot+** (having

received sotalol, as a surrogate for IKr blockade) and **Sot-** classes (normal ECG before sotalol intake). Furthermore, we have proposed a methodological approach that not only specifically illustrates the drug's imprint on the ECG (fully corresponding to known pathophysiological mechanisms) but also the importance of interpretability in the acceptability of AI models by cardiologists.

2.1 Methodology

2.1.1 Datasets

Our study primarly uses **the Generepol cohort** from which patients have been administered the Sotalol drug which induces ECG-observable biological effects similar to the ones observed on LQTS patients. Notable effects include prolongation of the QT segment coupled with a reduction of T wave amplitude. In this cohort[47], 990 healty subjects have been given 80mg of sotalol following a specific protocol. Throughout the protocol, patients have been monitored with triplicate ECG recordings of 10s each (3 ECG recordings). ECGs were recorded prior to drug administration (ECG baselines or Sot-, absence of Sotalol) and after the injection up to 6 hours (sotT0, sotT1, sotT2, sotT3, sotT4, sotT5, sotT6). ECGs were recorded at sampling rates 250 and 500 Hz, we upsampled those recorded at 250 Hz to 500 Hz using cubic interpolation. ECGs contained eight independent leads (LI, LII, V1–V6), allowing for the reconstruction of 12 leads (addition of LIII, aVF, aVL, aVR). ECGs were provided in .scp or .xml files depending on recording devices (General Electric MAC5500, Marquette MAC15/MACVU, M3700 System, PageWriter Touch/Trim/XL/TC, Mortara ELI200 and Cardionics Cardioplug devices). They were parsed using Biosig software, and Python scripts. We used the HDF5 file format to store the parsed signals along with their class label (protocol time). In total we obtained 15119 ECGs of 8 leads of 10s each. Each ECG was 5000 points long as illustrated as illustrated in Figure 12.

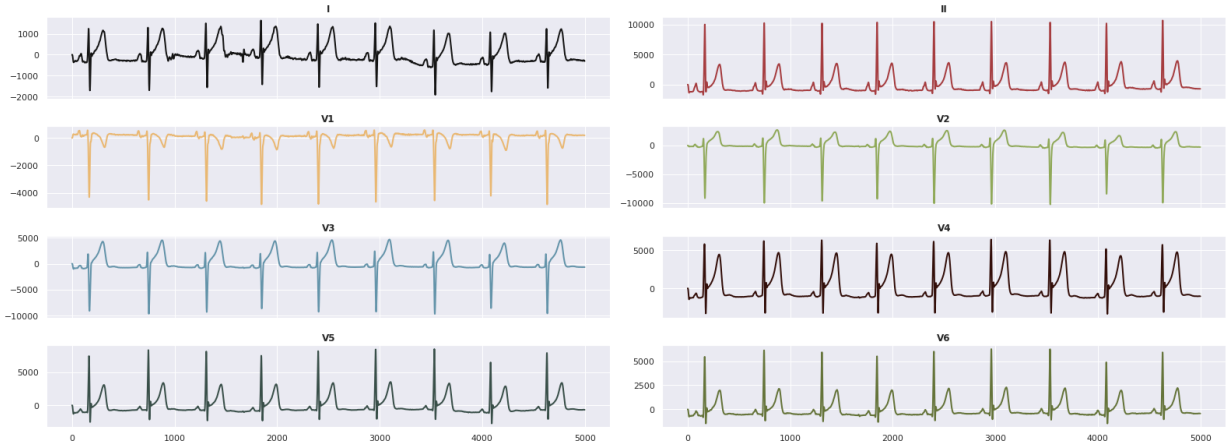


Figure 12: ***10s 8-Leads ECG from the Generepol Cohort***

Dataset partitionning We split the dataset into partitions for experimentation. The dataset was split in partitions across subjects and not signals to avoid overfitting:

- Experimentation partition: contains ECGs used for training the model and exploring hyperparameters.
  - Training: used for training the model, the loss function is computed on this partition and weights are updated
  - Validation: used to evaluate the model while training, after each epoch the model is tested on this partition without affecting or updating the weights. This gives us insight on the

performance of the model while training and allows updating the learning rate if there is no improvement (loss or accuracy) on the validation over a given number of epochs. The validation set can also be used to stop the training process if the model starts to overfit. This technique is called early stopping. In this process, if the model's performance starts to decrease (or the error starts to increase) on the validation set, we stop the training even if the performance on the training set is still improving. This is a clear sign of overfitting.

  - Evaluation: used to select best model hyper-parameters combination. It provides an unbiased evaluation of a model fit during the training phase while tuning the parameters. This allows us to select the best model and have an idea about how it's likely to perform on unseen data.

- Holdout: used to test the final model with final hyper-parameters, no decisions are taken based on this partition.

Table 1 details the number of samples per partition. For this study we only used baseline ECGs as the Sot- class and sotT1-T3 as Sot+ class. SotT4 and T5 were not used as the footprint of the drug might have been less significant compared to T3. No data-preprocessing nor filters were applied to signals except standardization.

	Experimentation (85%)			**Holdout (15%)**
	Training (75%)	Validation (10%)	Evaluation (15%)	
ECGs	6579	834	1282	1497
ECGs Sot+	3203 *(49%)*	418 *(50%)*	629 *(49%)*	763 *(48%)*
ECGs Sot-	3376 *(51%)*	416 *(50%)*	653 *(51%)*	834 *(52%)*
Patients	632	84	126	148

Table 1: Datasets partitions and number of samples/subjects per partitions.

We also used another dataset to test models performances, **the Pharmacia's cohort** was an open-label, nonrandomized study involving healthy controls (n = 39, 28 males) receiving a fixed oral sotalol sequence administered on 3 successive days: 24-h baseline without sotalol (Day 0); 160 mg in all participants at 8:00 am Day 1; and 320 mg in 21 males at 8:00 am Day 2. The study was conducted at Pharmacia's Clinical Research Unit (start–end: 2002; Kalamazoo, MI, USA).

Finally, we used a third dataset of dug-induced TdP ECGs to effectively evaluate models. **The diTdP cohort**. included 48 patients prospectively enrolled and followed at Vanderbilt. University Medical Center (start–end: 2002–19, Nashville, TN, USA). who had experienced at least one diTdP episode; acute cardiac ischaemia. at the time of the event and genetically confirmed underlying cLQTS. were exclusion criteria. All cohorts were approved by institutional review boards, and written informed consent was obtained from participants when appropriate. Recordings from these patients were reviewed by two expert cardiologists and tracings with ventricular or junctional tachycardia during the. 10-s acquisition were excluded from the analyses. In all cohorts, QTc was heart rate corrected with Fridericia's formula and details concerning the respective inter and intraobserver variability for QTc measurements in the cohorts are detailed elsewhere.

2.1.2 Classification model architecture

We trained two categories of models, one using all of the 8-leads: multilead model, and another using only lead at the time: unilead model. We obtained 8 unilead models, one for each lead and one single multilead model.

Multilead model The multilead model, (**M1:ecg_multilead**), is a linearly stacked convolutional network made of 11 blocks of convolution layers. Each block containing two successive 1D-Convolution layer with a kernel window of 3 with the same number of filters followed by and a 1D-MaxPooling layer with a pool size of 2. The number of convolutional filters for each block are: 8, 8, 8, 16, 32, 64, 128, 256, 512, 1024, and 2048, respectively. Zero padding was used for each 1D-Convolution to keep the same output dimensions. After the convolutional blocks, the extracted features are fed to a dense classifier, a linear layer of 512 hidden nodes followed by ReLU non linear activation, a dropout layer of 70% and a final linear layer of 2 hidden nodes (one for each class) followed by a softmax activation. The architecture is presented in Figure 13.

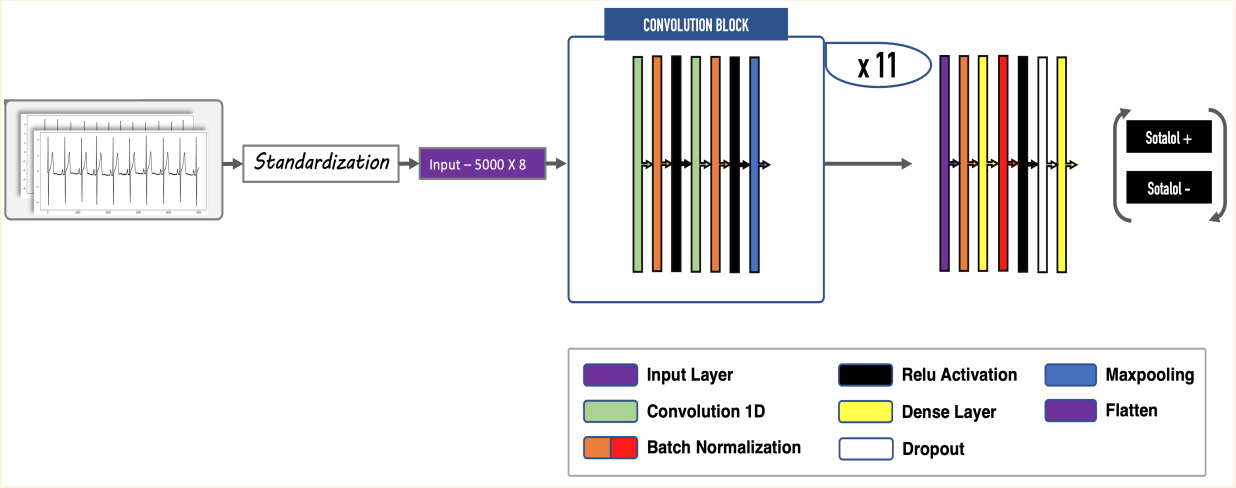


Figure 13: ***The Multilead Model Architecture***

To explore the potential effect of clinical data on the prediction of TdP risk, we also tranined another model (**M2:ecg_multilead + clin**) with the same architecture as the previous one but using ECG data and clinical data (age, sex, serum potassium).

Unilead model: DenseNet architecture We explored the capacity of any single ECG channel (lead) to provide information on the footprint of the sotalol intake. We designed eight CNN models based on the same architecture that were trained on each single channel (LI-LII, V1-6), and tested them on the same channel they were trained as well as on other leads. ECGs were not denoised nor pre-processed they were only standardized. The unilead model (**M5:ecg_unilead**) has a DenseNet-like model architecture for each lead.

DenseNet can be traced back to the innovations in the field of residual networks (ResNets [48]). ResNets were introduced to mitigate the vanishing gradient problem in deep neural networks by implementing skip connections, which allowed the gradient to flow directly through the network, bypassing several layers. This approach allowed for the training of very deep networks, leading to improved performance on various computer vision tasks. The key innovation in DenseNet is the concept of dense connectivity. Unlike traditional CNNs, which have layers connected sequentially, DenseNet connects each layer to every other layer in a feed-forward fashion within a dense block. This dense connectivity facilitates the reuse of features and encourages the learning of more diverse features, thereby improving the model's efficiency and performance.

Dense block and layer At the begining of the architecture, a first convolutional layer extracts higher features representations, the layer is then followed by a batch-normalization and a LeakyReLU activation [49]. The next step consists of eight successive densely connected blocks named *dense block*. Each DenseBlock is composed of several convolutional sub-blocks called *dense layers*. Each dense layer produces k feature maps, and these feature maps are concatenated with the feature maps of all preceding layers in the same dense block. This process results in an exponential growth of

the number of feature maps as the network becomes deeper. To control the growth of feature maps and maintain the model's computational efficiency, a hyperparameter called the growth rate (k) is introduced. The growth rate determines the number of additional feature maps learned by each layer. A dense layer is sequentially composed of a bottleneck layer, batch-normalization layer, LeakyRelU activation, convolutional layer and dropout layer. The dropout rate is a hyper-parameter and is determined before training.

Blottleneck layer To further enhance the model's efficiency, DenseNet introduces bottleneck layers within the dense blocks. These layers consist of a batch-normalization, followed by a LeakyReLU activation, a 1x1 convolution and a dropout layer. The 1x1 convolution is responsible for reducing the number of input channels to a multiple of the growth rate. This approach not only reduces the computational complexity of the subsequent convolutions but also encourages more efficient feature learning.

Transition layers and compression To manage the increasing number of feature maps and to reduce the computational complexity, DenseNet uses transition layers between dense blocks. These layers are composed of a bottleneck followed by LeakyReLU activation and an average pooling 1D layer which takes the average of points, reducing the dimensionality of the previous output. The primary function of the transition layers is to reduce the spatial dimensions and the number of feature maps produced by the preceding dense block.

A compression factor is introduced, which multiplies the number of feature maps produced by a dense block before they are passed to the transition layer. The compression factor is typically set to a value between 0 and 1, which reduces the number of feature maps and, in turn, the model's complexity.

Classifier The third step of the network is a fully connected classifier: final output of the dense convolutional blocks is flattened through a global average pooling 1D, and then fed to successive linear layers and LeakyReLU activation. All the dropout layers had a common rate of 0.2. The final output activation is a Softmax, which provides a posterior probability for each Sot class (Sot-, Sot+).

The model architecture is illustrated in Figure 14.

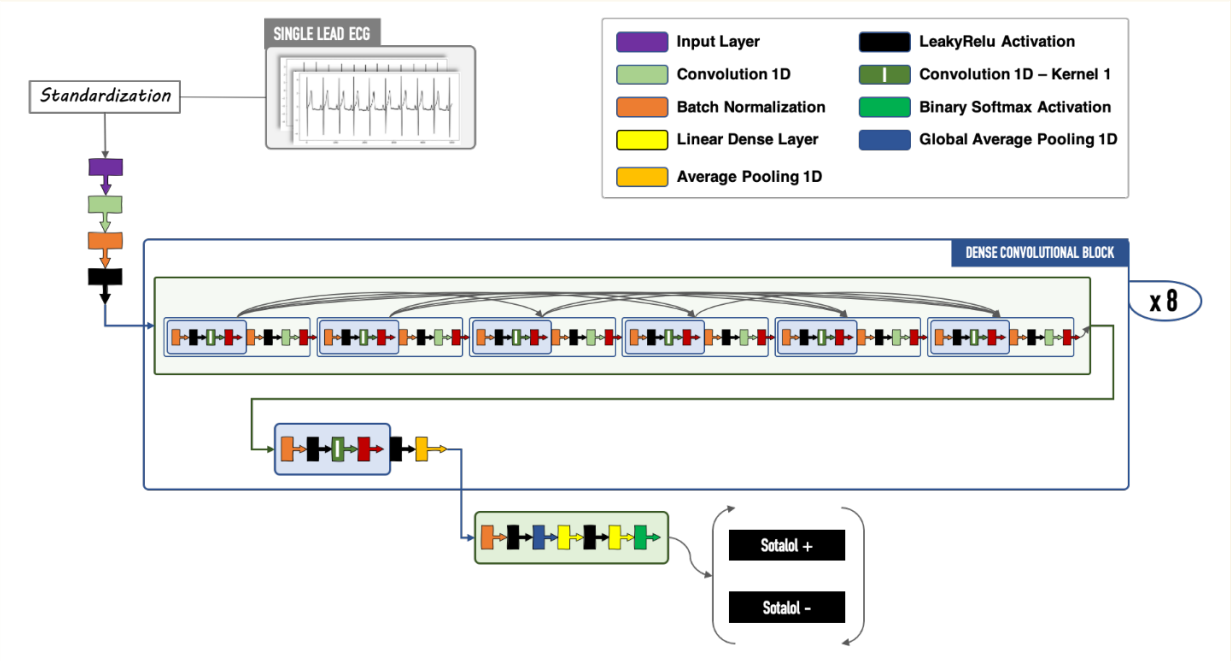


Figure 14: ***The Unilead Model Architecture***

DenseNet's dense connectivity pattern ensures that gradients can flow directly from the loss function to each layer, which mitigates the vanishing gradient problem. As a result, the network can be trained effectively, even when it is very deep. Features learned in early layers are directly passed to later layers through concatenation. This process facilitates the reuse of features, reducing the number of parameters required to learn new features in subsequent layers. Additionally, dense connectivity encourages feature collaboration between layers, allowing the network to learn a diverse set of features that are highly informative for the task at hand. This collaboration also results in improved generalization capabilities and lower risk of overfitting. The dense connectivity pattern also acts as an implicit regularizer, discouraging overfitting even in the absence of explicit regularization techniques such as dropout or weight decay. This inherent regularization property is particularly beneficial for small datasets, where overfitting is a common challenge.

2.1.3 Training and hyper-optimization

Both model categories were implemented and trained on Tensorflow 2.x.

Multilead model The multilead model was trained prior to the unilead models using 10-fold cross-validation approach. It allows the model to be optimized while minimizing the risk of overfitting. In this approach, the dataset is randomly partitioned into ten equally-sized subsets, or folds based on patient's ID thus preventing one of any patient's ECGs from being in multiple fold at the same time. Patient's ECGs isolation prevents overfitting on patients specificities. The training process then involves iterating through each fold as a validation set while using the remaining nine folds for training the model. This procedure is repeated ten times, and the model's performance is assessed by averaging the evaluation metrics across all iterations. By exposing the model to multiple training-validation splits, 10-fold cross-validation ensures a more robust assessment of its generalizability, reduces the likelihood of overfitting, and ultimately improves its predictive power on unseen data.

The loss function employed is a binary cross-entropy minimized by the Adam optimizer algorithm. The initial learning rate of the optimizer was set to 0.01. To improve the model learning process, an adaptive learning rate optimization technique was used: Reduce Learning Rate on Plateau. This strategy consists of monitoring the model's performance (loss) on the validation fold and lowering the learning rate when progress plateaus, ensuring that the model converges to an optimal solution. By incorporating this approach, we avoided oscillations or getting stuck in sharp, non optimal minima, which may lead to suboptimal results. Additionally, this method helped to stabilize the training process, facilitating smoother convergence and a more accurate model. The adaptive learning rate algorithm have a *patience* parameter which represents the number of consecutive epochs without any significant improvement in the validation loss before triggering the reduction of the learning rate. By incorporating patience, the training process becomes more resilient to temporary fluctuations in model performance, preventing premature or unnecessary adjustments to the learning rate. The *patience* parameter was set to 30 consecutive epochs. Whenever a plateau is reached the learning rate is reduced by a factor of 0.2.

Another technique to reduce overfitting was used during the training: Early Stopping. It is a regularization technique employed in the training to prevent overfittingb by monitoring the model's performance on the validation fold during training, early stopping halts the learning process when the validation loss ceases to decrease beyond a predefined threshold. The *patience* parameter, an integral component of early stopping, determines the number of consecutive epochs the model is allowed to continue training without significant improvement in validation loss before the process is terminated. By adjusting the patience parameter, we can effectively balance the tradeoff between overfitting and underfitting, ensuring that the model learns meaningful patterns from the data while maintaining its generalizability to unseen data points. In this experiment we set the the *patience* parameter to 50.

After cross-validation, the model was trained on the whole general training set and then tested on the holdout partition. Each training was performed during 200 epochs and batch size of 32. The models were trained on 4 GPUs Nvidia Tesla T4 of 16Gb each, 328 Gb of memory (RAM) and 56 CPU cores. Each training took approximately about 2 hours.

Unilead model Unlike the multilead model, the unilead model was not training using cross-validation technique, instead we performed a hyperparameter optimization process. Hyperparameter optimization is a process where the primary objective is to fine tune the configuration settings of a model to achieve optimal performance. These adjustable parameters, hyperparameters, govern various aspects of the learning algorithm, such as learning rate, network architecture, and regularization strength. By systematically exploring and refining the hyperparameter space, optimization techniques facilitate the discovery of high-performing models, leading to improved generalization and predictive capabilities. Table 2 describes the model architecture and training process hyperparameters. We also used the same adaptive learning rate algorithm as well as the early stopping technique. Adaptive learning rate had a reduction factor of 0.5 and a patience of 5 epochs, early stopping had a patience of 30 epochs. The maximum number of epochs was set to 400. From Table 2, for each combination of hyperparameter and for each lead we trained a model. The process resulted in numerous models, for each lead we chose the best model by comparing their performance on the evaluation subset, at the end of this process we obtained 8 models, one for each ECG lead. The hyperoptimization process on all leads took roughly 2 weeks of computing using 9 GPUs Nvidia Tesla T4, 656 Gb of memory (RAM) and 112 CPU cores.

At the end of the hyperoptimization process, we identified a common best hyperparameters combination for every ECG lead. 8 dense blocks each with 6 dense layers. The growth rate is set to 12 and the global convolution kernel size is set to 3. The initial convolution filters number is set to 24. The bottleneck is used, and compression rate is set to 1.0 which means no compression is used. Each model has around 3 millions trained variables.

Name	Description	Values
Model Architecture Hyperparameters		
Dense blocks	Number of dense blocks in the architecture	6, 7, 8
Dense layers	Number of dense layers in each dense block, this number is constant across all dense blocks	6, 7, 8
Dropout rate	The rate of all dropout layers	0, 0.2, 0.5
Activation function	Activation function used in all of the architecture except the for the last layer of the network which is a softmax function	ReLU, LeakyReLU
Compression rate	The compression factor used in the transition layers	1.0, 0.3
Bottleneck	Determines whether to used bottleneck or not in the architecture	True, False
Training Process Hyperparameters		
Learning rate	Initial learning rate for the Adam optimizer	0.01, 0.001
Batch size	Batch size per GPU	32, 64, 128, 256

Table 2: Unilead model explored hyperparameters

2.2 Results and Validation

The models (M1,M2,M5)'s output is a score indicating a likelihood of sotalol intake in the [0–1] range. A score of 0 predicted the absence of sotalol intake whereas 1 corresponded to the highest probability for sotalol exposure. The hyperoptimization process of the unilead model aproach M5 yielded 9 models with the same architecture but with different weights, the first 8 variants corresponded each to the 8 leads whereas the 9^{th} model was trained on the median of all 8 leads. We tested all 9 models on all leads to evaluate their perfomances on the evaluation partition in Table 3. From that table, the model trained on lead LII presented better performances on its own lead. Generelly, all models detected the footprint of TdP risk on leads eventhough they have not been trained on it. However on the holdout dataset the accuracy dropped between 72% and 89%.
The multilead model M1 obtained an accuracy of 90.8% on the same holdout dataset whereas the multilead + clinical data models M2 obtained an accuracy of 90% on the same partition. Integrating clinical data in the model did not improve performances.

For comparison with current practice, we also tested the performance of QTc. We designed two linear regression additional models: **(M3:QTcF)** trained on QTcF alone, **(M4:QTcF + clin)** trained on QTcF and with the same additional clinical as of M2. These models were designed to to discriminate on the presence/absence of sotalol intake.

		I	II	V1	V2	V3	V4	V5	V6	Med
M_I	**ACC**	92%	94%	56%	88%	89%	89%	87%	85%	92%
	LOSS	0.341	0.246	1.125	0.906	0.752	0.864	0.984	1.039	0.599
M_II	**ACC**	88%	96%	76%	79%	82%	86%	86%	86%	88%
	LOSS	0.368	0.214	0.948	0.874	0.673	0.569	0.553	0.44	0.449
M_V1	**ACC**	85%	92%	92%	89%	90%	90%	89%	89%	90%
	LOSS	0.549	0.392	0.360	0.483	0.462	0.451	0.487	0.509	0.429
M_V2	**ACC**	87%	88%	62%	92%	90%	90%	90%	90%	91%
	LOSS	0.681	0.641	2.893	0.454	0.574	0.542	0.479	0.570	0.514
M_V3	**ACC**	80%	89%	83%	91%	92%	91%	90%	89%	91%
	LOSS	0.538	0.360	0.436	0.272	0.295	0.291	0.309	0.352	0.276
M_V4	**ACC**	77%	86%	67%	91%	88%	90%	88%	87%	87%
	LOSS	0.662	0.401	0.828	0.272	0.347	0.304	0.328	0.377	0.388
M_V5	**ACC**	70%	86%	68%	91%	85%	88%	89%	86%	85%
	LOSS	1.248	0.541	0.979	0.304	0.524	0.453	0.436	0.519	0.551
M_V6	**ACC**	82%	88%	71%	87%	86%	89%	91%	90%	88%
	LOSS	0.928	0.703	1.206	0.579	0.733	0.606	0.547	0.538	0.665
M_Med	**ACC**	73%	85%	64%	88%	87%	88%	88%	88%	89%
	LOSS	0.703	0.437	0.786	0.332	0.444	0.356	0.341	0.368	0.364

Table 3: Accuracy and loss of the unilead models on all leads on the holdout partition. Each model results on its respective lead is colored in red.

Analysis Performance indicators were computed using both 10s single ECG signal analysis and by averaging risk scores from multiple recordings aquired within minutes of a same timepoint (multiple 10s recordings, i.e. the voting analysis) for a given patient and condition (Figure 15). The output provided by the models is a score ranging from 0 to 1 indicating a likelihood of being Sot+ (having ingested sotalol). To classify a patient into Sot+ or Sot- classes using the voting process, ECGs from the same patient were processed by the models and the patient was considered Sot+ (vs. Sot-) based on the mean classification score of the different 10s ECG, on which a threshold of 0.5 was applied (Sot+ if score > 0.5).

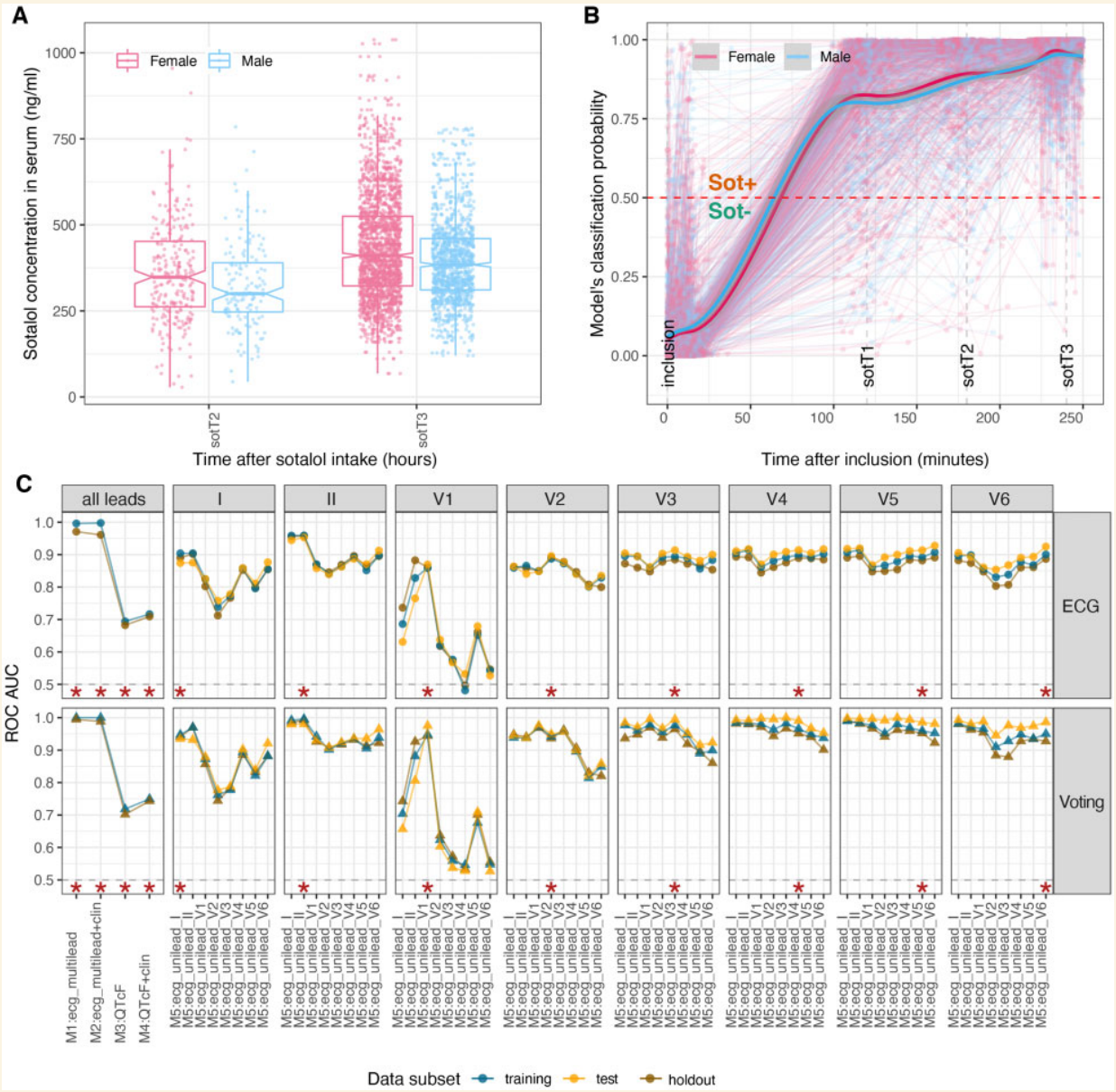


Figure 15: ***Performances of Classification Models CNN and linear regression (QT)*** *in discriminating baseline ECGs before sotalol from those after sotalol intake (SotT1, SotT2, SotT3) in Generepol. (A) Boxplots, illustrating the distribution of circulating sotalol concentration (ng/mL) in Generepol cohort two and three hours after 80mg oral sotalol intake. Data are displayed separated and coloured by gender. (B) Scatterplot illustrating the evolution of the M1:ecg_multilead classification score for the Sot+ class (y-axis) across time from inclusion (x-axis) in the Generepol cohort. All points (averaged ECGs) of a study participant are linked together as trajectories and are coloured by gender. Summarized loess (local regression) distribution of the data ± standard error is overlaid on top and grouped by gender. The red horizontal line corresponds to the Sot+/Sot- classification threshold (= 0.5). (C) Area under the receiver operating characteristic curve for the convolutional neural network multilead models (M1, M2), non-convolutional neural network standard QT-based linear regression models (M3, M4) as well as all convolutional neural network unilead M5 models in classifying each individual 10s ECG recording (top) or using a voting strategy (in triplicates of 10s ECG per study participant and time point, bottom). Multiple 10s ECGs recorded at each time point were assigned a Sot+ classification score. When the risk score was > 0.5, the electrocardiogram was classified as Sot+. With the voting approach, a mean Sot+ classification score was computed. The same threshold was applied to predict the Sot-/Sot+ class. Blue, orange, and brown colours, respectively, depict the training, test, and holdout subsets of the Generepol's cohort (see Figure 1). Each model tested on the same lead as trained is annotated by a red star. For the multilead models, all leads are used to train and test.*

Performance indicators [ROC-AUC, accuracy, precision, recall (all ranging within [0–1]), and F1 score (ranging within [0–0.5])] were evaluated for each model in 10s ECG recording individually or on the mean of multiple 10s ECG of the same participant at a given time point (voting strategy), in the training, cross-validation, and holdout sets. The mean cross-validation ROC-AUC of M1:ecg_multilead for discriminating the ECG of patients before vs. after sotalol intake was **0.948** when computed on single 10s ECG and **0.98** with the voting approach. Similarly, for M2:ecg_multilead + clin, the mean test accuracy was **0.948** (ECG) and **0.98** with voting. No difference was observed between M1 and M2. This indicated that the information contained in age, sex, and serum potassium was likely embedded in the ECG footprint captured by the CNN model. Based on the accuracy results, the multilead model presented slightly better perfomances than the unilead ones. Therefore, the M1:ecg_multilead model was deemed sufficient to be used thereafter. Its precision (voting), recall, and F1 score were very high (**0.955, 0.927, and 0.470**, respectively).

The linear regression model based on QTc alone (M3:QTcF) displayed a lower ROC-AUC of **0.695** (10s ECG) and **0.720** (voting) vs. M1:ecg_multilead (ROC-AUC: **0.948** and **0.98**, respectively; $p < 1.5e\text{-}141$). After integration of clinical data to QTc (M4:QTcF + clin), model performance increased significantly ($p < 3.3e\text{-}16$) to **0.717** (ECG) and **0.750** (voting) vs. M3:QTcF. Overall, QTc models were less effective than CNN models, even after integration of relevant clinical covariates. All four models (M1–5) displayed significantly higher ROC-AUC with the voting vs. individual 10s ECG strategy ($p < 1.2e\text{-}20$ for M1:ecg_multilead).

Unilead models performances were comparable to the multilead models. The best scores were obtained with the model trained and tested on lead LII [M5:ecg_unilead_LII; ROC-AUC = **0.958** (10s ECG) and **0.992** (voting) in the holdout set]. When this model trained on one lead was tested on the rest of the leads, it performed well, with mean holdout AUC-ROC of **0.883** (10s ECG) and **0.96** (voting). However, while the mean recall was high 0.913 (10-s ECG) and 0.963 (voting), the precision was lower 0.597 (10s ECG) and 0.605 (voting). Similar results were obtained with other unilead models, except for the one trained on V1, which did not generalize well on the other leads. Finally, we validated M1:ecg_multilead and M5:ecg_unilead_LII models (trained in the training subset of Generepol) in the Pharmacia's cohort, an independent dataset of healthy controls before and after sotalol intake. Both M1 and M5 models performed very well to discriminate sotalol intake using ECGs (ROC-AUC **0.94 0.98** depending on the models, 10s ECG vs. voting).

Evaluation on diTdP dataset We evaluated the usefulness of the M1 multilead CNN model to predict the risk of diTdP events in the third dataset (Figure 16). We quantified the association between Sot+ classification and the TdP footprint on ECG from patients having had at least one diTdP event. The TdP footprint was coded as a four class variable combining the time window since the diTdP event (<24h, 24-48h, >48h) along with the existence or absence of PVC for the >48h timeframe. We therefore fitted a mixed linear model to describe the TdP footprint phenotype as a function of QTc, intake of drugs with known risk for TdP, the Sot+ classification score and patient (ID). As expected, TdP footprint was associated with QTc (p<1.87e-10), as well as intake of drugs at risk of TdP (p<3.17e-7). Additionally, the TdP footprint was also associated with Sot+ classification score (24h from the event compared with >48h without PVC (p<0.0018; Figure 4; adjusting for QTc, TdP risky drug intake as fixed effects and patient ID as random effects). In other words, ECG closest to the TdP event had greater Sot+ score. Interestingly, this association persisted after adjustment for QTc and the intake of drugs with known risk of TdP. These observations indicate that the CNN model learned to recognize additional features in the signal of sotalol exposed ECG, which allowed discriminating diTdP risk, beyond QTc duration and intake of QT prolonging drugs.

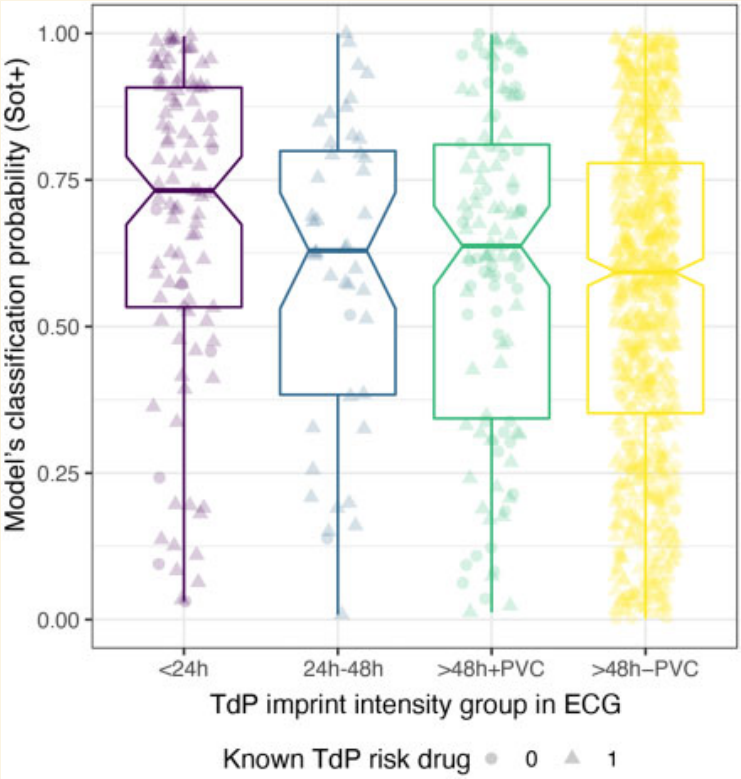


Figure 16: ***M1:ecg_multilead Sot+/Sot- model's classification score in relation to drug-induced torsade de pointes imprint intensity in ECG.*** *Boxplots indicating the distribution of convolutional neural network M1 model's classification score in patients' ECG as a function of torsade de pointes imprint intensity groups. Shape indicates the intake of drugs with known risk for torsade de pointes (triangles) vs. none (circles).*

2.3 Discovering important features of the data through state-of-the-art interpretability method: Occlusion

Methodology Machine learning research has focused on building accurate models for large data collections, often at the expense of interpretability. However, it is critical to understand why and how a decision is made, especially for healthcare providers when it comes to validation and trust. We explored the different CNN models trained to recognize sotalol intake and attempted to uncover the drug's footprint - meaning which features from the ECGs were the most useful for the classification as Sot+ and Sot- in regards to the models.

There are different approaches when considering interpretability of neural networks. Among the different algorithms for interpretability, we used the *Sliding Window Occlusion method (SWO)*, which allows us to explore the contribution of each part of the signal independently to the final prediction. Sliding Window Occlusion aims to visualize the decision making process of the neural network by selectively occluding portions of the input and observing the resulting changes in the network's output. By employing a sliding window, the technique systematically covers different regions of the input ECG to identify which areas are most critical for the model's prediction. The process of Sliding Window Occlusion involves four primary steps:

1. Selection of window size and stride: The occluding window's size is chosen based on the scale of features that we intends to analyze. A smaller window will reveal finer details, while a larger window may provide insights into more global features. The stride determines the step size by

which the window moves across the input, with smaller strides leading to a more fine grained analysis.

2. Occlusion of input regions: As the sliding window moves across the input, the covered region is replaced with a constant value, typically the mean, median value of the dataset or zeros, effectively "hiding" that portion of the input from the neural network.

3. Inference on occluded inputs: The occluded input is passed through the neural network, and the output is recorded. The difference in the output scores, compared to the original input, indicates the importance of the occluded region in the network's decision making process.

4. Generation of an importance map: The differences in output scores are aggregated into an importance map, highlighting the regions of the input that significantly contributed to the model's prediction. This map serves as a visualization of the network's decision making process, offering insights into which features the model focuses on.

SWO is agnostic, itcan be applied to any neural network architecture, as it relies solely on input manipulation and does not require any modifications to the model itself. It can help identify whether the neural network is relying on spurious correlations or adversarial perturbations in making its predictions, as these would not be visible in the importance map.

In this study, we applied the SWO algorithm on multilead and unilead models (M5:ecg_unilead_LII). We used a fixed stride of 1 and tested different sliding window values from 50 to 500. The most effective sliding window was 50 points (corresponding to 100ms in the 500 Hz recordings) that was iteratively moved across the signal to identify which parts of the ECG signal were the most useful for the classification of ECG as Sot+. Feature importance profile (FIP) was generated for each segment and provided us with a relevant score for identifying which ECG segments were more or less important for predicting Sot+. The occluded portion of the signal was replaced by zeros.

Results and analysis We first tested the interpretability on the multilead model. However, although this model was demonstrated to be highly accurate, it was difficult to interpret. Indeed, the signal from different ECG leads were mixed together during the convolution process in increasingly complex abstractions throughout the neural network. The same occlusion window was applied on each lead at the same moment. A positive contribution for the classification in lead LII, for instance, should be considered as a combination of all the positive contributions in all leads for any beat (Figure 17). Therefore, it was difficult to accurately determine which lead and beat was relevant because the occlusion method considered all time and space variables as a single entity in this model.

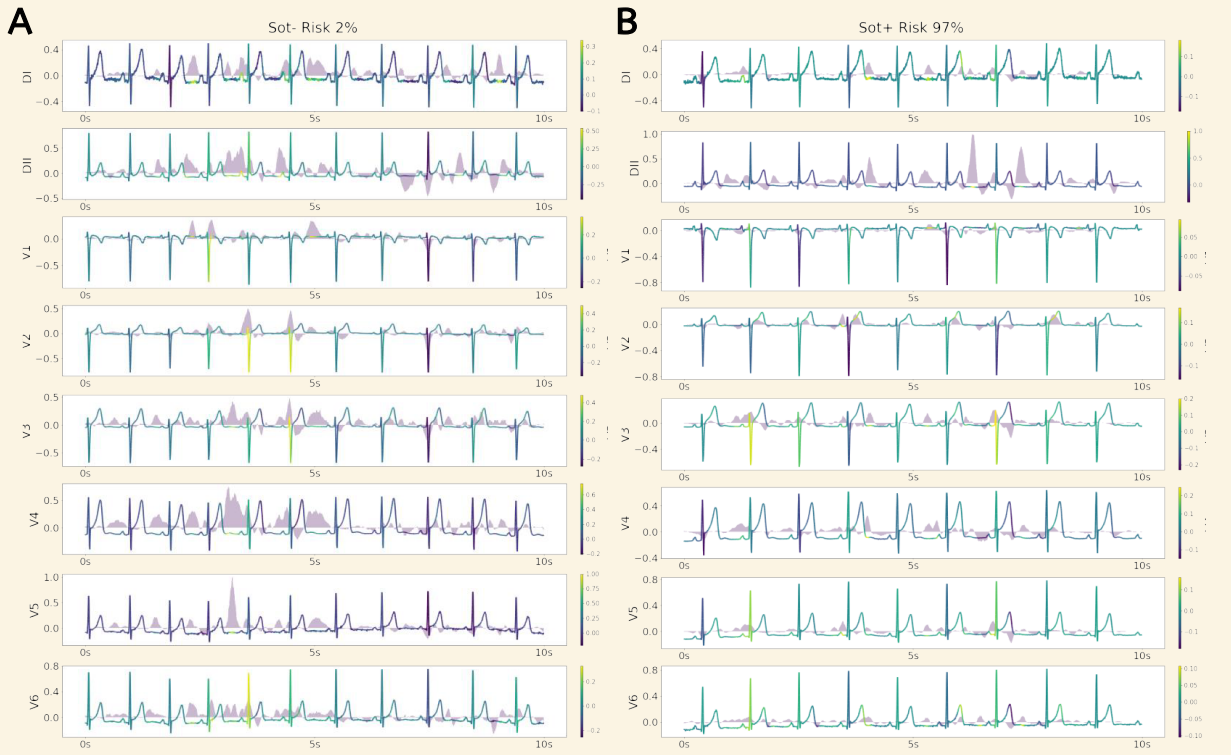


Figure 17: ***Occlusion Interpretability of the Multilead TdP Model:*** *feature importance profile is computed on all leads. A positive amplitude is interpreted as a contribution towards the prediction of class Sot+ wheras as a negative amplitude is interpreted as non contibuting in the prediction of class Sot+. Multilead model is difficultly interpretable as all leads informations are combined during the convolution step making it difficult to distinguish the contribution of each lead, each beat.*

To achieve a more human comprehensible interpretation, we used the unilead model. The unilead model allowed us to better identify patterns in ECG. We used segmented ECG signals by beats in order to summarize the importance of each point (i.e. feature) of the signal in the model's classification decision. In lead II, we found that the standardized FIP changed with increased sotalol blood concentration (maximum at 3 h in Generepol) (18). Initially, at inclusion (before sotalol intake), the FIP was highly negative over the QRS and positive, although with low amplitude, on the P-wave offset and T-wave onset and offset. These features are used by the model to recognize normal ECG complexes without a sotalol footprint: the QRS complex indicating a regularly occurring attribute used to calibrate the data input. One hour after sotalol intake, the FIP distribution started to change. The FIP intensity of the QRS decreased and the importance of the signal after the T-wave and before P-wave onset increased. This region corresponds to the RR time, which is, the cardiac heart rate. Indeed, sotalol has beta-blocking properties known to slow the sinus rate, which were captured by the model. Two hours after sotalol, the FIP increased in the first part of the T-wave (corresponding to the J-Tpeak interval), which reached maximum intensity 3 h after sotalol. At that time, IKr blockade was active and strongly apparent on ECG. We performed the same experiment in unilead models trained on V2 and V3 and FIP behaved similarly.

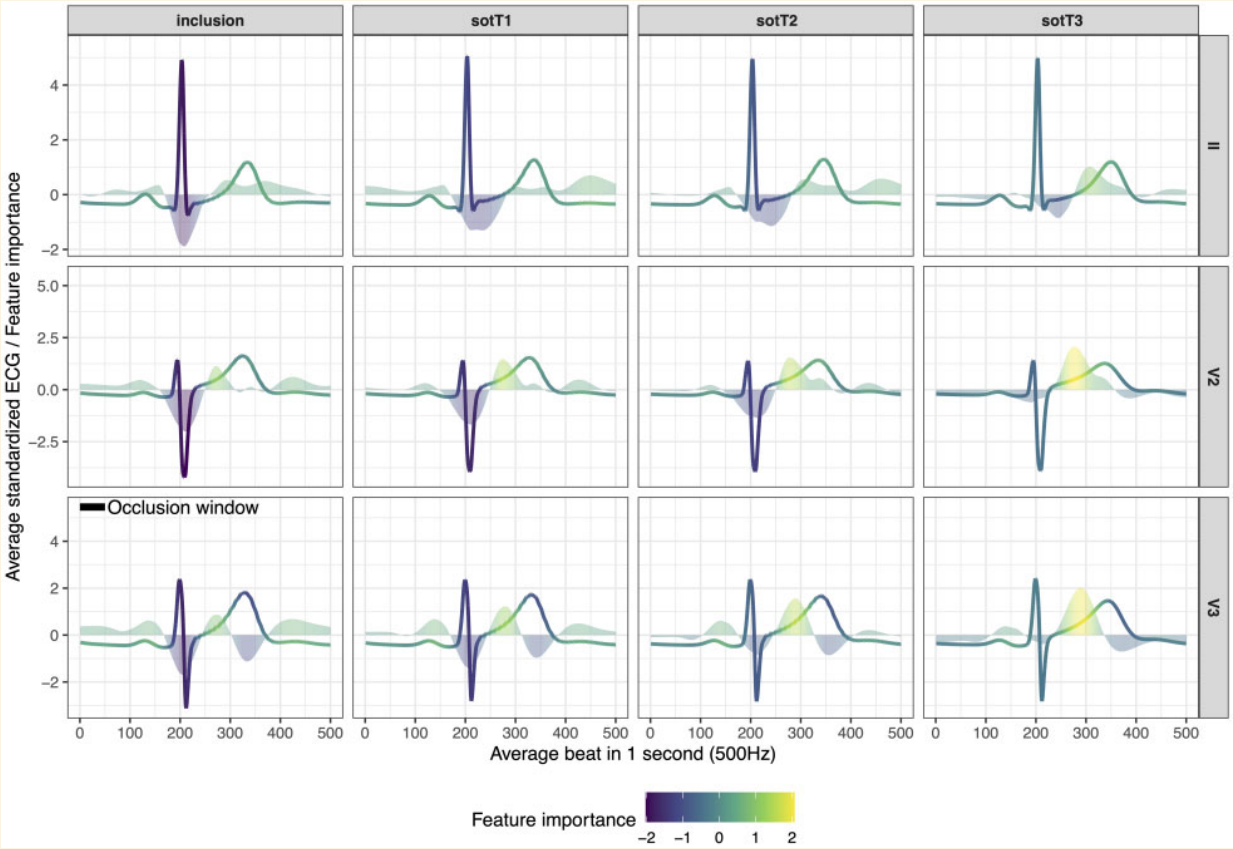


Figure 18: ***Occlusion Interpretability of the Unilead TdP Model:*** *This figure displays an averaged signal of the standardized ECG for each segmented beat for leads LII, V2, and V3. All signals from the same time points were analysed together. Similarly, the standardized feature importance profile is summarized and laid behind the electrocardiogram profile. Colours for both the electrocardiogram and FIP indicate intensity of the feature importance profile.*

Unilead models appear to be much more interpretable as each model processes a single lead making it easier to analyse each beat and determine the contribution of each region of the signal.

2.4 Conclusion

QT prolongation in electrocardiograms (ECGs) has been associated with Torsades de Pointes (TdP), a life-threatening ventricular arrhythmia. However, it is not a very reliable measure. This study proposes an innovative approach using deep convolutional neural networks (CNN) to improve TdP risk prediction by learning the drug footprints of high-risk medications, such as sotalol, in healthy volunteers.

The CNN models were trained using raw digital ECG data, enabling automated and comprehensive TdP risk stratification that complements the QTc measurement. Sotalol, known for prolonging ventricular repolarization through IKr-inhibition, was used as a model drug to induce QTc prolongation and predispose for TdP. The study found that the CNN models accurately detected ECG alterations induced by sotalol. Additionally, these models enhanced the prediction of drug-induced TdP (diTdP) events even after considering QTc and intake of drugs with known TdP risk. Interestingly, single-lead CNN models performed comparably to an 8-lead model, indicating that the sotalol footprint could be detected by all leads similarly, except for lead V1. The models were robust despite technical variability in the ECG data, which were recorded using different acquisition devices and sampling rates.

To identify a cause-to-effect relations in the ECGs towards models's predictions, we used a interpretability method to detect meaningfull patterns in ECGs. The occlusion based interpretability algo-

rithm identified sotalol footprints in ECGs, which changed over time as blood sotalol concentration increased. This finding was consistent with existing knowledge on how sotalol affects cardiomyocyte action potential, primarily through blockade of IKr and beta-adrenergic receptors. The J-Tpeak interval emerged as the main feature for discrimination between sotalol exposed (Sot+) and non exposed (Sot-) signals, supporting emerging literature on the importance of this segment for predicting drug induced TdP proarrhythmic risk beyond QTc.

This study is the first to successfully deploy CNN models to learn drug footprints for predicting heart pathology risk based on ECG. The approach has potential applications in pharmaceutical industry drug monitoring and clinical compliance ascertainment, potentially offering a more practical, less costly, and faster alternative to standard blood analysis. The CNN models also learned clinically relevant knowledge, and their deep embeddings could be used to automatically stratify ECGs and patients into novel classes that are yet to be characterized. Despite these promising findings, the exploration space of model architectures is vast, and identifying the best embeddings can be challenging. More research and training data are needed to advance the translational clinical applications of CNN models in TdP risk prediction and patient care. Furthermore, one major challenge in the adoption of neural networks for clinical decision making is the "black box" nature of these models. Physicians and healthcare professionals must trust the AI's predictions to integrate them into their practice. By making neural networks more interpretable, we can provide clinicians with insight into the rationale behind the model's decisions, enhancing their confidence in the diagnosis or prognosis. Understanding the reasoning behind AI findings allows physicians to compare them to their knowledge and patient records, resulting in more informed choices and better patient outcomes.

2.5 Original paper

ESC European Society of Cardiology

European Heart Journal (2021) **00**, 1–14
doi:10.1093/eurheartj/ehab588

CLINICAL RESEARCH
Arrhythmias

Deep learning analysis of electrocardiogram for risk prediction of drug-induced arrhythmias and diagnosis of long QT syndrome

Edi Prifti [1,2]*, Ahmad Fall [1], Giovanni Davogustto [3†], Alfredo Pulini [1,4†], Isabelle Denjoy [5], Christian Funck-Brentano[6], Yasmin Khan[7], Alexandre Durand-Salmon[7], Fabio Badilini[8], Quinn S. Wells[3,9], Antoine Leenhardt[5], Jean-Daniel Zucker [1,2], Dan M. Roden [3,9,10], Fabrice Extramiana [5], and Joe-Elie Salem [3,6,9]*

[1]IRD, Sorbonne University, UMMISCO, 32 Avenue Henri Varagnat, Bondy 93143, France; [2]Sorbonne University, INSERM, NutriOmics, 91 Boulevard de l'Hopital, Paris 75013, France; [3]Department of Medicine, Vanderbilt University Medical Center, Nashville, TN, USA; [4]Faculty of Medicine, Université de Paris, Paris, France; [5]CNMR Maladies Cardiaques Héréditaires Rares, Hôpital Bichat, Paris, France; [6]Clinical Investigation Center Paris-Est, CIC-1901, INSERM, UNICO-GRECO Cardio-Oncology Program, Department of Pharmacology, Pitié-Salpêtrière University Hospital, Sorbonne Universite, 47 Boulevard de l'Hopital, Paris 7513, France; [7]Banook Group, Nancy, France; [8]AMPS LLC, New York, USA; [9]Department of Pharmacology, Vanderbilt University Medical Center, Nashville, TN, USA; and [10]Department of Biomedical Informatics, Vanderbilt University Medical Center, Nashville, TN, USA

Received 11 April 2021; revised 13 June 2021; editorial decision 12 August 2021; accepted 12 August 2021

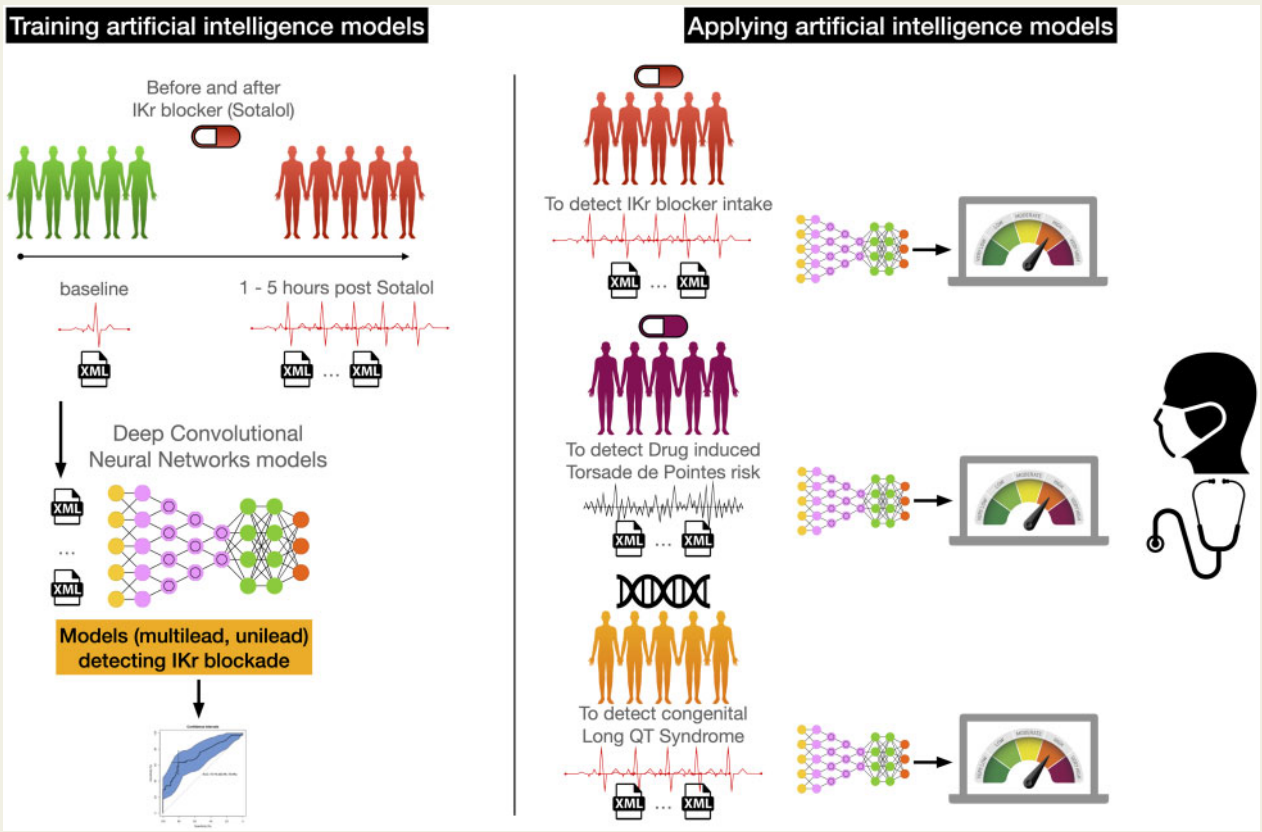
Aims Congenital long-QT syndromes (cLQTS) or drug-induced long-QT syndromes (diLQTS) can cause torsade de pointes (TdP), a life-threatening ventricular arrhythmia. The current strategy for the identification of drugs at the high risk of TdP relies on measuring the QT interval corrected for heart rate (QTc) on the electrocardiogram (ECG). However, QTc has a low positive predictive value.

Methods and results We used convolutional neural network (CNN) models to quantify ECG alterations induced by sotalol, an I_{Kr} blocker associated with TdP, aiming to provide new tools (CNN models) to enhance the prediction of drug-induced TdP (diTdP) and diagnosis of cLQTS. Tested CNN models used single or multiple 10-s recordings/patient using 8 leads or single leads in various cohorts: 1029 healthy subjects before and after sotalol intake ($n = 14\,135$ ECGs); 487 cLQTS patients ($n = 1083$ ECGs: 560 type 1, 456 type 2, 67 type 3); and 48 patients with diTdP ($n = 1105$ ECGs, with 147 obtained within 48 h of a diTdP episode). CNN models outperformed models using QTc to identify exposure to sotalol [area under the receiver operating characteristic curve (ROC-AUC) = 0.98 vs. 0.72, $P \leq 0.001$]. CNN models had higher ROC-AUC using multiple vs. single 10-s ECG ($P \leq 0.001$). Performances were comparable for 8-lead vs. single-lead models. CNN models predicting sotalol exposure also accurately detected the presence and type of cLQTS vs. healthy controls, particularly for cLQT2 (AUC-ROC = 0.9) and were greatest shortly after a diTdP event and declining over time ($P \leq 0.001$), after controlling for QTc and intake of culprit drugs. ECG segment analysis identified the J-T_{peak} interval as the best discriminator of sotalol intake.

Conclusion CNN models applied to ECGs outperform QTc measurements to identify exposure to drugs altering the QT interval, congenital LQTS, and are greatest shortly after a diTdP episode.

* Corresponding authors. Email: edi.prifti@ird.fr (E.P.); Tel: +33 (0)1.42.17.85.35, Email: joe-elie.salem@aphp.fr (J.-E.S.)
†These authors contributed equally to the work.

Graphical Abstract



Convolutional neural network models applied to ECGs outperform QTc measurements to identify exposure to drugs blocking IKr, congenital long QT syndrome, and are greatest shortly after a drug-induced Torsade-de-Pointes episode.

Keywords Torsades de pointes • Machine learning • Risk prediction • Interpretability • Long QT

Introduction

Torsades de pointes (TdP) is a distinctive form of life-threatening polymorphic ventricular arrhythmia associated with prolonged QT interval, corrected for heart rate (QTc), on the electrocardiogram (ECG).[1–3] TdP and QTc prolongation are favoured by congenital or drug-induced alterations in potassium and cardiac sodium channels.[4–6] There are three main forms of congenital long QT syndromes (cLQTS): type 1 and type 2 are caused by loss-of-function mutations in the potassium channels *KCNQ1* (cLQT1, I_{Ks} current) and *KCNH2* (cLQT2, I_{Kr} current), respectively, and type 3 is caused by mutations in *SCN5A* increasing the non-inactivating 'late' sodium current I_{NaL} (cLQT3).[6–8] Drug-induced LQTS (diLQTS) is the other main cause of TdP, with almost all culprit drugs blocking I_{Kr} and the most torsadogenic among them also activating I_{NaL}.[5] Over 100 cardiac or non-cardiac drugs are currently approved despite favouring TdP risk because these drugs are thought to have a favourable risk–benefit ratio in some patients.[9,10]

QTc, which reflects ventricular repolarization duration, is the time between the beginning of the QRS complex and the end of the T-wave.[11] QTc is prolonged in cLQTS and diLQTS and is a hallmark of TdP. Specific T-waveform patterns have been described for each subtype of cLQTS and for diLQTS.[12–15] Current individual and population risk stratification strategies for TdP are almost exclusively based on the quantification of QTc.[4] Regulatory agencies require new drugs to undergo thorough QT studies, where the magnitude of drug-induced QTc prolongation is evaluated as a surrogate for TdP risk.[16] However, limiting ECG evaluation to the QTc is poorly predictive of TdP.[17] An unbiased and complex examination of the ECG data beyond simple QTc prolongation could provide relevant insight into identifying drugs and patients at risk of TdP.

Artificial intelligence is being increasingly applied to complex medical problems.[18] Techniques such as deep learning, including convolutional neural networks (CNN), are bringing a radical change in the field of pattern recognition, improving earlier models in learning tasks such as image classification, ECG analysis, and natural language

processing.[19–21] Herein, we tested if such models were able to learn the ECG footprint of sotalol, an I_{Kr} blocker drug inducing TdP, to develop a new tool using ECG to recognize beyond QTc, exposure to I_{Kr} blocker drugs, and improve the prediction of drug-induced TdP (diTdP) events and classification of cLQTS types, particularly cLQT2 (*Graphical Abstract*).

Methods

Study cohort datasets and QTc measurement

We studied ECGs form four cohorts (*Figure 1*). The 'Generepol cohort' (NCT00773201)[15,22] was conducted at Pitié-Salpétrière Clinical Investigation Center (start–end: 2008–12, Paris, France): ECGs from 990 healthy subjects were recorded before and 1, 2, 3, and 4 h after an 80-mg oral sotalol dose (sotT1, sotT2, sotT3, and sotT4). The 'Pharmacia's cohort' was an open-label, nonrandomized study involving healthy controls (n = 39, 28 males) receiving a fixed oral sotalol sequence administered on 3 successive days: 24-h baseline without sotalol (Day 0); 160 mg in all participants at 8:00 am Day 1; and 320 mg in 21 males at 8:00 am Day 2. The study was conducted at Pharmacia's Clinical Research Unit (start–end: 2002; Kalamazoo, MI, USA).[23,24] The 'cLQTS cohort' included 487 patients confirmed by genetic testing to have one of the three main cLQTS followed at the Arrhythmia Unit of Bichat Hospital (start–end: 1992–2018, Paris, France; 64% asymptomatic).[25] The 'diTdP cohort' included 48 patients prospectively enrolled and followed at Vanderbilt University Medical Center (start–end: 2002–19, Nashville, TN, USA) who had experienced at least one diTdP episode; acute cardiac ischaemia at the time of the event and genetically confirmed underlying cLQTS were exclusion criteria. All cohorts were approved by institutional review boards, and written informed consent was obtained from participants when appropriate.

Recordings from these patients were reviewed by two expert cardiologists and tracings with ventricular or junctional tachycardia during the 10-s acquisition were excluded from the analyses. In all cohorts, QTc was heart rate corrected with Fridericia's formula and details concerning the respective inter and intra-observer variability for QTc measurements in the cohorts are detailed elsewhere.[11,15,22–25]

Data preparation

Raw 10-s ECG data (sampling frequency: 250 and 500 Hz) were acquired with a variety of devices at the different centres. The 250-Hz signals were up-sampled to 500 Hz using a cubic interpolation. The ECG contained eight independent leads (LI, LII, V1–V6), allowing for the reconstruction of 12 leads (addition of LIII, aVF, aVL, aVR). ECGs were provided in .scp or .xml files depending on recording devices (General Electric MAC5500, Marquette MAC15/MACVU, M3700 System, PageWriter Touch/Trim/XL/TC, Mortara ELI200 and Cardionics Cardioplug devices). They were parsed using Biosig software, and Python xmltodict library, as appropriate.[26] The data were stored in Python dictionaries and converted onto

Generepol (n=990; 10292 ECG **Training (80%)** (n=792; 8245 ECG) baseline (4014 ECG), sotalol (4231 ECG) **Holdout (20%)** (n=198; 2047 ECG) baseline (2047 ECG), sotalol (1048 ECG)	**Female** Age (28±11 years) (n = 614; 62%) QTc baseline = 391±15 ms QTc maximal (80 mg sotalol) = 425±21 ms Delta QTc max = 34±14 ms	**Male** Age (28±10 years) (n = 376; 38%) QTc baseline = 377±16 ms QTc maximal (80 mg sotalol) = 400±20 ms Delta QTc max = 22±12 ms
Pharmacia Total (n=39; 3843 ECG) Day 0 (n=39; 1542 ECG) Day 1 (n=39; 1482 ECG) Day 2 (n=21; 819 ECG)	**Female** Age (26±8 years) (n = 18; 46%) QTc baseline = 390±18 ms QTc maximal day 1 (160 mg sotalol) = 459±22ms QTc maximal day 2 (320 mg sotalol) = NA Delta QTc max day 1 (160 mg sotalol) = 74±16 ms Delta QTc max day 2 (320 mg sotalol) = NA	**Male** Age (27±8 years) (n = 21; 54%) QTc baseline = 376±13 ms QTc maximal day 1 (160 mg sotalol) = 424±25 ms QTc maximal day 2 (320 mg sotalol) = 450±22 ms Delta QTc max day 1 (160 mg sotalol) = 48±21 ms Delta QTc max day 2 (320 mg sotalol) = 76±13 ms
Congenital LQT (cLQTS) Total (n=487; 1083 ECG) LQT1 (n=266; 560 ECG) LQT2 (n=188; 456 ECG) LQT3 (n=33; 67 ECG)	**Female** Age (33±17 years) (n = 282; 58%) QTc cLQT1 = 448±35 ms QTc cLQT2 = 455±42 ms QTc cLQT3 = 446±33 ms	**Male** Age (30±19 years) (n = 205; 42%) QTc cLQT1 = 451±38 ms QTc cLQT2 = 450±38 ms QTc cLQT3 = 458±36 ms
Drug-induced TdP (diTdP) Total (n=48; 1105 ECG) 24h (n=38; 103 ECG) 48h (n=31; 44 ECG) >48h+PVC (n=28; 115 ECG) >48h-PVC (n=48; 843 ECG)	**Female** Age (56±16 years) (n = 29; 60%) QTc <24h = 515±60 ms QTc <48h = 471±54 ms QTc >48h+PVC = 460±46 ms QTc >48h-PVC = 460±43 ms	**Male** Age (64±16 years) (n = 19; 40%) QTc <24h = 514±81 ms QTc <48h = 493±54 ms QTc >48h+PVC = 457±54 ms QTc >48h-PVC = 469±50 ms

Figure 1 Experimental design and main characteristics of the study cohorts. Description of the main characteristics of the four study cohorts. The Generepol cohort was composed of healthy volunteers given a single 80-mg dose of oral sotalol. This dataset was used to train and test the models. The Pharmacia's cohort was composed of 39 healthy volunteers before (Day 0) and after a single 160-mg dose of oral sotalol (Day 1), followed in some men by a single 320 mg dose of oral sotalol (Day 2). This cohort was only used to test the models. The congenital long-QT syndrome (cLQTS) cohort was composed of congenital long-QT syndrome patients of Types 1, 2, and 3. The drug-induced torsade de pointes (diTdP) cohort included patients who experienced events of drug-induced torsade de pointes with no underlying identified congenital long-QT syndrome.

3D tensors (8 leads, 5000 time points for each lead, recordings) used to train and test the models. Standardization was performed at the whole ECG level, with each lead signal standardized by the mean of all other lead signals for models including all eight leads ('multilead') and at the lead level for 'unilead' models. No other transformations, including filtering, were used.

Sotalol-intake classification with the multilead and unilead models

We used either the eight leads concomitantly (LI, LII, V1–6; termed 'multilead') or each of the eight leads independently ('unilead') to train a CNN model to predict *Sot+* (having received sotalol, as a surrogate for I_{Kr} blockade) and *Sot−* classes (normal ECG before sotalol intake). The Generepol cohort (healthy volunteers before and after sotalol intake) was split into two sets: general training (80% for multilead models, 90% for unilead models) and holdout (20% for multilead models, 10% for unilead models). Ten times 10-fold cross-validation was performed in the general training set for parameter optimization. Each split was performed according to the subjects' IDs and, therefore, each training partition had distinct subjects from the testing split. Descriptions of how the multilead and unilead models were constructed are provided in the Supplementary material online, *Figures S1 and S2*. After cross-validation, each model was trained on the training set of the Generepol cohort. Then, models were tested on the holdout Generepol set (completely independent of the training set) and the three other study cohorts.

Voting vs. single electrocardiogram analysis

Performance indicators were computed using both 10-s single ECG signal analysis (ECG level in figures) and by averaging risk scores from multiple recordings acquired within minutes of a same timepoint (multiple 10-s recordings; patients' level in figures; i.e. the voting analysis) for a given patient and condition. The output provided by the models was a score ranging from 0 to 1 indicating a likelihood of being *Sot+* (having ingested sotalol). To classify a patient into *Sot+* or *Sot−* classes, ECGs from the same patient were processed by the models and the patient was affected as being *Sot+* (vs. *Sot−*) based on the mean classification score of the different 10-s ECG, on which a threshold of 0.5 was applied (*Sot+* if score ≥0.5). The performance metrics of all tested models can be found in *Figures 2–4*, Supplementary material online, *Figures S3 and S4*, and Supplementary material online, *Table S1*.

Embedding analyses

CNN models generate outputs, such as *Sot+* or *Sot−*, by analysing raw input through a series of intermediate 'layers' termed embeddings.[27] A distinctive feature of CNN models is their ability to discover novel representations of complex data, and one way to access such knowledge is by extracting the embeddings (transformation of the input data by the neural network). In this study, ECGs were transformed in the CNN embeddings by deriving vectors of 512 values. To represent these complex datasets in two dimensions for human interpretation, a nonlinear dimension reduction technique was applied based on the t-SNE algorithm[28] (perplexity = 100, iteration = 1000) using the Rtsne package. ECG data (vectors of 512 values) were thus visualized and annotated as points on these maps. All dimensions of the embeddings were used to identify partitions with the k-means method with default parameters implemented in base R. Details concerning embedding analyses are in Supplementary material online, *Figure S5*.

Electrocardiogram segment occlusion analysis (interpretability)

We sought to identify which parts of the ECG signals were most useful in our classification models to classify an ECG as *Sot+*. To accomplish this goal, we iteratively dropped ('occluded') a predefined portion of the data (in this case, a segment of the ECG signal) and re-performed the prediction. Here, we used a window of 50 points (corresponding to 100 ms in 500 Hz recordings) that was iteratively moved across the signal to identify which parts of the ECG signal were the most useful for the classification of ECG as *Sot+*. Feature importance profile (FIP) was generated for each segment and provided us with a relevant score for identifying which ECG segments were more or less important for predicting *Sot+*. Details concerning occlusion methods are in Supplementary material online, *Figure S6*. We implemented the occlusion method in Python with Tensorflow-2.

Statistical analyses

Data are presented as count and frequencies, or median and interquartile range (IQR) for categorical and continuous variables, respectively. We used mixed-effects linear models to best describe the data and their relations while controlling for random effects such as patient ID (multiple recordings per patient). Models were compared using ANOVA and the best models were selected based on the Akaike information criterion. Accuracy, recall, precision, F1 score, and area under the receiver operating characteristic curve (ROC-AUC) were used to evaluate the different models generated. The Chi^2-test was used for comparing proportions. Statistics and graphics were performed using R-packages (lme445, lmerTest, ggplot2, pROC). A $P \leq 0.05$ was deemed statistically significant; all tests were two-tailed.

Results

Study population characteristics

The main characteristics of the four study cohorts are summarized in *Figure 1*.

The Generepol cohort contained 10 292 10-s ECG recordings from 990 healthy subjects (62% women, median [range] age 24 [18–60] years) in sinus rhythm before and 1, 2, 3, and 4 h after the administration of 80 mg sotalol (respectively, denoted as baseline and sotT1–sotT4). The median number of 10-s ECG/participant in this cohort was 15 [range: 12–18].

The Pharmacia's cohort contained 3843 10-s ECG recordings from 39 healthy subjects (46% women, median [range] age 25 [18–45] years) in sinus rhythm before and up to 12 h after the intake of 160 mg sotalol on Day 1 and 320 mg sotalol on Day 2. The median number of 10-s ECG/participant in this cohort was 114 [range: 42–117].

The cLQTS cohort included 487 participants (median [range] age 28 [0–84] years; confirmed by genetic testing) with 1083 10-s ECG recordings (median number of ECG/patient 3, IQR 6, longest follow-up 23 years). The three cLQTS types were represented, with 266 cLQT1 (62% women), 188 cLQT2 (54% women), and 33 cLQT3 (45% women) patients. A total of 213 participants (44%) had at least one recording performed while on beta-blocker, with 116, 88, and 9 (44%, 47%, and 27%) participants for cLQT1, cLQT2, and cLQT3, respectively. ECGs were in sinus rhythm, except for 8 (0.7%) with supra-ventricular arrhythmia and 2 (0.2%) with either atrial and/or ventricular pacing.

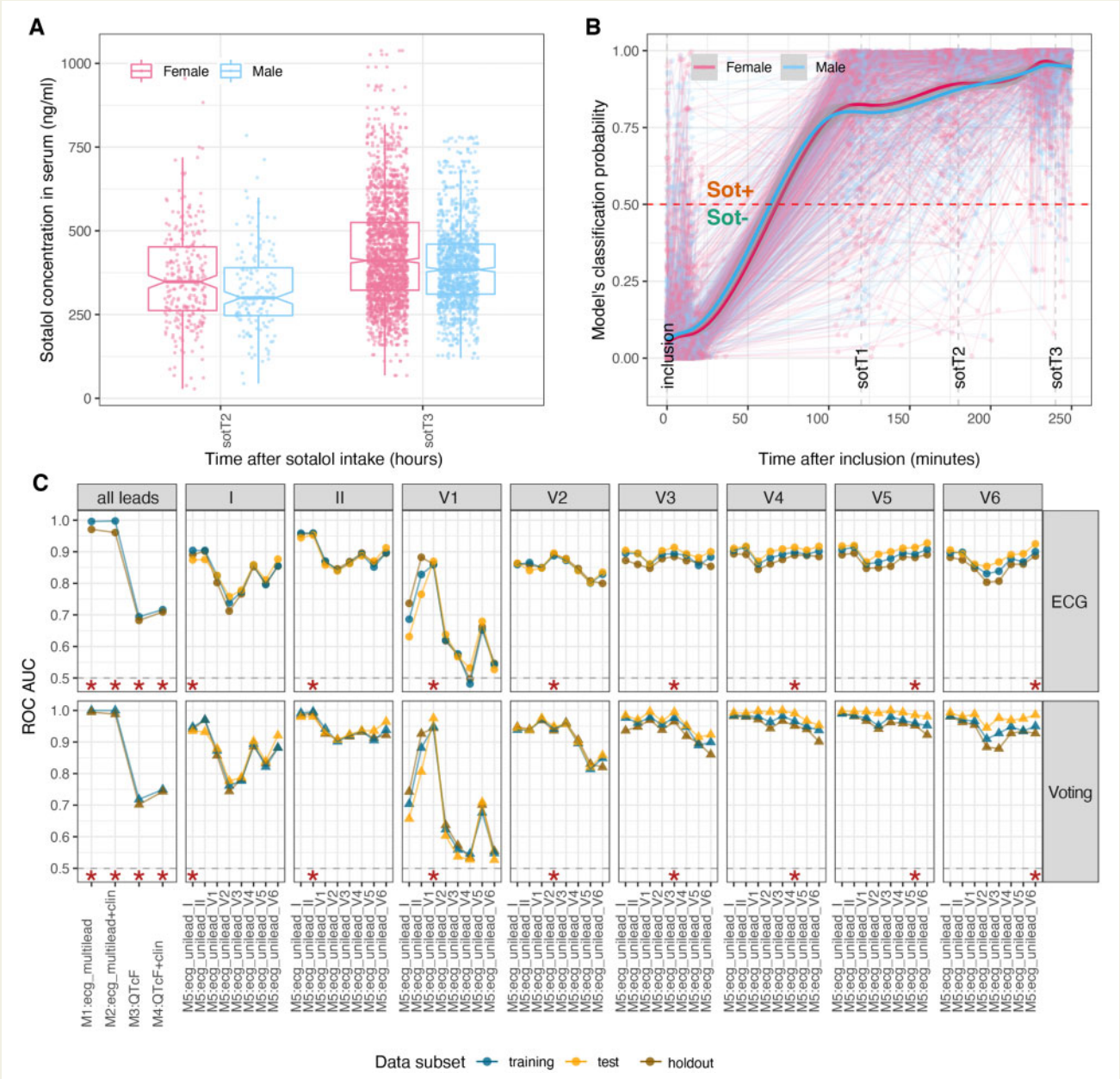


Figure 2 Classification performance of convolutional neural network and linear regression (QT) models in discriminating baseline electrocardiogram before sotalol from those after sotalol intake (SotT1, SotT2, SotT3) in Generepol. (*A*) Boxplots, illustrating the distribution of circulating sotalol concentration (ng/mL) in Generepol cohort two and three hours after 80mg oral sotalol intake. Data are displayed separated and coloured by gender. (*B*) Scatterplot illustrating the evolution of the M1: ecg_multilead classification score for the *Sot+* class (*y*-axis) across time from inclusion (*x*-axis) in the Generepol cohort. All points (averaged electrocardiograms) of a study participant are linked together as trajectories and are coloured by gender. Summarized loess (local regression) distribution of the data ± standard error is overlaid on top and grouped by gender. The red horizontal line corresponds to the *Sot+*/*Sot−* classification threshold (= 0.5). (*C*) Area under the receiver operating characteristic curve for the convolutional neural network multilead models (M1, M2), non- convolutional neural network standard QT-based linear regression models (M3, M4) as well as all convolutional neural network unilead M5 models in classifying each individual 10-s electrocardiogram recording (top) or using a voting strategy (in triplicates of 10-s electrocardiogram per study participant and time point, bottom). Multiple 10-s electrocardiograms recorded at each time point were assigned a *Sot+* classification score. When the risk score was ≥0.5, the electrocardiogram was classified as *Sot+*. With the voting approach, a mean *Sot+* classification score was computed. The same threshold was applied to predict the *Sot−*/*Sot+* class. Blue, orange, and brown colours, respectively, depict the training, test, and holdout subsets of the Generepol's cohort (see *Figure 1*). Each model tested on the same lead as trained is annotated by a red star. For the multilead models, all leads are used to train and test.

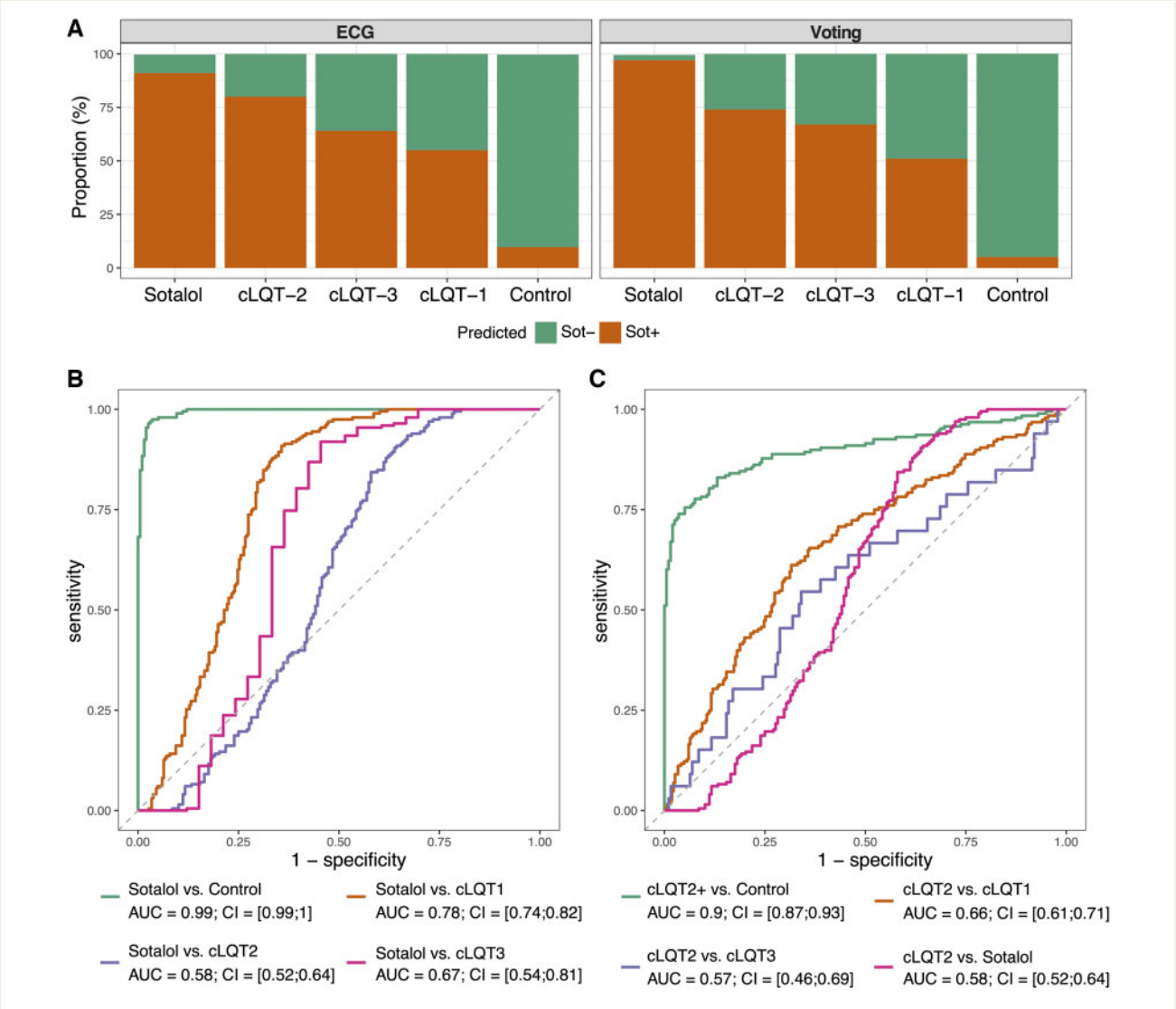


Figure 3 Convolutional neural network model performance in classifying study participants as Sot+/Sot- in Generepol holdout dataset and congenital long-QT syndrome cohort. (*A*) Left: Percentage of all electrocardiogram for study participants, which are classified as *Sot+* in the holdout Generepol dataset [healthy volunteers before (Control) and 1–3 h after sotalol intake (Sotalol)] as well as the cLQT1, cLQT2, and cLQT3 groups. Right: Similar to the left panel, with the exception that groups of electrocardiogram were classified as *Sot+* using the patient voting strategy instead of individual 10-s electrocardiogram. (*B*) Receiver operating characteristic curves indicating the separation between patients on sotalol (Sotalol) and each of the control, cLQT1, cLQT2, and cLQT3 groups. (*C*) Receiver operating characteristic curves indicating the separation between cLQT2 and cLQT1, cLQT3, sotalol exposed, and control groups.

The diTdP cohort included 48 participants (60% women; median [range] age at the time of the first ECG 60 [18–85] years) with 1105 10-s ECG recordings (median number of ECG/patient 31). The median follow-up was 4 years [range: 0–17]. Sixty-six percent of the 10-s ECG ($n = 733/1105$) were recorded while patients were on I_{Kr} blocker drugs with known risk for TdP,[9] with amiodarone (29/48), sotalol (12/48), dofetilide (9/48), fluconazole (7/48), and hydroxychloroquine (4/48) being the most prevalent.[4,15,29] Some patients took multiple drugs with TdP known risk (one drug: 69%, two drugs: 24%, three drugs: 5%). Recordings from these patients were classified into four categories using the combination of delay between ECG intake and the diTdP event, associated with the presence/absence of premature ventricular contractions (PVC): <24 h, 24–48 h, >48 h + PVC and >48 h - PVC. Of these 1105 ECG recordings, 930 were obtained in sinus rhythm (84%), 171 (15%) in supraventricular arrhythmia, and 4 in junctional rhythm. A total of 162 (15%) and 183 (17%) 10-s ECG had at least one ventricular and/or atrial paced complex. At least one PVC was seen in 143 (13%) ECGs.

QTc evaluation

Serial QTc surveillance is the method cardiologists use to evaluate TdP risk in clinical practice.[16] When QTc is >480 ms or is increased

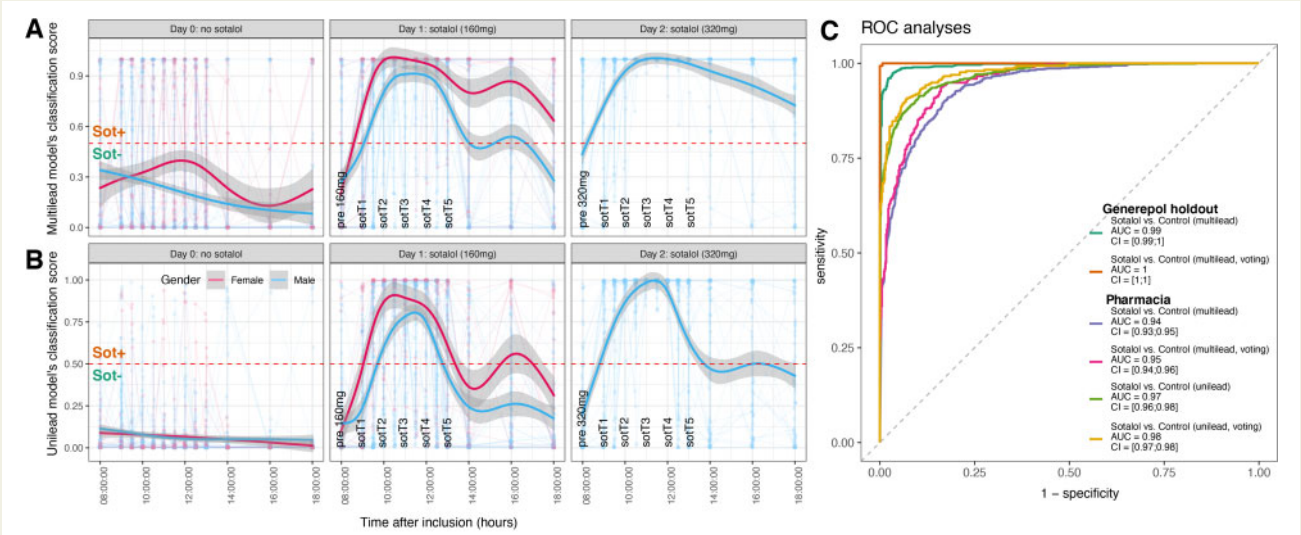


Figure 4 Convolutional neural network model performance in classifying study participants as Sot+/Sot- in Pharmacia's cohort. Scatterplot illustrating the evolution of the M1: ecg_multilead (*A*) and M5: ecg_unilead_II (*B*) classification score for the *Sot+* class (*y*-axis) across time from inclusion (*x*-axis) in Pharmacia's cohort. All points, single electrocardiograms of a study participant, are linked together as trajectories and are coloured by gender. Summarized loess (local regression) distribution of the data ± standard error is overlaid on top and grouped by gender. The horizontal dotted line corresponds to the *Sot+*/*Sot−* classification threshold. (*C*) Receiver operating characteristic curves indicating the separation between subjects on sotalol (Sotalol; 2–4 h post 160 mg intake on Day 1 and post 320 mg intake on Day 2) versus before (Day 0 and before intake of Day 1) in Pharmacia's study.

by ≥60 ms after drug intake compared to baseline, patients are considered at potential TdP risk.[16] In Generepol, the mean QTc at baseline was 14 ms higher in women vs. men (391 ± 15 vs. 377 ± 16 ms; $P < 2e\text{-}16$) as is well-recognized.[8,30] Maximal QTc prolongation after sotalol was more pronounced in women vs. men (34 ± 14 vs. 23 ± 12 ms; $P < 2e\text{-}16$). Similar results were obtained in the Pharmacia's study when comparing QTc before and after sotalol intake (*Figures 1 and 5*).

In cLQTS, no difference in QTc was detected among the three types of cLQTS on the first ECG available for each patient (449 ± 36, 453 ± 40, and 452 ± 35 ms for cLQT1, cLQT2, and cLQT3, respectively, $n = 483$; *Figure 5*). The mean QTc in the cLQTS cohort was 65 ms greater than the pre-sotalol values from Generepol (451 ± 38 vs. 386 ± 18 ms, $P < 2e\text{-}16$).

In the diTdP cohort, QTc values were higher within 24-h of diTdP events (501 ± 70 ms) vs. within 24–48 h (478 ± 45 ms; $P < 0.02$), or vs. >48 h with and without PVC (455 ± 50 ms, $P < 2.14e\text{-}8$ and 459 ± 45 ms, $P < 8.5e\text{-}12$, respectively; *Figure 5*). The mean QTc in the diTdP cohort was 86 ms longer than the pre-sotalol values from Generepol (469 ± 63 vs. 386 ± 18 ms, $P < 2e\text{-}16$).

Convolutional neural network models and sotalol intake on electrocardiogram

To learn the sotalol footprint as a proxy of drug-induced I_{Kr} blockade on ECG, we trained different CNN models (M) on a subset of Generepol. The first model used all leads (LI–II, V1–V6) from raw ECG data (M1: ecg_multilead). A second model used clinical information (age, sex, and serum potassium) in addition to the ECG data (M2: ecg_multilead +clin). In this study, we first focused on 10-s ECG recordings at baseline before sotalol, and 1, 2, and 3 h after sotalol intake.

The models provided an output score indicating a likelihood of sotalol intake in the [0–1] range. A score of 0 predicted the absence of sotalol intake whereas 1 corresponded to the highest probability for sotalol exposure. The mean predicted score at baseline was low (0.06) but increased rapidly for ECG recorded at one (*SotT1*, 0.80) and two (*SotT2*, 0.88) and peaked at 3 h after sotalol (*SotT3*, 0.95) (*Figure 2*); there were no sex differences. Notably, this increase in model score predictions tracked the increase in sotalol blood concentration (*Figure 2*). The output score was then converted into a binary variable based on a threshold (*Sot-* if model-derived score <0.5, *Sot+* if ≥0.5). Performance indicators [ROC-AUC, accuracy, precision, recall (all ranging within [0–1]), and F1 score (ranging within [0–0.5])] were evaluated for each model in 10-s ECG recording individually or on the mean of multiple 10-s ECG of the same participant at a given time point ('voting strategy'), in the training, cross-validation, and holdout sets (*Figures 2 and 3* and Supplementary material online, *Figures S3 and S4*). The mean cross-validation ROC-AUC of M1: ecg_multilead for discriminating the ECG of patients before vs. after sotalol intake was 0.948 when computed on single 10-s ECG and 0.98 with the voting approach. Similarly, for M2: ecg_multilead + clin, the mean test accuracy was 0.948 (ECG) and 0.98 with voting (*Figure 2* and Supplementary material online, *Figure S3*). No difference was observed between M1 and M2. This indicated that the information contained in age, sex, and serum potassium was likely embedded in the ECG footprint captured by the CNN model.

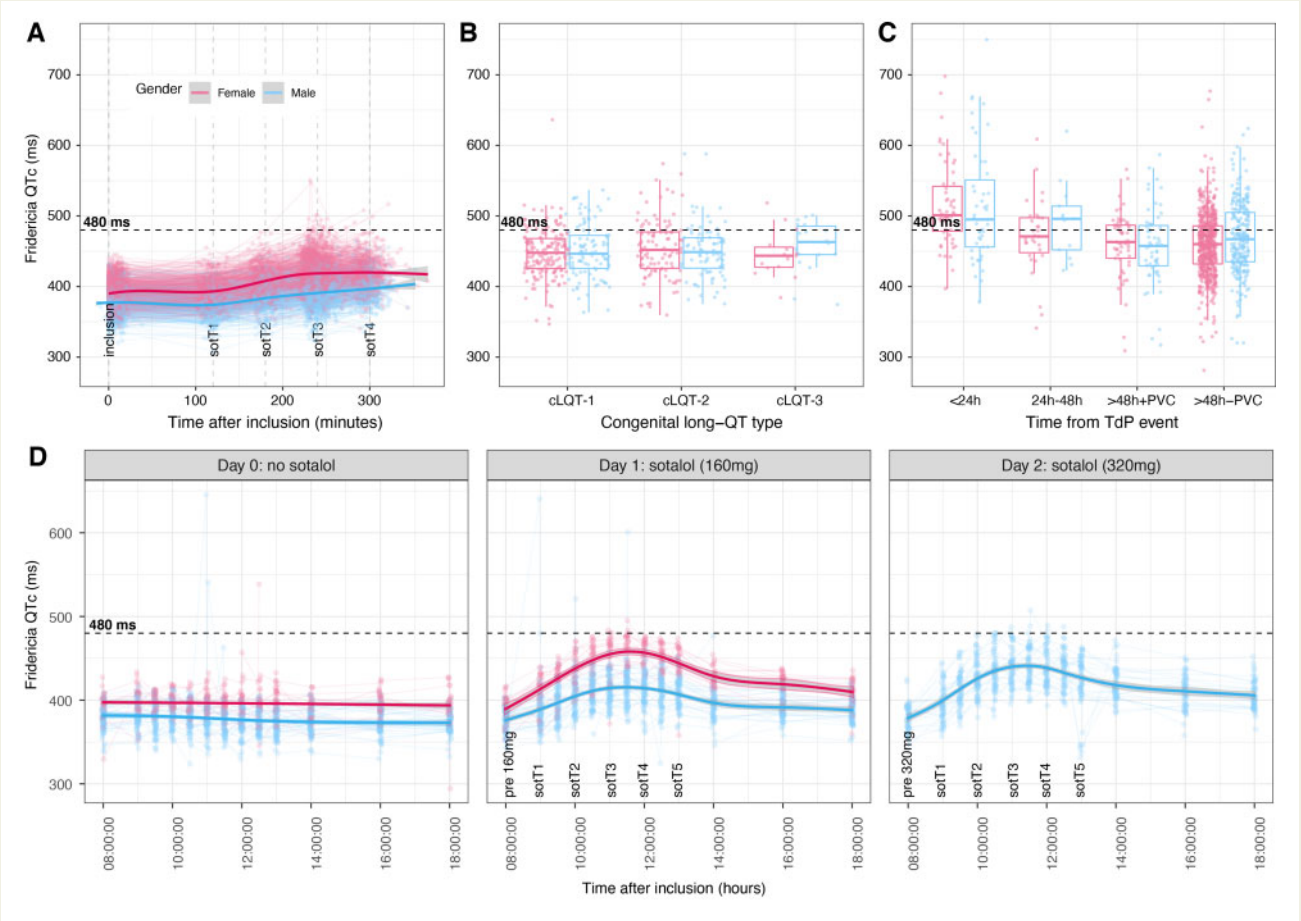


Figure 5 QTc distribution across study cohorts. The distribution of QTc values following the sotalol-induced QTc prolongation in the Generepol (*A*) and Pharmacia's cohort (*D*). The *X*-axis represents the time (min) following sotalol administration and lines link electrocardiogram recordings from the same participant over the duration of the protocol. Summarized loess (local regression) distribution of the data ± standard error is overlaid on top and grouped by gender (males in blue and females in red). (*B*) Boxplots of the estimated QTc values in the congenital long-QT syndrome cohort by subtypes. (*C*) Boxplots of QTc values across the drug-induced torsade de pointes cohort grouped by time to torsade de pointes event and presence or not of premature ventricular contractions. Electrocardiograms are grouped and coloured by gender and the black horizontal lines indicate the 480-ms at-risk QTc threshold (*A*–*D*).

Therefore, the M1: ecg_multilead model was deemed sufficient to be used thereafter. Its precision (voting), recall, and F1 score were very high (0.955, 0.927, and 0.470, respectively).

For comparison with current practice, we also tested the performance of QTc (M3: QTcF) alone, and with the same additional clinical information as above (M4: QTcF + clin) to discriminate on the presence/absence of sotalol intake. The linear regression model based on QTc alone (M3: QTcF) displayed a lower ROC-AUC of 0.695 (10 s ECG) and 0.720 (voting) vs. M1: ecg_multilead (ROC-AUC: 0.948 and 0.98, respectively; $P < 1.5e\text{-}141$). After integration of clinical data to QTc (M4: QTcF + clin), model performance increased significantly ($P < 3.3e\text{-}16$) to 0.717 (ECG) and 0.750 (voting) vs. M3: QTcF (*Figure 2* and Supplementary material online, *Figure S3*). Overall, QTc models were less effective than CNN models, even after integration of relevant clinical covariates. All four models (M1–4) displayed significantly higher ROC-AUC with the voting vs. individual 10-s ECG strategy ($P < 1.2e\text{-}20$ for M1: ecg_multilead, Supplementary material online, *Table S1*). This demonstrates the importance of having longer recordings of at least 30 s (mainly triplicates of 10-s ECG in our study). Results were similar in the holdout set (*Figures 2 and 3* and Supplementary material online, *Figure S3*). All performance indicators for all these models are in Supplementary material online, *Figures S3 and S4* and Supplementary material online, *Table S1*.

Thereafter, we tested the hypothesis that the ECG footprint for sotalol exposure could also be detected by the analysis of single leads. For this, we trained eight different models—one for each lead (LI, LII, V1–V6; see 'Methods'). Their performances were comparable to the multilead models (*Figure 2* and Supplementary material online, *Figure S4*). The best scores were obtained with the model trained and tested on lead LII [M5: ecg_unilead_LII; ROC-AUC = 0.958 (10 s ECG) and 0.992 (voting) in the holdout set]. When this model trained on one lead was tested on the rest of the leads, it performed well, with mean holdout AUC-ROC of 0.883 (10-s ECG) and 0.96 (voting). However, while the mean recall was high 0.913 (10-s ECG) and 0.963 (voting), the precision was lower 0.597 (10-s ECG) and 0.605 (voting). Similar results were obtained with other unilead

models, except for the one trained on V1, which did not generalize well on the other leads (Supplementary material online, *Figure S4* and Supplementary material online, *Table S1*).

Finally, we validated M1: ecg_multilead and M5: ecg_unilead_LII models (trained in the training subset of Generepol) in the Pharmacia's cohort, an independent dataset of healthy controls before and after sotalol intake. Both M1 and M5 models performed very well to discriminate sotalol intake using ECGs (ROC-AUC 0.94–0.98 depending on the models, 10-s ECG vs. voting; *Figure 4*).

Convolutional neural network models and congenital long-QT syndromes types

Since diLQTS and cLQTS are both characterized by prolonged QTc, we hypothesized that the models (M1: ecg_multilead) trained to recognize the sotalol EGG footprint would also be able to discriminate ECG from cLQTS subjects compared to Generepol baseline data, particularly for cLQT2, which shares the same pathophysiological mechanism of I_{Kr} blockade with sotalol-induced LQTS. We used M1: ecg_multilead trained on a subset of Generepol (80%) and applied it to evaluate its potential in discriminating ECG from the heathy volunteers before and after sotalol intake (20% holdout from Generepol, never used for training) and cLQTS patients. The model prediction results confirmed our hypothesis (*Figure 3*). First, we showed that the vast majority of ECGs before and after sotalol intake (95%, 97%, voting, respectively) from the holdout Generepol cohort were correctly classified as *Sot+* and *Sot−*, respectively (*Figure 3*). Second, most ECGs from cLQTS (66%) were classified as *Sot+*. cLQT2 displayed the strongest proportion (80%, 74%) of *Sot+*, followed by cLQT3 (64%, 67%) and cLQT1 (55%, 51%) at the individual 10-s ECG level and after voting, respectively. *Figure 3* displays AUC-ROC results comparing ECGs from healthy participants on sotalol vs. their baseline ECG before sotalol (controls), cLQT1, cLQT2, and cLQT3. M1: ecg_multilead was highly efficient (AUC-ROC = 0.9) in discriminating cLQT2 from healthy controls contrasting with low AUC-ROC (0.58) of ECG analysis from cLQT2 vs. healthy subjects having received sotalol. These results indicate that M1: ecg_multilead could not discriminate well between these latter two groups, supporting the hypothesis of shared ECG footprints alterations between cLQT2 and sotalol intake (I_{Kr} blockade). Notably, M1: ecg_multilead moderately separated cLQT2 from cLQT1 and cLQT3 (*Figure 3C*). The mean M1: ecg_multilead ECG-derived score in cLQT2 was 0.53, significantly higher than cLQT1 (0.34, $P < 7.4e-7$) and cLQT3 (0.43, $P < 0.14$), after adjustment for beta-blockers (accounting for significant interaction between beta-blockers intake and cLQT2, effect size = 0.19, $P < 7.3e-6$; but not for other cLQTS types). Of note, age and sex were not significantly associated with M1: ecg_multilead score in cLQTS.

Convolutional neural network models and drug-induced torsade de pointes events

We evaluated M1: ecg_multilead model to predict the risk of diTdP events in the diTdP cohort. We quantified the association between M1: ecg_multilead score and the TdP footprint on ECG from patients who had had a diTdP event. The TdP footprint was coded as a four-class variable combining the delay from the diTdP event (<24, 24–48, and >48 h) and the existence or absence of PVCs in the >48-h subgroup. Using a mixed linear model, we showed that TdP footprint was associated with the M1: ecg_multilead score (highest within 24 h from diTdP vs. >48 h from diTdP without PVC (mean: 0.68 vs. 0.56, $P < 0.0018$; *Figure 6*) after adjusting for a significant association with QTc ($P < 1.87e-10$) and intake of drugs with a known risk for TdP ($P < 3.17e-7$).

Convolutional neural network and novel representation of electrocardiogram data

The complex representation of an ECG, learned by the layers of the M1: ecg_multilead CNN model, is contextual to the presence or absence of the sotalol footprint. We extracted these representations (embeddings) of all the ECGs of the studied cohorts by accessing the output of the last convolutional layers (see Supplementary material online, *Methods*). When annotating all ECGs from Generepol as a function of the M1: ecg_multilead predicted risk score (*Figure 7A*), we noticed a gradient pattern corresponding closely to the time between ECG acquisition and sotalol intake (*Figure 7B*). This demonstrated the relevance of what the model 'learned' from the ECG data in recognizing sotalol exposure. In cLQTS, most of cLQT2 ECG were located in the high-level score zone of the t-SNE map (top part of the map), indicating ECG features resembling those of sotalol-induced I_{Kr} blockade as seen previously. This contrasts with those from cLQT1 and cLQT3, which were uniformly distributed in the t-SNE map (*Figure 7C*). In the diTdP cohort, most ECGs were located near the average to high-risk zones of the t-SNE map, being particularly high when recorded within 24 h of the diTdP (*Figure 7D*), at a time when residual I_{Kr} blockade was most likely to be present. Taken together, these results indicate that the classification accuracy in recognizing the sotalol footprint also extends to CNN M1 model-identified embeddings, which condense clinically relevant information. Such novel representations of the data open perspectives for novel TdP risk stratification of ECG and patients (Supplementary material online, *Figure S5*).

Interpretability analyses of convolutional neural network

Figure 8 displays the results of the 'occlusion analysis' designed to identify ECG sub-segments (i.e. features) most important for the models. In lead II, we found that the standardized FIP changed with increased sotalol blood concentration (maximum at 3 h in Generepol). Initially, at inclusion (before sotalol intake), the FIP was highly negative over the QRS and positive, although with low amplitude, on the P-wave offset and T-wave onset and offset. These features are used by the model to recognize normal ECG complexes without a sotalol footprint—the QRS complex indicating a regularly occurring attribute used to calibrate the data input. One hour after sotalol intake, the FIP distribution started to change. The FIP intensity of the QRS decreased and the importance of the signal after the T-wave and before P-wave onset increased. This region corresponds to the RR time, that is, cardiac heart rate. Indeed, sotalol has beta-blocking properties known to slow the sinus rate, which were captured by the model. Two hours after sotalol, the FIP increased in the first part of the T-wave

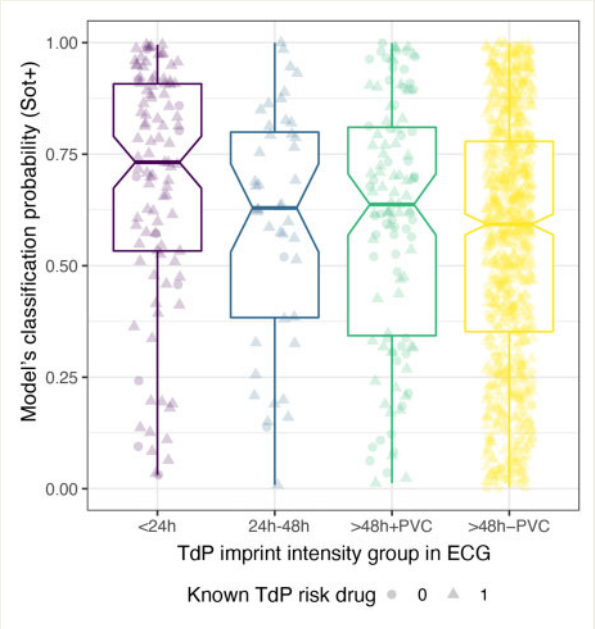


Figure 6 M1: ecg_multilead *Sot+/Sot−* model's classification score in relation to drug-induced torsade de pointes imprint intensity in electrocardiogram. Boxplots indicating the distribution of convolutional neural network M1 model's classification score in patients' electrocardiogram as a function of torsade de pointes imprint intensity groups. Shape indicates the intake of drugs with known risk for torsade de pointes (triangles) vs. none (circles).

(corresponding to the J-Tpeak interval), which reached maximum intensity 3 h after sotalol. At that time, I_{Kr} blockade was active and strongly apparent on ECG. We performed the same experiment in unilead models trained on V2 and V3 and FIP behaved similarly (*Figure 8* and Supplementary material online, *Figure S6*).

Discussion

QTc prolongation, although imperfect, has been shown to be associated with TdP and is currently used in clinical practice as a surrogate for evaluating the risk of TdP.[31] Here, we propose a new approach to improve TdP risk prediction. We hypothesized that it would be possible to use cutting edge artificial intelligence models to learn the footprint of drugs at the high risk of TdP in healthy volunteers. We then used these models to quantify a novel risk score in other participants exposed to these drugs or in patients with cLQTS. The main finding of our study is that training deep CNN models using raw digital ECG data allows for an automated and comprehensive TdP risk stratification that complements QTc measurement. The CNN was trained to recognize ECG alterations induced by sotalol as a model of I_{Kr} blockade, the major mechanism by which drugs cause QTc prolongation, and predispose to TdP.[15] The CNN models accurately detected ECG associated with the intake of drugs at risk of TdP and discriminated the presence and type of cLQTS, being particularly accurate for cLQT2. Moreover, these models improved the prediction of diTdP event, even after controlling for QTc and intake of drugs at known risk of TdP. Analyses of the CNN models highlighted specific interpretable ECG features, particularly the J-Tpeak interval to recognize the sotalol-induced ECG footprint. Models based on a single lead performed in general as well as those using eight leads, except for V1.

Because TdP is a relatively rare event, we first used a population of healthy volunteers exposed to sotalol so we could generate enough labelled data for the CNN model to be robust. The rationale for using a cohort exposed to sotalol is that this drug is known to prolong ventricular repolarization through I_{Kr} inhibition, that rarely but dose dependently can lead to TdP.[32,33] The CNN models developed here were able to accurately classify if a patient was or not exposed to sotalol, regardless of the time after drug intake. Furthermore, multiple acquisitions taken together with a voting approach improved the classification. This demonstrated the presence of rich information contained within the ECGs, exceeding the sole measurement of QTc including with relevant clinical information. Classification from ECG features learned in the CNN models could become a useful approach in compliance ascertainment and drug adjustment, eventually more practical, less costly, and faster than standard blood analysis.

Similar molecular and physiological mechanisms to sotalol action are known to be involved in cLQT2 patients with *KCNH2* mutations, which also lead to decreased I_{Kr} current.[15] Here, we demonstrated that the similarities of the sotalol ECG footprint with cLQT2 allowed to accurately classify 80% of the ECG from cLQT2 patients. This result has potential clinical applications such as screening incoming patients for cLQTS and discrimination of types, with very low cost, before using more expensive genetic tests or scarce expert ECG repolarization evaluation. Although QTc is prolonged in all cLQTS, the ECG waveforms carry specificities including T-wave morphology abnormalities that are specific to each type of cLQTS.[34] However, the models developed herein were not trained to distinguish the different cLQTS groups, particularly cLQT1 and 3, for which more data are needed.

When applied to an independent study cohort of patients who experienced diTdP events, our CNN model-derived scores were higher within 24 h of the diTdP events vs. ECGs from same individuals >24 h (and even more 48 h) after or before the event. These results indicate that such models could be helpful to diagnose patients who experienced an out-of-hospital TdP event or even risk stratify patients with continuous surveillance for emerging diTdP events.

To the best of our knowledge, this is the first study that successfully deploys the original approach of learning drug footprints to predict drug-induced heart pathology risk based on ECG. A prior study was able to correlate drug concentrations on ECG using CNN.[35] The authors analysed 10-s ECG recordings of 42 patients receiving dofetilide, another I_{Kr} blocker antiarrhythmic drug, or placebo. In their experiments, they used the data from two distinct prospective randomized controlled trials available in the PhysioNet repository[36] and found that their CNN model was superior to QTc alone in predicting plasma dofetilide concentration. However, the database used in their study was relatively small (dozens of patients) and they did not use cross-validation in training, with the well-known risk of overfitting. Furthermore, they could not assess the capacity of their artificial intelligence model to detect an arrhythmic risk, or cLQTS, and interpretability of their findings was not performed (*Figure 8*) as done in the present study.

Other studies have focused on CNN modelling of other cardiac diseases using multilead ECG input. For the detection of anterior

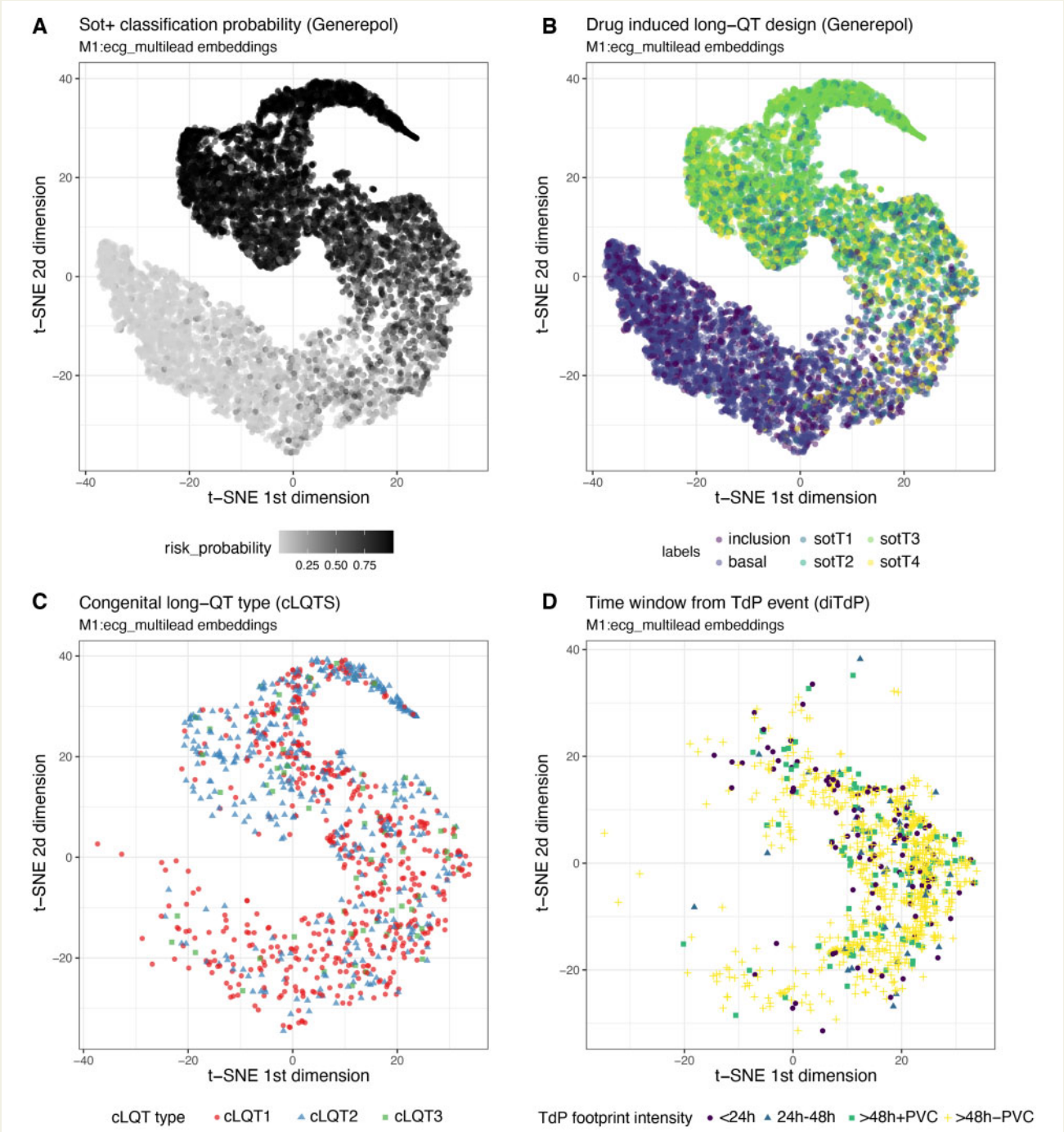


Figure 7 Risk prediction model's deep embeddings reveal clinically relevant data structure. All panels illustrate two dimensions after a t-SNE transformation of the 512 dimensions of the multilead M1 model embeddings (see 'Methods'). (*A*) t-SNE map where each point represents an electrocardiogram from the Generepol cohort. Greyscale indicates the M1: ecg_multilead classification score ranging in [0:1]. (*B*) Same t-SNE map as (*A*) where electrocardiograms are coloured by the experimental setup, from inclusion before and 1, 2, 3, or 4 h after sotalol intake. (*C*) Same t-SNE map with electrocardiogram from the congenital long-QT syndrome cohort. Electrocardiograms are annotated by congenital long-QT syndrome (cLQTS) type. (*D*) Same t-SNE map as (*A*) with electrocardiogram from the drug-induced torsade de pointes (diTdP) cohort of patients having experienced at least one drug-induced torsade de pointes event. Electrocardiograms are coloured by the four groups of torsade de pointes intensity footprint (timeframe from the drug-induced torsade de pointes event and presence/absence of PVCs on the electrocardiogram).

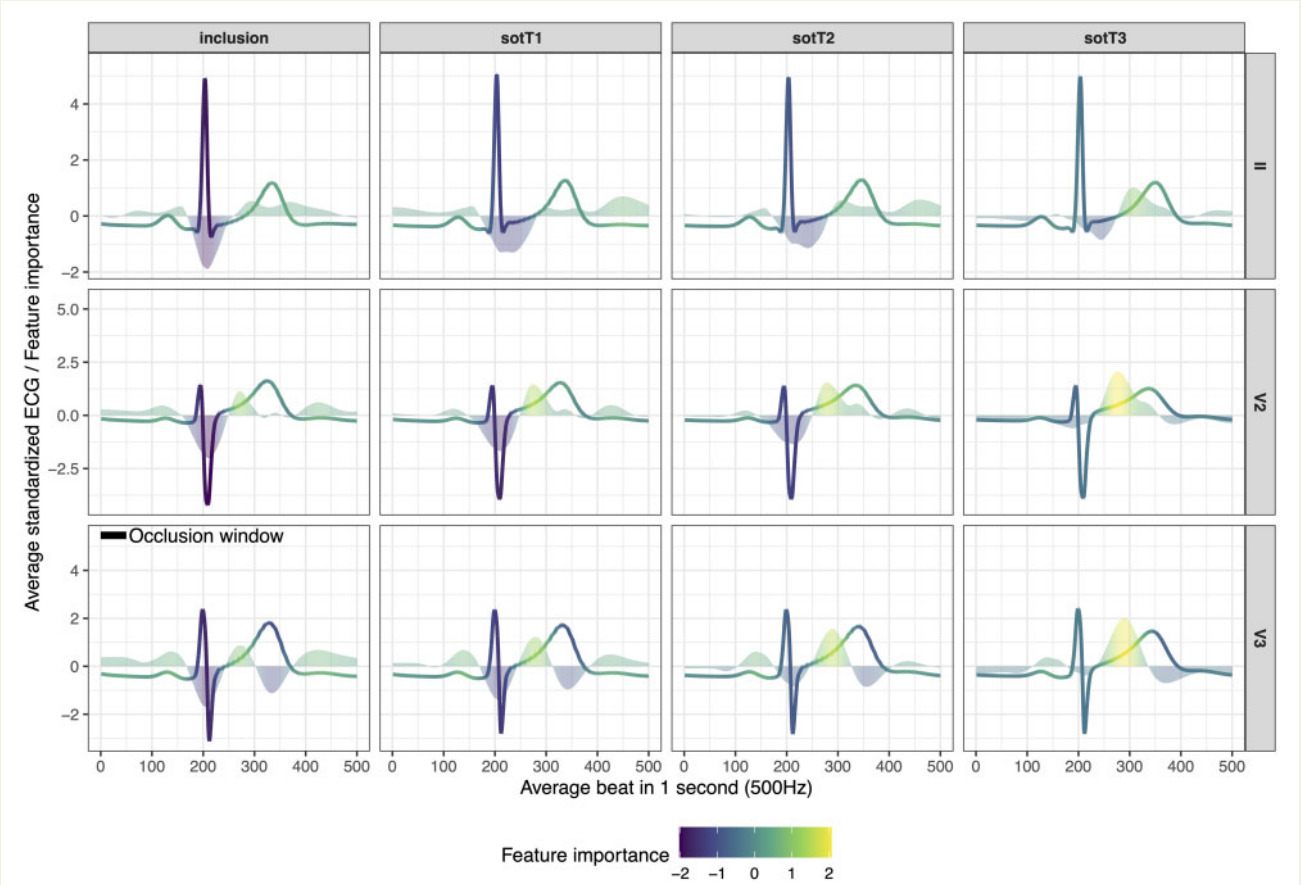


Figure 8 Interpretability of the sotalol footprint on the electrocardiogram signal. This figure displays an averaged signal of the standardized electrocardiogram for each segmented beat for leads LII, V2, and V3. All signals from the same time points were analysed together. Similarly, the standardized feature importance profile is summarized and laid behind the electrocardiogram profile. Colours for both the electrocardiogram and FIP indicate intensity of the feature importance profile.

myocardial infarction,[37] Liu *et al.* used a 4-lead approach that led to accuracies >90% with a five-fold cross-validation. Tison *et al.*[38] created a CNN-hidden Markov model that took 12-lead input in order to detect pulmonary arterial hypertension, hypertrophic cardiomyopathy, cardiac amyloid, and mitral valve prolapse. The ROC-AUC was in the 77–94% range for the four conditions. Similar technology was also used by Attia *et al.*,[39] who applied a CNN model on a large database (>97 000 patients) to detect left ventricular dysfunction. They used a large holdout set (>52 000 patients) and achieved an overall accuracy of 86%. Moreover, a subset of the patients, which were erroneously classified as having ventricular dysfunction, later developed a low ejection fraction, suggesting that the model was able to detect features of this condition before it became clinically diagnosed. Unfortunately, the healthy volunteers from Generepol misclassified by our model as taking sotalol before any intake were not followed, precluding any evaluation of their subsequent risk for TdP and sudden death.

We introduced CNN models trained with data obtained from one lead only. They were as accurate as the multilead model not only when classifying holdout data from the same leads but also from leads on which they were not trained. This is an unexpected result and indicates that the sotalol footprint is detected by all leads and in similar ways, with the exception of lead V1. Moreover, ECG data were recorded with different acquisition devices and some ECGs, recorded in 250 Hz, were upsized using interpolation techniques. Still, the results were robust, regardless of the recording device. This paves the way to clinical applications where the patients or physicians could record a single electrode ECG, which could then be sent to a centralized server and analysed by the CNN models, with the goal of stratifying the risk for the patient to develop a TdP.

We also explored the CNN models to understand how the decision process was made and what was the model looking for in the ECG to provide a prediction. The occlusion-based interpretability algorithm uncovered the sotalol ECG footprint, which changed with time as blood sotalol concentration increased. The analysis of the footprint was consistent with existing knowledge on how sotalol influences cardiomyocyte action potential, mainly through blockade of I_{Kr} and beta-adrenergic receptor blockade. This approach opens novel avenues of research and applications in the context of drug monitoring for the pharmaceutical industry but also plays an important role in the acceptability of artificial intelligence in clinics. Providing an explanation for the prediction process is increasingly requested

when not mandatory,[40,41] especially for 'black boxes' such as deep neural networks, which train millions of parameters. J-Tpeak features emerged as the main attribute allowing for discrimination of sotalol intake. This is concordant with the emerging literature on the importance of this specific segment when predicting for diTdP beyond QTc.[42]

Lastly, we demonstrated that besides risk prediction, the CNN models learn clinically relevant knowledge. A post hoc analysis of the network's deep embeddings grouped ECGs from the studied cohorts according to their clinical relevance (*Figure 7*). These embeddings can be used to automatically stratify ECG, and ultimately patients, in novel classes that are yet to be characterized. However, identifying the best embeddings can be challenging since the number of model architectures to explore can be very large. More research and training data are needed in the context of translational clinical applications of CNN models for the diagnosis of the different types of cLQTS and prediction of diTdP.

Supplementary material

Supplementary material is available at *European Heart Journal* online.

Funding

This study was partly funded by the French Research Agency (2021–24): Agence nationale de la recherche (ANR) DeepECG4U project. Further validation of the algorithm developed herein is planned in real-life patients prospectively included with various age, gender, ethnicity, geographic location, and cardiovascular risk factors within the ANR funded DeepECG4U project.

Conflict of interest: A.L. reports consultancy and travel grants from SANOFI. A.L., A.P., J.-E.S., E.P., J.-D.Z., C.F.-B., and F.E. report a patent pending for detecting the risk of torsade de pointes (EP19305730.4, WO/2020/245322). G.D. reports a postdoctoral fellowship funding (18SFRN34110369). Other authors have nothing to disclose.

Data availability

Data and models used herein are available from the corresponding authors, upon reasonable request.

References

1. Dessertenne F. Ventricular tachycardia with 2 variable opposing foci. *Arch Mal Coeur Vaiss* 1966;**59**:263–272.
2. Rosso R, Hochstadt A, Viskin D, Chorin E, Schwartz AL, Tovia-Brodie O, Laish-Farkash A, Havakuk O, Gepstein L, Banai S, Viskin S. Polymorphic ventricular tachycardia, ischaemic ventricular fibrillation, and torsade de pointes: importance of the QT and the coupling interval in the differential diagnosis. *Eur Heart J* 2021. doi:10.1093/eurheartj/ehab138
3. Stramba-Badiale M, Karnad DR, Goulene KM, Panicker GK, Dagradi F, Spazzolini C, Kothari S, Lokhandwala YY, Schwartz PJ. For neonatal ECG screening there is no reason to relinquish old Bazett's correction. *Eur Heart J* 2018;**39**:2888–2895.
4. Roden DM. Drug-induced prolongation of the QT interval. *N Engl J Med* 2004;**350**:1013–1022.
5. Yang T, Chun YW, Stroud DM, Mosley JD, Knollmann BC, Hong C, Roden DM. Screening for acute IKr block is insufficient to detect torsades de pointes liability: role of late sodium current. *Circulation* 2014;**130**:224–234.
6. Schwartz PJ, Ackerman MJ, Antzelevitch C, Bezzina CR, Borggrefe M, Cuneo BF, Wilde AAM. Inherited cardiac arrhythmias. *Nat Rev Dis Primers* 2020;**6**:58.
7. Itoh H, Crotti L, Aiba T, Spazzolini C, Denjoy I, Fressart V, Hayashi K, Nakajima T, Ohno S, Makiyama T, Wu J, Hasegawa K, Mastantuono E, Dagradi F, Pedrazzini M, Yamagishi M, Berthet M, Murakami Y, Shimizu W, Guicheney P, Schwartz PJ, Horie M. The genetics underlying acquired long QT syndrome: impact for genetic screening. *Eur Heart J* 2016;**37**:1456–1464.
8. Salem JE, Yang T, Moslehi JJ, Waintraub X, Gandjbakhch E, Bachelot A, Hidden-Lucet F, Hulot JS, Knollmann BC, Lebrun-Vignes B, Funck-Brentano C, Glazer AM, Roden DM. Androgenic effects on ventricular repolarization: a translational study from the international pharmacovigilance database to iPSC-cardiomyocytes. *Circulation* 2019;**140**:1070–1080.
9. Schwartz PJ, Woosley RL. Predicting the unpredictable: drug-induced QT prolongation and torsades de pointes. *J Am Coll Cardiol* 2016;**67**:1639–1650.
10. Salem JE, Nguyen LS, Moslehi JJ, Ederhy S, Lebrun-Vignes B, Roden DM, Funck-Brentano C, Gougis P. Anticancer drug-induced life-threatening ventricular arrhythmias: a World Health Organization pharmacovigilance study. *Eur Heart J* 2021:ehab362. doi: 10.1093/eurheartj/ehab362.
11. Saque V, Vaglio M, Funck-Brentano C, Kilani M, Bourron O, Hartemann A, Badilini F, Salem JE. Fast, accurate and easy-to-teach QT interval assessment: the triplicate concatenation method. *Arch Cardiovasc Dis* 2017;**110**:475–481.
12. Moss AJ, Zareba W, Benhorin J, Locati EH, Hall WJ, Robinson JL, Schwartz PJ, Towbin JA, Vincent GM, Lehmann MH, Keating MT, MacCluer JW, Timothy KW. ECG T-wave patterns in genetically distinct forms of the hereditary long QT syndrome. *Circulation* 1995;**92**:2929–2934.
13. Schwartz PJ, Ackerman MJ. The long QT syndrome: a transatlantic clinical approach to diagnosis and therapy. *Eur Heart J* 2013;**34**:3109–3116.
14. Salem JE, Bretagne M, Lebrun-Vignes B, Waintraub X, Gandjbakhch E, Hidden-Lucet F, Gougis P, Bachelot A, Funck-Brentano C; French Network of Regional Pharmacovigilance Centres. Clinical characterization of men with long QT syndrome and torsades de pointes associated with hypogonadism: a review and pharmacovigilance study. *Arch Cardiovasc Dis* 2019;**112**:699–712.
15. Salem JE, Germain M, Hulot JS, Voiriot P, Lebourgeois B, Waldura J, Tregouet DA, Charbit B, Funck-Brentano C. GENomE wide analysis of sotalol-induced IKr inhibition during ventricular REPOLarization. "Generepol Study": lack of common variants with large effect sizes. *PLoS One* 2017;**12**:e0181875.
16. U.S. Department of Health and Human Services Food and Drug Administration. *E14 Clinical Evaluation of QT/QTc Interval Prolongation and Proarrhythmic Potential for Non-Antiarrhythmic Drugs—Questions and Answers (R3)*. Guidance for Industry; 2017. https://www.fda.gov/files/drugs/published/E14-Clinical-Evaluation-of-QT-QTc-Interval-Prolongation-and-Proarrhythmic-Potential-for-Non-Antiarrhythmic-Drugs-Questions-and-Answers-%28R3%29-Guidance-for-Industry.pdf (11 August 2021).
17. Drew BJ, Ackerman MJ, Funk M, Gibler WB, Kligfield P, Menon V, Philippides GJ, Roden DM, Zareba W; American Heart Association Acute Cardiac Care Committee of the Council on Clinical Cardiology, the Council on Cardiovascular Nursing, and the American College of Cardiology Foundation. Prevention of torsade de pointes in hospital settings: a scientific statement from the American Heart Association and the American College of Cardiology Foundation. *Circulation* 2010;**121**:1047–1060.
18. Obermeyer Z, Emanuel EJ. Predicting the future—big data, machine learning, and clinical medicine. *N Engl J Med* 2016;**375**:1216–1219.
19. LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature* 2015;**521**:436–444.
20. Hinton G. Deep learning—a technology with the potential to transform health care. *JAMA* 2018;**320**:1101–1102.
21. Bos JM, Attia ZI, Albert DE, Noseworthy PA, Friedman PA, Ackerman MJ. Use of artificial intelligence and deep neural networks in evaluation of patients with electrocardiographically concealed long QT Syndrome from the surface 12-lead electrocardiogram. *JAMA Cardiol* 2021;**6**:532–538.
22. Salem JE, Dureau P, Bachelot A, Germain M, Voiriot P, Lebourgeois B, Tregouet DA, Hulot JS, Funck-Brentano C. Association of oral contraceptives with drug-induced QT interval prolongation in healthy nonmenopausal women. *JAMA Cardiol* 2018;**3**:877–882.
23. Extramiana F, Badilini F, Sarapa N, Leenhardt A, Maison-Blanche P. Contrasting time- and rate-based approaches for the assessment of drug-induced QT changes. *J Clin Pharmacol* 2007;**47**:1129–1137.
24. Sarapa N, Morganroth J, Couderc JP, Francom SF, Darpo B, Fleishaker JC, McEnroe JD, Chen WT, Zareba W, Moss AJ. Electrocardiographic identification of drug-induced QT prolongation: assessment by different recording and measurement methods. *Ann Noninvasive Electrocardiol* 2004;**9**:48–57.
25. Extramiana F, Badilini F, Denjoy I, Vaglio M, Green CL, Kligfield P, Leenhardt A, Maison-Blanche P. Sex influences on ventricular repolarization duration in normal subjects and in type 1, 2 and 3 long QT syndrome patients: different effect in acquired and congenital type 2 LQTS. *J Electrocardiol* 2020;**62**:148–154.
26. Baillet S, Friston K, Oostenveld R. Academic software applications for electromagnetic brain mapping using MEG and EEG. *Comput Intell Neurosci* 2011;**2011**:972050.
27. Su C, Tong J, Zhu Y, Cui P, Wang F. Network embedding in biomedical data science. *Brief Bioinform* 2020;**21**:182–197.
28. Van der Maaten L, Hinton G. Visualizing data using t-SNE. *J Mach Learn Res* 2008;**9**:2579–2605.
29. Nguyen LS, Dolladille C, Drici MD, Fenioux C, Alexandre J, Mira JP, Moslehi JJ, Roden DM, Funck-Brentano C, Salem JE. Cardiovascular toxicities associated

with hydroxychloroquine and azithromycin: an analysis of the World Health Organization Pharmacovigilance Database. *Circulation* 2020;**142**:303–305.
30. Salem JE, Alexandre J, Bachelot A, Funck-Brentano C. Influence of steroid hormones on ventricular repolarization. *Pharmacol Ther* 2016;**167**:38–47.
31. Hondeghem LM. Drug-induced QT prolongation and torsades de pointes: an all-exclusive relationship or time for an amicable separation? *Drug Saf* 2018;**41**: 11–17.
32. Funck-Brentano C. Pharmacokinetic and pharmacodynamic profiles of d-sotalol and d,l-sotalol. *Eur Heart J* 1993;**14**:30–35.
33. Haverkamp W, Breithardt G, Camm AJ, Janse MJ, Rosen MR, Antzelevitch C, Escande D, Franz M, Malik M, Moss A, Shah R. The potential for QT prolongation and pro-arrhythmia by non-anti-arrhythmic drugs: clinical and regulatory implications. Report on a Policy Conference of the European Society of Cardiology. *Cardiovasc Res* 2000;**47**:219–233.
34. Porta-Sanchez A, Spillane DR, Harris L, Xue J, Dorsey P, Care M, Chauhan V, Gollob MH, Spears DA. T-wave morphology analysis in congenital long QT syndrome discriminates patients from healthy individuals. *JACC Clin Electrophysiol* 2017;**3**:374–381.
35. Attia ZI, Sugrue A, Asirvatham SJ, Ackerman MJ, Kapa S, Friedman PA, Noseworthy PA. Noninvasive assessment of dofetilide plasma concentration using a deep learning (neural network) analysis of the surface electrocardiogram: a proof of concept study. *PLoS One* 2018;**13**:e0201059.
36. Moody GB, Mark RG, Goldberger AL. PhysioNet: a research resource for studies of complex physiologic and biomedical signals. *Comput Cardiol* 2000;**27**: 179–182.
37. Liu W, Zhang M, Zhang Y, Liao Y, Huang Q, Chang S, Wang H, He J. Real-time multilead convolutional neural network for myocardial infarction detection. *IEEE J Biomed Health Informatics* 2018;**22**:1434–1444.
38. Tison GH, Zhang J, Delling FN, Deo RC. Automated and interpretable patient ECG profiles for disease detection, tracking, and discovery. *Circ Cardiovasc Qual Outcomes* 2019;**12**:e005289.
39. Attia ZI, Kapa S, Lopez-Jimenez F, McKie PM, Ladewig DJ, Satam G, Pellikka PA, Enriquez-Sarano M, Noseworthy PA, Munger TM, Asirvatham SJ, Scott CG, Carter RE, Friedman PA. Screening for cardiac contractile dysfunction using an artificial intelligence-enabled electrocardiogram. *Nat Med* 2019;**25**:70–74.
40. Martens D, Vanthienen J, Verbeke W, Baesens B. Performance of classification models from a user perspective. *Decis Support Syst* 2011;**51**:782–793.
41. Goodman B, Flaxman S. Union regulations on algorithmic decision-making and a "right to explanation". *AI Mag* 2017;**38**:50–57.
42. Vicente J, Zusterzeel R, Johannesen L, Mason J, Sager P, Patel V, Matta MK, Li Z, Liu J, Garnett C, Stockbridge N, Zineh I, Strauss DG. Mechanistic model-informed proarrhythmic risk assessment of drugs: review of the "CiPA" initiative and design of a prospective clinical validation study. *Clin Pharmacol Ther* 2018; **103**:54–66.

Chapter 3

Interpretability of Neural Networks: ECG Applications

Chapter Summary

3.1 Introduction

Electrocardiogram (ECG) analysis is a crucial diagnostic tool for identifying cardiac abnormalities and monitoring the heart's electrical activity. Deep learning techniques have increasingly been employed for ECG analysis, exhibiting high accuracy and performance. However, understanding the decision making process of these complex models remains challenging. In our previous work we proposed and developped a deep learning approach to predict the risk of TdP occurence. The neural network was valdiated with clinicians. However, for them to adopt the method in their practice, they must trust the predictions made by the model.

To better understand the input data as well as the model, we performed neural network interpretability. Neural network interpretability refers to the extent to which the internal workings, decision making process, and output of a neural network model can be understood, explained, and justified. In essence, interpretability aims to make the "black box" nature of complex neural networks more transparent and accessible to humans, facilitating trust and confidence in the model's predictions and enabling more informed decision making. When this process is transparent and comprehensible, clinicians, in general, can feel more confident about using deep learning models to support their diagnoses and treatment plans. Moreover, as patients become increasingly involved in their healthcare, interpretability can facilitate better communication between physicians and patients, enabling more informed decision making. Beyond trust and confidence, one must consider ethical considerations. In fact, the ethical implications of using deep learning models in medical applications cannot be over-

looked. Ensuring that the model's decision making process is transparent and free from biases is essential for upholding ethical standards.

Interpretability allows researchers and practitioners to examine the underlying factors that contribute to predictions, ensuring that the model is not discriminating against specific populations or perpetuating existing biases. It also allows them to assess how well a model generalizes across diverse data sets, ensuring its robustness and reliability in real world clinical settings. Moreover, interpretability can expose potential weaknesses, biases, or errors in the model, enabling researchers to fine tune it and enhance its performance. This, in turn, can lead to more accurate and reliable ECG analysis and ultimately improved patient outcomes. Indeed, models must be capable of handling a wide range of variations in data, including noise, different morphologies, and various patient populations.

In our previous paper [46], we explored the predictions of our model by interpreting its outputs with respect to the input ECGs. We used one of the state-of-the-art methods, occlusion, based on the concept of signal perturbation. While the occlusion interpretability provided valuable insights into the model's decision making process as well relevant features from input ECGs, it also presented inherent limitations that can restrict its applicability and usefulness. Factors such as sensitivity to occlusion size and shape, lack of causality, lack of consideration for feature inter-dependencies and focus on local interpretability, need to be considered when employing this method.

In this chapter we propose a novel original approach for neural interpretability as an evolution of the occlusion approach with aims of overcoming its limits. The approach is multivariate and explores simultaneously different regions of the input data (let it be ECG signals or images). The problem is NP complex and needs heuristics to be solved. We implemented a genetic algorithm to approximate the problem. Moreover, we developed the method as an extensible framework package for further improvements. We tested this method, named **evocclusion** on the Generepol dataset described in the previous chapter and compared it with standard occlusion and other state-of-the-art methods, including saliency maps [50].

3.2 State of the art

3.2.1 Limits of occlusion interpretability method

Occlusion based interpretability technique has emerged as one promising approach to shed light on the inner workings of neural networks, particularly in the context of image or sequences classification tasks. However, despite its potential to provide valuable insights, occlusion is not without its limitations. By discussing these limitations, we aim to provide a comprehensive understanding of the challenges associated with occlusion based interpretability and highlight areas our new method strives to improve.

Sensitivity to Occlusion Size and Shape Occlusion relies on systematically occluding (i.e. multiplying by zero) parts of the input data with the objective to evaluate the importance of these parts of the data in the final output. This allows to understand the neural network's response to these perturbations. However, these methods are sensitive to the choice of occlusion size and shape, which can impact significantly the results. A poorly chosen occlusion size may result in either too much or too little information being occluded, misleading the interpretation of the model's behavior. Further, occlusion shapes may not adequately capture the relevant features, thus limiting the interpretability of the results.

Lack of Causality Occlusion methods provide a measure of the neural network's sensitivity to input perturbations but do not necessarily establish causal relationships between input features and model predictions. While the occlusion of a specific input region may lead to changes in the model's output,

it does not guarantee that this region is causally responsible for the prediction. This limitation makes it challenging to draw definitive conclusions about the model's decision making process.

Local Interpretability vs. Global Understanding The state of the art occlusion focuses on local interpretability, aiming to understand how specific input regions influence the model's predictions. However, this focus on local interpretability may not provide a complete understanding of the neural network's global behavior. In some cases, understanding the interactions between different features or the higher level abstractions learned by the model is crucial for gaining a comprehensive understanding of the model's decision making process. As a result, occlusion methods may not always provide sufficient insight into the model's overall functioning.

Univariate Problem and Limited Occlusion Strategies The occlusion interpretability methods typically occlude one region at a time, resulting in a univariate analysis of the input features. This approach might not capture the complex interactions between different regions or features, limiting the insights that can be gained from the occlusion process. In many cases, understanding how multiple features or regions interact to influence the model's predictions is essential for a comprehensive interpretation of the model's behavior.

Moreover, most occlusion techniques rely on replacing the occluded region with a constant value, such as zero or the mean pixel value. This replacement strategy might not always yield the most informative results, as it does not consider alternative scenarii where the occluded region could be replaced with meaningful information. For example, replacing the occluded part with information from the opposite target class or other contextually relevant information might provide a more nuanced understanding of the model's decision making process. Another significant limitation of occlusion is the way the signal is occluded, in fact replacing part of the data with zeroes or mean value can lead to out-of-distribution scenario where the model has not been trained on such data and therefore might give random, inaccurate and uninterpretable results.

By exploring more sophisticated occlusion strategies that account for multivariate interactions and utilize meaningful replacements, it may be possible to derive more insightful interpretations of neural network behavior. However, it is important to note that these advanced occlusion techniques could further increase computational complexity and exacerbate existing challenges associated with other occlusion based interpretability methods.

3.2.2 Interpretability methods

Explainable Artificial Intelligence (XAI) has emerged as a rapidly evolving research area dedicated to developing techniques that can provide insights into the decision making processes of these models, making them more transparent, interpretable, and trustworthy. The primary goal of XAI is to provide human understandable explanations for the predictions and decisions made by AI and ML models. These explanations can help users understand the rationale behind the model's predictions, validate its behavior, identify potential biases or limitations, and ultimately build trust in the system. XAI can be considered as multifaceted concept encompassing various dimensions that contribute to a comprehensive understanding of neural network behavior. These dimensions as described in Figure 19, include the dichotomy between passive and active approaches to interpretability, the diverse types of explanations that can be generated to describe model behavior, and the spectrum of local to global interpretability. By examining these dimensions, we can develop a deeper appreciation of the factors influencing model decisions, identify potential biases, and establish more transparent and trustworthy deep learning systems.

Types of explanations

The need for interpretable and explainable AI has become increasingly important, ensuring that models are transparent, accountable, and trustworthy. It requires a diverse set of explanation techniques

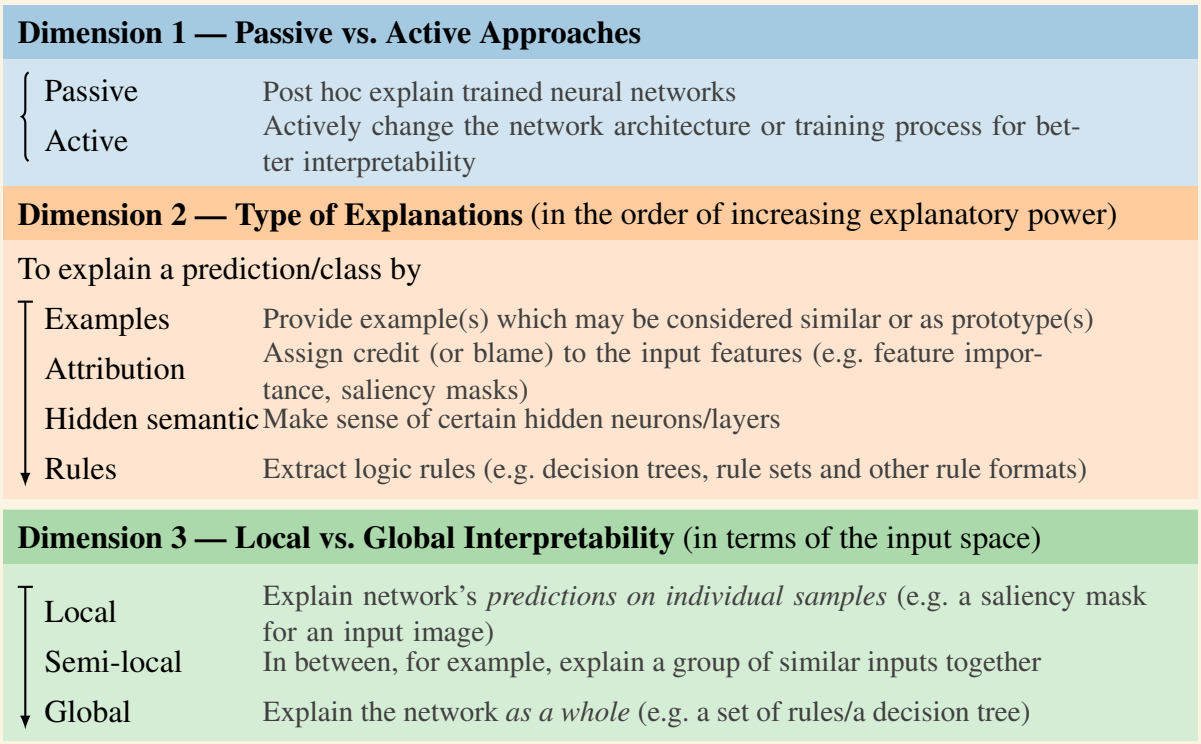


Figure 19: ***The 3 dimensions of interpretability taxonomy*** *from Yu Zhang et al., A Survey on Neural Network Interpretability [3]*

that cater to the varying needs of stakeholders, such as domain experts, practitioners, and end users. One crucial aspect of interpretability is the ability to provide meaningful explanations of a model's predictions, which can help build trust, facilitate communication with domain experts, and ensure that the model is reliable and robust. To address this need, multitude explanation techniques have been proposed that enable different perspectives on understanding the model's behavior.

Example based explanations Example based explanations leverage representative samples from the dataset to provide insights into the model's behavior, allowing users to understand the relationships between input features and model predictions. One approach to example-based-explanations is identifying *prototypes* and *criticisms* [51]. *Prototypes* are instances from the dataset that are representative of the model's learned concepts or classes, while criticisms are instances that contradict or challenge the model's understanding. Examining these examples can reveal insights into the model's decision-making process and uncover potential biases or shortcomings. In the context of ECG analysis, prototypes and criticisms can help identify characteristic patterns of cardiac arrhythmia and reveal atypical cases that may require further investigation. ECG prototypes can present specific variations on the signals overtime and between different classes. Prototypes and criticisms provide insights into the behavior of neural networks by presenting representative samples from the dataset, a subpartition of the dataset, class or population. In the context of electrocardiogram analysis, prototypes can reveal the common ECG waveform patterns associated with specific conditions, such as LongQT, Torsades de Pointes arrhythmia, atrial fibrillation, ventricular tachycardia, or myocardial infarction. These prototypical examples can be used as a reference for clinicians, assisting them in understanding the model's decisions and validating its diagnostic capabilities. Criticisms, on the other hand, are instances that deviate from the expected patterns, providing insights into the model's limitations and areas where improvements can be made. Identifying criticisms in ECG analysis helps detect cases where the model may struggle to differentiate between similar arrhythmia or misclassify noisy or artifact laden signals. These insights can guide further model refinement and development, ensuring that the model is robust and reliable in various clinical scenarii. In 2019, Ming et al proposed ProSeNet

[52], an interpretable and steerable deep sequence model with natural explanations derived from case based reasoning. The prediction is obtained by comparing the inputs to a few prototypes, which are exemplar cases in the problem domain. For better interpretability, several criteria for constructing the prototypes are defined, including simplicity, diversity, and sparsity. ProSeNet also provides a user friendly approach to model steering: domain experts without any knowledge on the underlying model or parameters can easily incorporate their intuition and experience by manually refining the prototypes. Y. Wu and C. Lian, in 2022 [53], introduced another approach based on transformers architecture and prototypes for ECG classification. The use of a transformer architecture enhanced the model parallelism via an attention mechanism, overcoming the limitations of recurrent neural networks that are unable to parallelize while also possessing strong feature extraction capabilities. Their network utilizes a multi channel feature map cutting approach for ECG classification. After processing the raw univariate ECG using dilated causal convolutions, they divided the feature map into several non overlapping patches, encoding the patch set as a sequence of tokens. This allows them to obtain correlations between patches in parallel, utilizing prototype learning to enhance the model's performance. Indeed, prototypes are shown to be very useful to evaluate how close a sample is from a general population of samples, they can also be used to evaluate interpretability methods by computing a distance between identified relevant parts of the input data and the differences highlighted in prototypes. [54] also proposed a similar approach without transformers to classify ECG heartbeats using model agnostic explanations, they also provided a review of existing interpretability methods, specifically in the field of ECGs. Another noteworthy research work is [55], which presents a novel approach to explain the classification of two dimensional time-series data using deep learning models. The authors proposed to learn stereotypical representations or "prototypes" from the latent space induced by the models, and use them to understand the model's algorithmic behavior. The prototypes are optimized for diversity and robustness, and can capture meaningful features of the data that distinguish different classes. The authors applied their method to three domains: electrocardiogram (ECG) wave-forms to detect clinical bradycardia in preterm infants, respiration wave-forms to detect apnea of prematurity, and audio wave-forms to classify spoken digits. They show that the prototypes can provide explainable insights into the classification tasks, such as identifying bradycardia in ECG, apnea in respiration, and articulation in speech. The authors also visualized how the prototypes were used by the models to make predictions, and how they varied across sub-classes. The paper demonstrates that the learned prototypical framework can produce interpretable and faithful explanations for deep classification of time-series data.

Another example-based explanation method is the generation of *counterfactual explanations* [56], as proposed by Wachter et al. (2017) [57]. Counterfactuals are hypothetical instances that are minimally different from a given input but result in a different prediction. Counterfactual explanations offer a unique perspective on neural network prediction process by revealing how minimal changes in input features can lead to different predictions. By understanding the model's decision boundaries and pinpointing the specific features that contribute to different outcomes, counterfactual explanations can facilitate a deeper understanding of the model's reasoning process and support more informed decision making in various applications, including ECG analysis. In the context of ECG analysis, counterfactual explanations can be particularly valuable for distinguishing between subtle differences in ECG wave-forms that correspond to different cardiac conditions. For instance, considering a scenario where the model classifies an ECG waveform as ventricular tachycardia. By generating a counterfactual explanation, one can identify the minimal changes required in the ECG signal to yield a different diagnosis, such as supraventricular tachycardia in this case. By examining these differences, clinicians can gain insights into the model's rationale for the original diagnosis and develop a better understanding of the distinguishing features between the two conditions. Moreover, counterfactual explanations can be instrumental in identifying potential limitations and biases within the model. By analyzing the model's decision boundaries, one can uncover areas where the model may be overly sensitive to specific features or prone to misclassification. For example, if the counterfactual explanations reveal that the model frequently misclassifies noisy ECG signals or struggles to differentiate between sim-

ilar arrhythmia, these insights can guide further model refinement and improvement. Furthermore, counterfactual explanations can play a crucial role in enhancing the communication between deep learning models and domain experts. By offering a clear understanding of the specific features that lead to different outcomes, counterfactual explanations can facilitate more effective collaboration between AI researchers and cardiologists. This improved understanding and communication can result in the development of more accurate and clinically relevant models for ECG analysis. There are several ways when it comes to generating counterfactuals. In their paper [58], Peiyu Li et *al.* proposed a novel method for generating counterfactual explanations for time series models using motifs. Motifs are recurring patterns in time series data that capture important features or events. The authors propose Motif-Guided Counterfactual Explanation (MG-CF), a model that leverages motifs to guide the generation of minimal and intuitive counterfactuals that can flip the model's decision. The main idea is to identify the most influential motifs in the original time series and replace them with alternative motifs from the opposite class. The authors formulate this as an optimization problem and propose an efficient algorithm to solve it. They also introduce several metrics to evaluate the quality of the counterfactuals, such as sparsity, label flip rate, L1 distance, and number of independent segments. They conduct experiments on six real world time series datasets (including ECGs) from the UCR (UCR Time Series Archive) repository and compare their method with three state-of-the-art baselines. The results show that MG-CF outperforme the baselines in terms of balancing all the desirable properties of counterfactual explanations. Their work also provides some illustrative examples of how MG-CF can help users understand and trust time series models better.

Attribution based explanations Attribution based explanations aim to quantify the contribution of each input feature to the model's prediction, allowing stakeholders to understand the factors driving model outcomes and ensure the model relies on meaningful features for its predictions. In the context of ECG analysis, these methods can help identify the most critical segments of the ECG waveform (e.g., P wave, QRS complex, T wave) that contribute to the model's diagnosis. One of the most well-known attribution-based explanation methods is Integrated Gradients, proposed by Sundararajan et al. int 2017 [59]. This technique assigns importance scores to individual features by approximating the integral of the gradients of the model's output concerning the input along a straight path from a baseline instance to the input instance as shown in Figure 20. The Integrated Gradients method has been used in the ECG domain to identify the most influential segments of ECG signals responsible for the model's predictions. For instance, Hannun et al. in 2019 [60] employed Integrated Gradients in their deep learning based ECG arrhythmia classification model, enabling them to identify key ECG features that contributed to the model's diagnoses. Another popular attribution based method is Local Interpretable Model-agnostic Explanations (LIME), introduced by Ribeiro et al. in 2016 [61]. LIME computes locally faithful explanations by approximating the model's behavior with a linear model around a specific input instance. LIME has been applied to various types of models and data, including ECG analysis, to help clinicians understand the model's decision-making process for individual patients by highlighting the most influential ECG features for each case. In a study by Yao et al. [62], LIME was used to explain the predictions of a deep learning model for detecting myocardial infarction from ECG signals. The authors demonstrated that LIME could highlight the relevant segments of the ECG waveform, making the model's predictions more interpretable for clinicians. They presented an interpretable multi-class ECG classification method and heartbeat anomaly detection based on deep learning. Their research aimed to address the interpretability challenges in ECG analysis using deep learning models, particularly in providing meaningful explanations for model predictions. The authors developed a deep learning model for 12-leads ECG classification and employed the LIME technique to generate interpretable explanations for the model's predictions. By doing so, they identified the critical segments of the ECG waveform that contributed to the model's classification decision, such as P wave, QRS complex and T wave. The results demonstrated that LIME could provide meaningful explanations, allowing clinicians to better understand and trust the model's predictions. Layer-wise Relevance Propagation (LRP), proposed by Bach et al. in 2015 [63], is another attribution based method that back-propagates the model's output through the layers of

the neural network to compute feature importance scores. LRP has also been used in ECG analysis, such as in the work of Strodthoff and Strodthoff [64], who applied LRP to a convolutional neural network for detecting ventricular fibrillation in ECG signals. The authors found that LRP could provide meaningful explanations, highlighting the relevant morphological features of the ECG waveform responsible for the model's predictions.

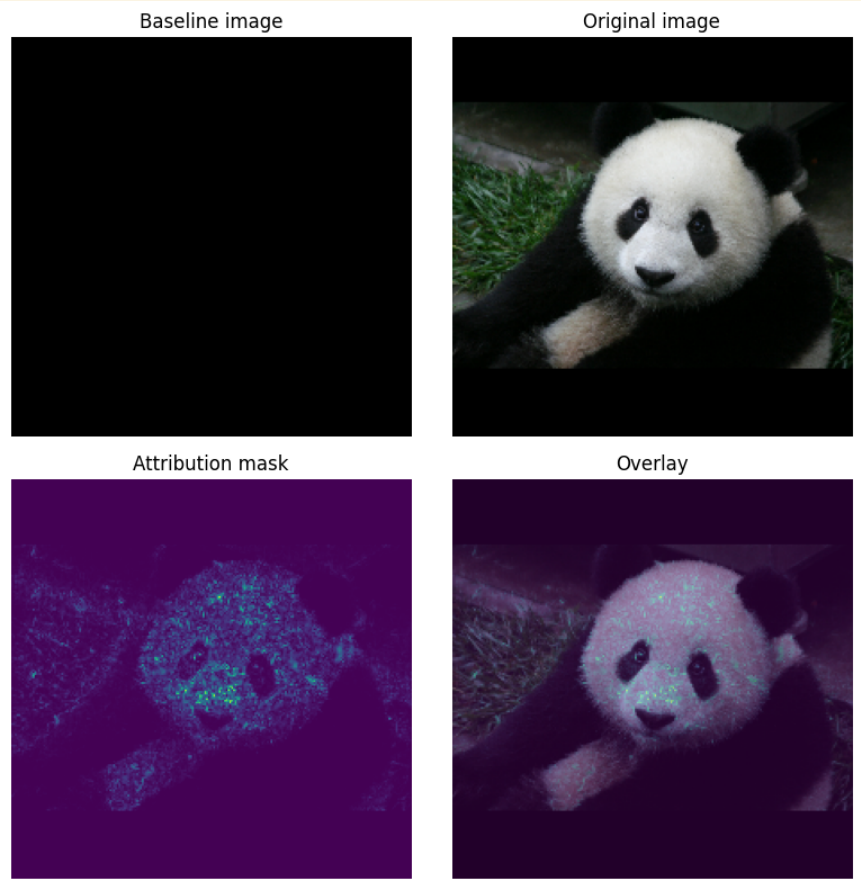


Figure 20: ***Integrated Gradients attributions map example:*** *a path of transformations of the input image is computed with an increasing magnitude given a number of steps from 0 to 1. After applying computing the gradients of each transformed image and calculating the integrated gradients function, attributions map is obtained and highlights relevant part on the input image.*

Semantic explanations Semantic explanations aim to provide human understandable insights into the model's prediction process by connecting the model's internal representations with meaningful concepts or structures. This approach facilitates a more intuitive understanding of the model's reasoning process, as it translates complex mathematical representations into interpretable and contextually relevant information. One notable technique for generating semantic explanations is the use of concept activation vectors (CAVs), as proposed by Kim et al. [65]. Concept activation vectors are derived by first identifying high-level semantic concepts that are relevant to the domain or the problem at hand. These concepts can be either predefined by domain experts or discovered through unsupervised learning techniques, such as clustering or dimensionality reduction. Once these concepts are defined, the next step involves training linear classifiers that can distinguish between the presence or absence of these concepts in the model's internal representations (i.e., activation of neurons or layers). The trained linear classifiers' weights essentially serve as CAVs, representing the direction in the high dimensional activation space that is most indicative of the presence of a particular concept. By projecting the model's internal representations onto these concept activation vectors, users can assess the extent to which specific concepts are activated or utilized by the model for its predictions. This projection can be done by computing the dot product between the internal representations and the

CAVs or by measuring the cosine similarity between them. Semantic explanations through concept activation vectors offer several benefits. Firstly, they enable users to understand the model's behavior in terms of familiar and interpretable concepts, making the reasoning process more accessible and relatable. Secondly, they can help identify potential biases or shortcomings in the model's decision process by revealing over-reliance on specific concepts or ignoring relevant ones. Lastly, CAVs can be used as a diagnostic tool to compare different models or model architectures to assess how well they align with the desired high level concepts. However, it is important to note that semantic explanations using concept activation vectors heavily rely on the chosen high level concepts, and their effectiveness is contingent upon the relevance and interpretability of these concepts. Additionally, CAVs may not always provide a complete picture of the model's decision process, as they primarily focus on linear relationships between the internal representations and the defined concepts. Nonlinear relationships or interactions between features may not be adequately captured by CAVs, necessitating complementary explanation techniques to provide a more comprehensive understanding of the model's behavior. In the context of ECGs several human understandable concepts can be derived such as typical amplitude and length of P, T waves and QRS complex, specific distances such as RR, ST or PR segments, however these concepts can limit the interpretability to concepts given to the method and ignore the internal concepts the model could learn and could be different from human known concepts.

Another semantic explanation method is the extraction of disentangled representations, which aims to separate the underlying factors of variation in the input data. Hinton et al. [66] proposed a deep learning architecture called the "Transforming Autoencoder" that learns disentangled representations by explicitly modeling the transformations between input instances. By learning disentangled representations, the model can provide more interpretable insights into the relationships between input features and the model's internal structures. In ECG applications, disentangled representations can help isolate different aspects of the ECG signal, such as heart rate variability, morphological features, or other clinically relevant factors, enabling a better understanding of the model's decision process.

Rule based explanations Rule-based explanations seek to provide an interpretable description of the model's behavior using a set of human comprehensible rules. These explanations can help understand the model's predictions in a more transparent and digestible format. One approach to rule based explanations is the extraction of decision trees or rule sets from the model, as proposed by Bastani et al. [67]. This technique involves approximating the model's behavior using a decision tree or a set of rules, which can then be used to generate explanations for the model's predictions. In ECG analysis, rule based explanations can help translate the model's complex predictions process into a set of simple rules that mimic the clinical guidelines or heuristics used by cardiologists, making the model's predictions more interpretable and actionable.

Gradients based explanations Gradient based methods, in contrast to attributions based methods, focus on measuring the sensitivity of the model's output to small perturbations in the input features. By computing the gradients of the output with respect to the input features, these techniques can reveal the regions in the input space where the model's predictions are most influenced by the input, providing insights into the factors that the model considers important. Gradient based methods have the advantage of being relatively simple and computationally efficient, as they rely on the model's gradients, which can be computed using standard backpropagation techniques. This makes gradient based methods well suited for large scale models and high dimensional input spaces, where other interpretability techniques may struggle. Furthermore, gradient based methods can provide a more global view of the model's behavior, capturing the overall sensitivity of the output to the input features across the entire input space. This can be particularly valuable in understanding the model's robustness and generalization properties, as well as in identifying potential vulnerabilities, such as adversarial examples or other pathological inputs. One significant method that uses gradients and attributions together is the Integrated Gradients approach we introduced earlier. However, gradient based methods have some limitations. They may be sensitive to noise and can sometimes highlight high frequency patterns or other artifacts that are not meaningful or relevant to human observers.

Additionally, gradient based methods may not always provide accurate or interpretable importance scores, as the gradients can be influenced by the model's architecture and the choice of the loss function.

Passive vs Active Approaches

Interpretability can be approached in two distinct ways: passive and active.

Passive approaches to neural network interpretability focus on analyzing and understanding the model after it has been trained. Visualization techniques form a cornerstone of passive interpretability, as they enable researchers to visually inspect the activation patterns and feature representations within the model. By presenting this information in a visually comprehensible format, researchers can better discern the underlying structures and relationships that govern the model's behavior. There are several visualization techniques that have been proposed for different purposes. One such technique is t-SNE (t-distributed Stochastic Neighbor Embedding) [68], which is used for reducing high-dimensional data to a lower-dimensional space while preserving the original data's structure. This approach, facilitates the identification of clusters and patterns in the data, thereby illuminating the relationships between input features and the model's internal representations. Other visualization methods include activation maximization, class activation maps and Deep Dream

Activation maximization Activation maximization is a visualization technique used to understand the features and patterns learned by a neural network [69]. By generating synthetic inputs that maximize the activation of specific neurons or layers, researchers can gain insights into the model's internal representations and the features that it has learned to recognize. The activation maximization technique involves optimizing an input image to maximize the activation of a target neuron in a specific layer of the neural network. The optimization process typically involves gradient ascent, which iteratively updates the input image by following the gradient of the neuron's activation with respect to the input. By doing so, the algorithm adjusts the input image to activate the target neuron as much as possible. By visualizing the synthetic inputs generated through activation maximization, we can analyze the features and patterns that the CNN has learned to recognize at different layers, thereby gaining a deeper understanding of the model's decision process.

Class activation maps (CAMs) CAMs [70] are another visualization technique used to understand the behavior of neural networks, particularly in the context of image classification tasks. CAMs provide a way to visualize the regions in the input image that contribute the most to a particular class prediction, thereby offering insights into the model's decision process. To generate a class activation map, the model's output for a specific class is computed as a weighted sum of the feature maps from the last convolutional layer, where the weights are determined by the gradient of the class output with respect to the feature maps. This results in a spatial map that highlights the regions in the input image that are most relevant for the class prediction. The class activation map can be visualized by overlaying it on the input image, enabling to see which regions of the image are responsible for the model's classification decision. This can be particularly useful for identifying biases or misclassifications in the model's predictions and can also serve as a valuable debugging tool.

While both activation maximization and class activation maps are visualization techniques for understanding the behavior of neural networks, they serve different purposes and offer different insights. Activation maximization focuses on generating synthetic inputs that maximize the activation of specific neurons, providing insights into the features that the model has learned to recognize at different layers. This technique can help understand the model's internal representations and the hierarchical structure of learned features. In contrast, class activation maps focus on visualizing the regions in the input image that contribute the most to a specific class prediction. This technique offers insights into the model's behavior for individual instances, enabling to identify potential biases or misclassifications in the model's predictions.

Deep Dream Deep Dream [71] is a visualization technique developed by Google researchers that has gained significant attention for its ability to generate striking, artistic images that offer insights into the behavior of neural networks, particularly CNNs. This technique has been widely recognized for its unique approach to understanding the features and patterns that neural networks learn during training. The Deep Dream algorithm is built upon the activation maximization technique. It involves optimizing an input image to maximize the activation of specific neurons or layers in a neural network. However, unlike activation maximization, which typically starts with a random input image, Deep Dream begins with an existing image and enhances the patterns and features that the model recognizes within the image. The algorithm operates by iteratively updating the input image based on the gradients of the neuron's activation with respect to the input. The process of gradient ascent is applied to modify the input image, emphasizing the features that the target neuron responds to. By enhancing these features and patterns, the algorithm generates a "dream-like" image that visualizes the model's internal representations. Deep Dream can be applied to different layers of a neural network, unveiling the hierarchical structure of learned features. For instance, applying the algorithm to lower layers of a CNN might reveal simple patterns and textures, whereas applying it to higher layers can result in more complex, abstract features and object like structures. While activation maximization focuses on generating synthetic inputs that maximize the activation of specific neurons, class activation maps visualize the regions in the input image that contribute the most to a specific class prediction. Deep Dream, on the other hand, emphasizes the features and patterns within an existing image that the model recognizes, generating dream like, artistic visualizations of the model's internal representations.

Integrated Gradients (IG) Integrated Gradients (IG) [59] which combines attributions as well as gradients techniques, aims to provide a principled way of assigning importance scores to input features in deep neural networks. It is designed to satisfy several desirable axioms, including completeness, sensitivity, and implementation invariance. These axioms ensure that the importance scores produced by IG accurately reflect the contributions of each input feature to the model's prediction. The core idea behind IG is to compute the importance of each input feature by integrating the gradients of the model's output with respect to the input features along a straight line path from a baseline input to the actual input. The baseline input is an essential aspect of IG, as it represents a reference point against which the actual input's feature contributions are measured. In practice, the choice of the baseline input depends on the problem domain and can be determined using domain specific knowledge or by averaging over a set of baseline inputs. The algorithm for computing Integrated Gradients involves the following steps which are also detailed in Algorithm 1:

1. Choose a baseline input: Select an appropriate baseline input, which serves as the reference point for measuring feature contributions. This can be a constant input or an average over a set of inputs (e.g: ECGs).

2. Define the straight line path: Construct a straight line path in the input space between the baseline input and the actual input. This path can be parameterized using a scalar variable, where 0 corresponds to the baseline input and 1 corresponds to the actual input.

3. Compute the gradients along the path: Calculate the gradients of the model's output with respect to the input features at each point along the straight line path. This can be done using standard gradient computation techniques, such as backpropagation.

4. Integrate the gradients: Integrate the computed gradients along the straight-line path to obtain the Integrated Gradients for each input feature. This integration can be performed using numerical integration methods, such as the trapezoidal rule or Simpson's rule.

Algorithm 1 Integrated Gradients algorithm

procedure INTEGRATEDGRADIENTS($\mathbf{x}$, F, B, n)
  $\mathbf{x}$baseline $\leftarrow B()$ ▷ Choose baseline input
  IG $\leftarrow \mathbf{0}$ ▷ Initialize Integrated Gradients vector
  for $i \leftarrow 1$ to n **do**
    $\alpha \leftarrow \frac{i}{n}$ ▷ Compute interpolation coefficient
    $\mathbf{x}_\alpha \leftarrow \alpha\mathbf{x} + (1-\alpha)\mathbf{x}_{\mathbf{baseline}}$ ▷ Interpolate input
    $\mathbf{g}_\alpha \leftarrow \nabla_{\mathbf{x}} F(\mathbf{x}_\alpha)$ ▷ Compute gradients
    IG $\leftarrow$ IG $+ \mathbf{g}_\alpha$ ▷ Accumulate gradients
  end for
  IG $\leftarrow \frac{(\mathbf{x}-\mathbf{x}_{\text{baseline}})}{n} \odot$ IG ▷ Scale and element-wise multiply
  return IG ▷ Return Integrated Gradients
end procedure

Where:

- $\mathbf{x}$: The actual input for which we want to compute the Integrated Gradients.
- F: The neural network model function, which takes an input $\mathbf{x}$ and returns the output prediction. The function F represents the input output mapping of the neural network.
- B: A function that returns the baseline input. The baseline input is an essential aspect of Integrated Gradients, as it serves as the reference point for measuring the contributions of each input feature.
- n: The number of steps used for numerical integration along the straight line path (interpolation) between the baseline input and the actual input. A higher value of n leads to a more accurate approximation of the integral, but at the cost of increased computation time. In practice, an appropriate value of n can be chosen based on the desired tradeoff between accuracy and computational efficiency.

The output of the Integrated Gradients algorithm is an importance score for each input feature, indicating its contribution to the model's prediction relative to the baseline input. These scores can be used to gain insights into the factors driving the model's predictions. Integrated Gradients has several properties that make it an attractive choice for neural network attribution. Firstly, it is applicable to a wide range of deep learning models, including feedforward, recurrent, and convolutional neural networks. Secondly, it satisfies several desirable axioms, such as completeness, which ensures that the sum of the Integrated Gradients for all input features equals the difference in the model's output between the actual input and the baseline input. Moreover, IG is robust to changes in the model's implementation, as it is invariant to any reparameterizations of the model that preserve its input output mapping. This property makes IG a reliable and consistent method for understanding feature importance across different model architectures and training procedures.Integrated Gradients has been successfully applied in various domains, including computer vision, natural language processing, and healthcare.

DeepLIFT DeepLIFT (Deep Learning Important FeaTures) [72] is a powerful attribution method designed to provide insights into the process of the network. By computing the contribution of each input feature to the model's output, DeepLIFT enables to identify the most influential features and assess their impact on the model's predictions. The central idea behind DeepLIFT is to compare the activation of each neuron in the network to a reference activation, which is typically chosen as the average activation over a set of baseline inputs. This comparison yields a set of contribution scores that quantify the importance of each input feature for a specific prediction. By backpropagating these contribution scores through the network, DeepLIFT assigns an importance score to each input feature,

effectively attributing the model's decision to the relevant features. DeepLIFT possesses several key properties and advantages that make it a popular choice for neural network interpretability tasks. One of the main strengths of DeepLIFT is its ability to provide a detailed and fine grained decomposition of the contributions made by individual input features. This level of granularity can be particularly useful for identifying subtle feature interactions and understanding how the model's internal representations change in response to variations in the input. Another advantage of DeepLIFT is its consistency property, which states that the sum of the contribution scores for all input features must equal the difference between the model's output and the reference output. This property ensures that DeepLIFT's importance scores accurately reflect the model's behavior and can be directly compared across different input instances. Furthermore, DeepLIFT is computationally efficient, as it can be easily integrated into existing deep learning frameworks and does not require extensive modifications to the model architecture or training process. This efficiency makes DeepLIFT a practical choice for researchers seeking to understand the inner workings of large scale and complex neural networks.

Despite its strengths, DeepLIFT is not without limitations and challenges. One potential limitation is its reliance on a reference activation, which may not always be easy to define or obtain for certain domains or applications. In cases where a suitable reference activation is not available, DeepLIFT's results may be sensitive to the choice of baseline inputs, potentially affecting the interpretability of the importance scores. Another challenge associated with DeepLIFT is its applicability to more recent and advanced neural network architectures, such as transformers and graph neural networks. Adapting DeepLIFT to these architectures may require modifications to the original method, which could introduce additional complexity and affect its computational efficiency.

Several extensions and adaptations of DeepLIFT have been proposed to address its limitations and broaden its applicability. For instance, Layer-wise Relevance Propagation (LRP) [73] [74] is a related attribution method that shares similarities with DeepLIFT but operates under a different set of constraints. LRP is based on the principle of conserving relevance throughout the network. In LRP, the relevance of each neuron in the output layer is propagated back to the input layer through a series of layer-wise relevance redistribution rules. These rules ensure that the total relevance is conserved at each layer, meaning that the sum of the relevance scores for all input features will equal the model's output.

DeepLIFT and LRP share a common goal of attributing the model's output to individual input features. However, there are some fundamental differences between the two methods that can affect their performance and interpretability in different contexts. One key difference is the way in which the methods handle the reference activation or baseline input. In DeepLIFT, the importance scores are computed by comparing the activation of each neuron to a reference activation, which is typically chosen as the average activation over a set of baseline inputs. In LRP, on the other hand, the relevance scores are derived through a series of layer-wise redistribution rules that conserve the total relevance at each layer. This difference can lead to varying importance scores and interpretations depending on the choice of reference activation or redistribution rules. Another difference between DeepLIFT and LRP is their consistency property. DeepLIFT satisfies the consistency property, which states that the sum of the contribution scores for all input features must equal the difference between the model's output and the reference output. LRP, in contrast, conserves the total relevance at each layer, ensuring that the sum of the relevance scores for all input features will equal the model's output. These different properties can result in different importance score distributions and interpretations of the model's behavior.

DeepLIFT has been applied to a wide range of domains, including genomics, natural language processing, and computer vision. In genomics, DeepLIFT has been used to identify important DNA sequence elements that contribute to gene regulation, while in natural language processing, it has been employed to reveal the importance of individual words or phrases for sentiment analysis and text classification tasks. In computer vision, DeepLIFT has been utilized to understand the contributions of different image regions to object recognition and segmentation. In the context of ECGs, DeepLIFT has been utilized to enhance the interpretability of deep learning models used for detecting and classifying cardiac arrhythmias, heart rate variability, and other cardiovascular conditions. By

attributing importance scores to individual input features, such as specific time points or segments within the ECG signal.

Saliency Maps Saliency Maps [75] are a popular gradient based technique for visualizing the importance of input features in a neural network's decision making process. The fundamental idea behind saliency maps is to compute the gradients of the model's output with respect to the input features, which indicates how the output changes concerning small perturbations in the input. To generate a saliency map, the absolute values of the gradients are computed, and then the maximum value across all color channels is taken for each pixel in the input image. The resulting map highlights the areas where the input has the most significant influence on the model's output, effectively revealing the most salient features for a specific prediction. Despite their simplicity and computational efficiency, saliency maps have some limitations. They can be sensitive to noise and may not provide accurate importance scores for all input features. Additionally, saliency maps may sometimes highlight high frequency patterns that are not meaningful or relevant to human observers.

Grad-CAM Grad-CAM (Gradient-weighted Class Activation Mapping) [76] is another gradient based technique designed to provide more accurate and interpretable visualizations of feature importance. Grad-CAM addresses some of the limitations of saliency maps by considering not only the gradients but also the activation patterns in the convolutional layers of the neural network. The Grad-CAM method involves computing the gradients of the model's output with respect to the feature maps of a target convolutional layer. These gradients are then globally average pooled to obtain the weights that represent the importance of each feature map for the specific output. Next, the feature maps are multiplied by their corresponding weights and summed, resulting in a coarse localization map. This map is then upsampled to the input resolution using bilinear interpolation, providing a visual representation of the areas in the input image that contribute the most to the model's prediction. Grad-CAM has several advantages over saliency maps. It is more robust to noise, as it considers both the gradients and the activations, which leads to smoother and more interpretable visualizations. Grad-CAM is also class discriminative, meaning it highlights the regions in the input image that are most relevant to a specific class, making it suitable for multi class classification problems. However, Grad-CAM is constrained to the spatial resolution of the target convolutional layer, which may not capture fine grained details in the input image. Furthermore, Grad-CAM relies on the choice of a suitable target layer, which may require domain knowledge or experimentation.

Active approaches In contrast to passive approaches, active interpretability methods incorporate interpretability constraints or objectives during the model training process.

Attention mechanisms Attention mechanisms are an example of active interpretability, as they provide an interpretable way of understanding the relative importance of different elements in the model's decision process [77][78]. They were first introduced to address the limitations of fixed size representations in sequence-to-sequence models, particularly in the context of machine translation. These mechanisms enable models to dynamically weigh the importance of different input elements when generating outputs, essentially allowing them to "focus" on the most relevant parts of the input for a particular task. By doing so, attention mechanisms not only improve the performance of neural networks but also offer a means to interpret the decision process. The interpretability of attention mechanisms comes from their ability to highlight the relationships between input and output elements. In the context of sequence-to-sequence models, for example, attention weights can reveal the alignment between words in the source and target sentences. By examining the attention weights, one can gain insights into how the model is processing the input data and which elements it deems most relevant for generating its predictions. One common approach to interpreting attention mechanisms is to visualize the attention weights as heatmaps or matrices. In the case of machine translation, this can involve creating a matrix where rows correspond to the words in the source sentence, columns represent the words in the target sentence, and the cell values represent the attention weights assigned

by the model. By examining this matrix, one can observe which source words the model is attending to when generating each target word, revealing the underlying relationships between the input and output sequences. Visualizations of attention weights can also be employed in other domains, such as computer vision and natural language processing. For instance, in image captioning tasks, attention weights can be overlaid on input images to highlight the regions the model is focusing on when generating specific words in the caption. Similarly, in sentiment analysis or question-answering tasks, attention weights can be used to identify the words or phrases in the input text that the model considers most relevant for determining its output. Similarly, in the context of ECGs, attention mechanisms can be used to highlight specific areas of the signal that the model use to perform a prediction. Beyond visualizing attention weights, we can also analyze attention mechanisms to gain insights into the model's behavior. This can involve examining the distribution of attention weights, investigating the properties of learned attention functions, or probing the model's behavior under different input conditions. For example, by analyzing the distribution of attention weights, one can detect potential issues such as the model overly focusing on a single input element or uniformly distributing its attention across all elements. In such cases, the attention mechanism may not be providing meaningful interpretability, and further investigation or adjustments may be required. Another approach to analyzing attention mechanisms is to study the learned attention functions themselves. By examining these functions, researchers can gain insights into how the model combines different input elements and the nature of the relationships it learns. This can help uncover potential biases or inconsistencies in the model's prediction process. While attention mechanisms provide a valuable means of interpreting neural network behavior, it is important to recognize their limitations and challenges. Attention weights may not always provide a faithful representation of the model's reasoning process, and they can sometimes be sensitive to small changes in input or model parameters. Furthermore, attention mechanisms are not universally applicable across all neural network architectures, and their effectiveness in providing interpretability may vary depending on the specific model and task.

Distillation techniques Distillation techniques [79] can also contribute to active interpretability. Model distillation involves training a simpler, more interpretable model (called the "student") to mimic the behavior of a complex, less interpretable model (the "teacher"). By transferring knowledge from the teacher to the student, distillation techniques can yield more interpretable models while retaining high performance. Examples of models used for distillation include decision trees, linear models, and shallow neural networks. Decision trees, for instance, provide a transparent representation of the model's decision process through a hierarchical structure of binary decisions based on input features. Linear models, on the other hand, offer a simple and interpretable way of understanding feature importance through their learned weights, while shallow neural networks can provide a balance between model complexity and interpretability. Distillation techniques can also be combined with other active approaches, such as attention mechanisms or regularization techniques, to further enhance the interpretability of the student model.

Local, semi-local, and global interpretability

Interpretability can be considered from different perspectives, including local, semi-local, and global. Each perspective offers unique insights into the model's behavior and prediction process, enabling researchers to evaluate model performance and identify potential issues that may arise.

Local interpretability Local interpretability focuses on understanding individual model predictions. By examining specific instances, local interpretability aims to provide insights into how the model arrives at its decisions for a particular input. This perspective is valuable for debugging and identifying biases or inconsistencies in the model's decision-making process for individual instances. Local explanations can help uncover the contributions of various input features to the model's prediction, highlighting the importance of individual features for that specific instance.

Semi-local interpretability Semi-local interpretability aims to provide insights into the model's behavior within a specific region of the input space, often defined by a subset of similar instances. This perspective can help understand how the model generalizes across different groups or categories, enabling them to detect potential biases or inconsistencies in the model's performance. Techniques for achieving semi-local interpretability include clustering-based methods that group similar instances together, enabling researchers to analyze the model's behavior within each cluster. Aggregating local explanations within the region of interest can also provide valuable insights into the model's prediction process for a particular group of instances. By examining these semi-local explanations, we can gain a better understanding of the model's generalization capabilities and identify potential issues that may arise when applying the model to new instances within the defined region.

Global interpretability Global interpretability seeks to understand the overall behavior of the neural network across the entire input space. This perspective is crucial for comprehending the model's general decision process, identifying overarching trends, and ensuring that the model aligns with human expectations. Methods for achieving global interpretability include feature importance rankings, which provide an overall measure of the significance of each input feature in the model's behavior. By examining these rankings, researchers can identify the most influential features and assess their impact on model predictions. Surrogate models, such as decision trees or linear regression models, can also be employed to approximate the behavior of the neural network. These surrogate models offer a more interpretable representation of the model's behavior, enabling us to gain insights into the global behavior of the complex neural network. Rule extraction techniques, such as decision tree induction or association rule mining, can be used to generate a set of human-readable rules that approximate the model's overall behavior. Prototypes differences can also be used as database of concepts or rules as they provide insights of the dataset sub groups.

		Local	Semi-local	Global
Passive	**Rule**	CEM[80], CDRPs[81], CVE2[82], DACE[83]	Anchors[84], Interpretable partial substitution[85]	KT[86], $MofN$[87], NeuralRule[88], NeuroLinear[89], GRG[90], Gyan^{FO}[91], $\bullet^{FZ}$[92][93], Trepan[94], $\bullet$[95], DecText[96], Global model on CEM[97]
	Hidden semantics	(*No explicit methods but many in the below cell could be applied here.)	—	Visualization [98] [75] [99] [100][101] [102] [103], Network dissection[104], Net2Vec[105], Linguistic correlation analysis[106]
	Attribution[1]	LIME[61], MAPLE[107], Partial derivatives[75], DeconvNet[108], Guided backprop[109], Guided Grad-CAM[110], Shapley values[111][112][113][114], Sensitivity analysis[108][115][116], Feature selector[117], Bias attribution[118], Occlusion	DeepLIFT[119], LRP[73], Integrated gradients [59], Feature selector[117], MAME[120]	Feature selector[117], TCAV[65], ACE[121], SpRAy[3][122], MAME[120] DeepConsensus[123]
	By example	Influence functions[124], Representer point selection [125], Counterfactuals	—	—
Active	**Rule**	—	Regional tree regularization[126]	Tree regularization[127]
	Hidden semantics	—	—	"One filter, one concept"[128]
	Attribution	ExpO[129], DAPr[130]	—	Dual-net (feature importance)[131]
	By example	—	—	Network with a prototype layer[132], ProtoPNet[133]

[FO] First-order rule
[FZ] Fuzzy rule
[1] Some attribution methods (e.g., DeconvNet, Guided Backprop) arguably have certain non-locality because of the rectification operation.
[2] Short for counterfactual visual explanations
[3] SpRAy is flexible to provide semi-local or global explanations by clustering local (individual) attributions.

Table 4: An overview of the interpretability papers [3].

3.3 Proposed solution and methodology

3.3.1 Proposed solution

To address the issues of the occlusion method we tested in our previous work, we proposed and developed a novel original method of neural network interpretability named evocclusion. We chose to improve the occlusion method as it is easier to understand and interpret compared to other methods. Our method aims to address the limitations of occlusion as it is known to be often limited in its ability to identify minimal perturbations that lead to changes in model predictions or to comprehensively reveal the most influential input features contributing to the decision process. The method primarily

focuses on classification tasks. It identifies the different parts of the input data that are relevant for the model's decision process as well as possible relationships between them. Our approach begins with the application of perturbations/transformations to the input data and the evaluation of the trained model's response to these perturbations using integrated gradients and prediction deltas, similar to the occlusion techniques. However, instead of relying solely on occlusion, we use the concept of counterfactuals, which are the minimal perturbations that cause a model's prediction to change to the opposite class. To efficiently explore the infinite number of possible perturbation combinations, we developed and implemented a genetic algorithm. The genotype, represented in binary, encodes each gene, which represent a specific transformation (perturbation) and its parameters. The genetic algorithm iteratively applies population creation, individual selection, genetic crossover and mutation operations to optimize the population of individuals. The fitness function is formulated to balance the contributions of proximity to the model's inter-class decision boundary and the minimal perturbation required to flip the prediction. By identifying counterfactuals that flip the model's prediction with minimal perturbations, our method can effectively highlight the input features most responsible for the model's decision process. To reveal the relevant features from the identified counterfactuals, we compute attribution masks for each individual at the final generation of the genetic algorithm. These attribution masks are generated using a technique similar to the integrated gradients, which measures the contribution of each input feature to the model's output. By examining the attribution masks, we can identify the regions of the input data that have the greatest impact on the model's decision. In essence, the attribution masks derived from the counterfactuals highlight the most influential input features, providing a deeper understanding of the model's behavior. This information is particularly valuable for practitioners who wish to verify the model's predictions, uncover potential biases, or improve the model's performance by incorporating the insights gained from the interpretability method.

3.3.2 Methodology

The Evocclusion method is designed to work on different type of input data and classification models. In this study we focus on ECG and our previous TdP risk prediction model.

3.3.2.1 Prototypes as domain-specific knowledge transformations

To ensure the perturbations applied to the input data are meaningful and representative of the underlying data distribution, our method uses domain-specific prototypes and transformations. This approach aims to leverage the intrinsic structure of the data and incorporate domain knowledge into the perturbation process, resulting in more informative and interpretable counterfactuals.

For a given ECG dataset, we compute the prototypes for each patient and class. These prototypes serve as reference points that embody the characteristic patterns of the respective classes, such as the morphology of the P wave, QRS complex, and T wave, as well as temporal features like the RR interval and QT distance.

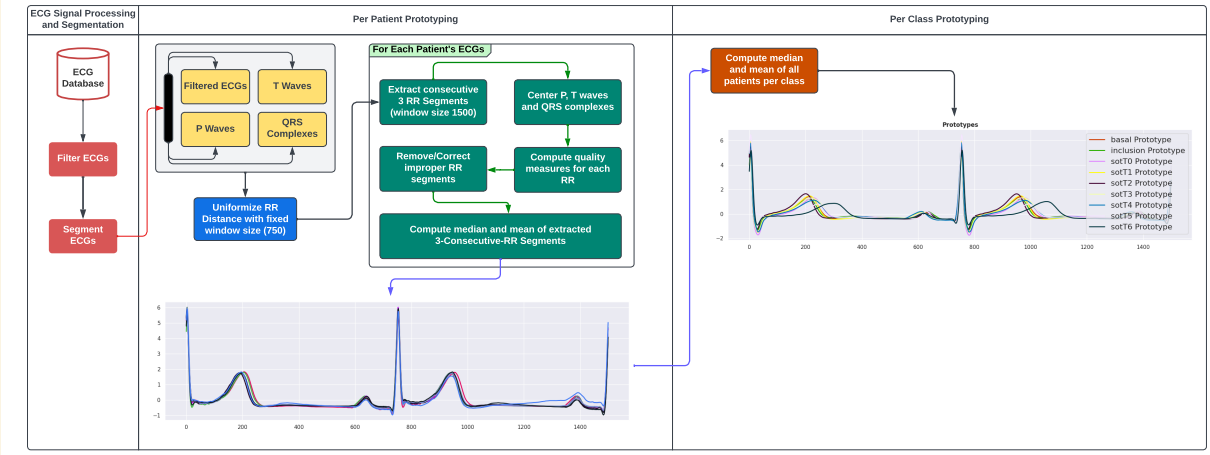
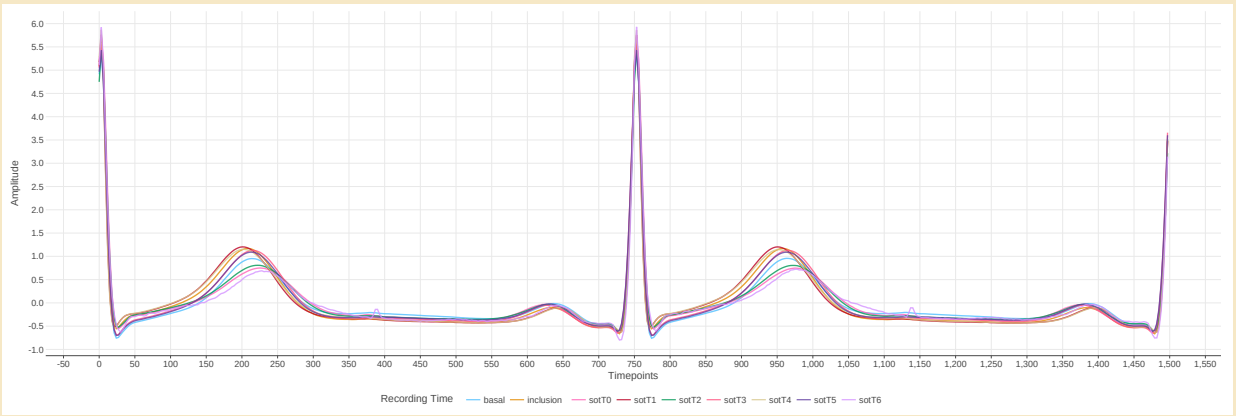


Figure 21: ***ECGs prototyping process*** *We compute for each ECG a mean or median prototype of 1500 points and 3 consecutive R peaks. For each patients then class we compute a mean or median prototype.*

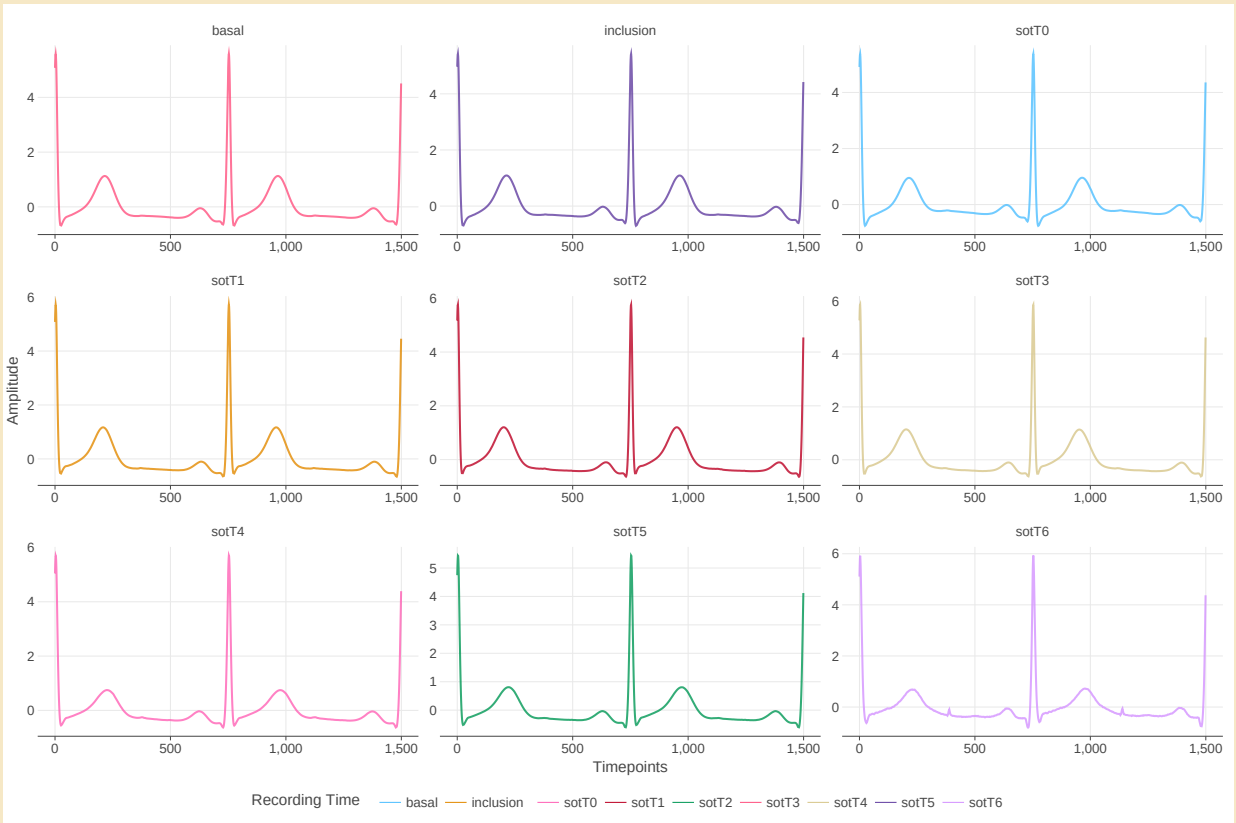
Figure 21 illustrates the process of computing prototypes. For each ECG we identify R peaks and extract from the signal all 3 consecutive R peaks. The extracted patches are interpolated to a fixed length of 1500 points (750 points between 2 R peaks after interpolation), this process is necessary for computing an average prototype and can be considered as a temporal normalization. However, this process removes the heart rate frequency which is stored for the reverse downsizing operation. For a given ECG of 9 R peaks we can typically extract 7 prototypes of 3 consecutive R peaks of 1500 points length each. We then compute the mean or median of them and obtain a prototype for a given ECG. The same process is done on all prototypes of all patients individually, then on all classes individually. The process results in ECG prototypes for each ECG, patient and class. Because ECGs are frequently subject to noise that can arise during the recording process, we developed a score to evaluate the quality of the signal, which is coded as a vector with the same length as the ECG ranging in [0 to 1]. A value close to 1 indicates that the corresponding part of the signal has a good quality whereas a value close to 0 indicates the corresponding part of the ECG has a bad quality due to noise or distortion (baseline drift). Prototypes computing process uses a quality threshold of 0.5 per beat, every beat with an average quality score below 0.5 is discarded and replaced with an average representative beat. Thus we ensure that the resulting prototype for a given ECG does contain minimal noise. To further reduce the noise in the signal and ensure that the R peaks are accurately detected and that the extracted patch are correctly aligned and centered, we developed a denoising auto-encoder neural network *(in Chapter 4)* and other models to automatically segment the signal *(in Chapter 5)*. The signals are first denoised, then R peaks are detected. Dedicated segmentation models detect P, T waves and QRS peaks position, which are further used to center patches. However denoised signals are not used in the mean/median operation of computing the prototype as we intend to keep original signal. In Figure 22a and 22b, we present computed prototypes on the Generepol dataset for each class: basal, inclusion sotT0-T6. We can observe changes in the waveform across time with a visible prolongation of the T wave at 6 hours after sotalol intake. We also notice changes in the P wave and S peak.

To generate perturbations, we transform the input ECG by adding various deltas calculated from the prototypes. These deltas can be derived from the differences between the prototypes or their statistical properties, such as the mean and covariance of the feature distributions within specified regions of the signal. By constructing the perturbations based on the prototypes, our method ensures that the generated perturbations are relevant to the problem domain and respect the underlying data distribution. For example, a given ECG at sotT0 can be translated to sotT6 by adding to it the delta $sotT6 - sotT0$.

This approach allows the method to explore different classes and combinations in a more informed manner. For instance, the perturbations may involve altering the amplitude or duration of the P wave, T wave or QRS complex, or modifying the distance between specific ECG components, such as the QT interval. This is done using the segmentation models, which pinpoint the coordinates of specific areas on the signal. By focusing the search on perturbations that have a meaningful impact on the ECG signal, our method can identify more effectively the critical input features that influence the model's decision process. Furthermore, using domain-specific prototypes and transformations enables the incorporation of expert knowledge and clinical guidelines into the perturbation process. This can lead to the discovery of more clinically relevant counterfactuals and enhance the interpretability of the results, as the identified perturbations will be in line with established patterns and relationships in the data.



(a)



(b)

Figure 22: ***Generepol ECG prototypes per class****: For each ECG we compute a prototype using 3 consecutive R peaks and average them within the ECG then per patient and finally per class.*

3.3.2.2 Introduction to genetic algorithms

Genetic algorithms are a class of optimization techniques inspired by the process of natural selection in biological systems. They have gained significant attention in recent years due to their ability to tackle complex optimization problems that are difficult to address using traditional methods, such as gradient based or exhaustive search techniques. Genetic algorithms have been successfully applied to a wide range of problems, including function optimization, machine learning, scheduling, and game playing, among others.

The fundamental idea behind genetic algorithms is to mimic the process of natural evolution, which operates on the principle of survival of the fittest. In genetic algorithms, a population of candidate solutions to an optimization problem, referred to as individuals, evolves over multiple generations to converge towards an optimal or near-optimal solution. The evolutionary process involves the application of genetic operators, such as selection, crossover, and mutation, which are designed to facilitate the exploration and exploitation of the search space.

A critical aspect of genetic algorithms is the representation of individuals, which determines how candidate solutions are encoded in the algorithm. Common representations include binary strings, real valued vectors, or more complex data structures (object oriented representation), depending on the problem domain. The choice of representation can significantly impact the performance of the genetic algorithm and the ease with which domain-specific knowledge can be incorporated.

The genetic algorithm begins with the initialization of an initial population, typically generated randomly or using a combination of random and heuristic techniques. The size of the population and the method of initialization can influence the diversity of the initial search space and the rate at which the algorithm converges towards an optimal solution.

Concepts

Fitness evaluation The quality of each individual in the population is assessed using a fitness function, which measures the individual's performance with respect to the optimization problem. The fitness function should be designed to reflect the problem's objectives and constraints, and it plays a crucial role in guiding the search process. In many cases, the design of an appropriate and efficient fitness function can be a challenging aspect of applying genetic algorithms to complex problems.

Indivual selection Selection is the process by which individuals are chosen to participate in the reproduction process based on their fitness. The goal of selection is to bias the search process towards higher quality solutions while maintaining sufficient diversity in the population. Various selection schemes have been proposed, such as roulette wheel selection, tournament selection or rank based selection, each with its advantages and disadvantages. The choice of selection scheme can affect the algorithm's convergence properties and its ability to maintain diversity in the population.

Crossover Crossover, also known as recombination, is the process by which pairs of individuals exchange genetic material to produce offspring. The crossover operator is intended to combine the best features of the parent individuals, potentially leading to the creation of better solutions in the offspring. Numerous crossover operators have been proposed for different representations, such as one-point, two-point, or uniform crossover for binary strings, and blend or arithmetic crossover for real valued vectors. The choice of crossover operator can significantly impact the algorithm's ability to explore the search space and converge towards an optimal solution.

Mutation Mutation is the process by which small, random changes are introduced into an individual's genetic material. The purpose of mutation is to maintain diversity in the population and

prevent premature convergence to suboptimal solutions. Like crossover, the choice of mutation operator depends on the representation used, and different mutation schemes have been proposed for binary strings, real valued vectors, and other representations. The mutation rate, which determines the probability of a mutation occurring, is an important parameter that can affect the balance between exploration and exploitation in the search process.

Convergence A genetic algorithm iterates through successive generations of selection, crossover, and mutation until a termination criterion is met. Common termination criteria include reaching a maximum number of generations, achieving a specified level of fitness, or detecting a lack of progress in the search process. The choice of termination criterion can affect the computational cost of the algorithm and the quality of the final solution obtained. Convergence properties of genetic algorithms are an important aspect of their analysis and design. The algorithm should ideally converge towards an optimal or near-optimal solution, but the rate of convergence and the ability to escape local optima depend on the choice of genetic operators, the representation, and the parameter settings. Theoretical analysis and empirical studies of genetic algorithms have provided insights into their convergence properties and guided the development of improved algorithmic variants and parameter tuning strategies.

To improve the performance of genetic algorithms and adapt them to specific problem domains, various advanced techniques and hybrid approaches have been proposed. Some of these techniques include incorporating domain-specific knowledge into the representation, fitness function, or genetic operators; using adaptive or self-tuning parameter settings; or incorporating local search techniques, such as hill climbing or simulated annealing, to enhance the algorithm's exploitation capabilities. Hybrid approaches, often referred to as memetic algorithms or genetic local search, combine the global search capabilities of genetic algorithms with the local search capabilities of other optimization techniques. These hybrid algorithms have been shown to provide improved performance on a wide range of problems, particularly those with complex search spaces or multiple local optima.

3.3.2.3 Evocclusion genetic algorithm

In our proposed interpretability method illustrated in Figure 23, the genetic algorithm plays a central role in searching for the most informative perturbations that can flip the neural network's predictions. The algorithm's genotype, which is essentially the blueprint for generating perturbations, is encoded as a binary string. Each gene within the binary string corresponds to a specific transformation, along with its associated parameters. These transformations are essentially the perturbations that will be applied to the input data.

Transformations and genotype representation An individual in the context of our genetic algorithm represents a specific combination of perturbations, as determined by its genotype. The genotype of an individual dictates which transformations are applied to the input data, and how these transformations are parameterized. For example, a gene may represent a transformation that modifies the amplitude of a specific ECG waveform component, and the associated parameters in the genotype would define the degree of amplitude change. The population, which is the collection of individuals, consists of multiple unique genotypes, each encoding different combinations of perturbations. This diversity in the population allows the genetic algorithm to explore a broad range of potential perturbations, increasing the likelihood of finding those that minimally flip the neural network's predictions. We considered meaningful transformations based on prototypes, as well as standard occlusion transformation. To transform an ECG from an initial class to another we add to the signal the delta between the two prototypes representing the respective classes, in the context of our study using the Generepol dataset, we also used patient's prototypes as well ass class prototypes. For a given patient, their ECGs transformations can be based on their prototypes instead of the global ones. With the help of our segmentation method we were able to restrict the transformation on specific areas of the ECG. These

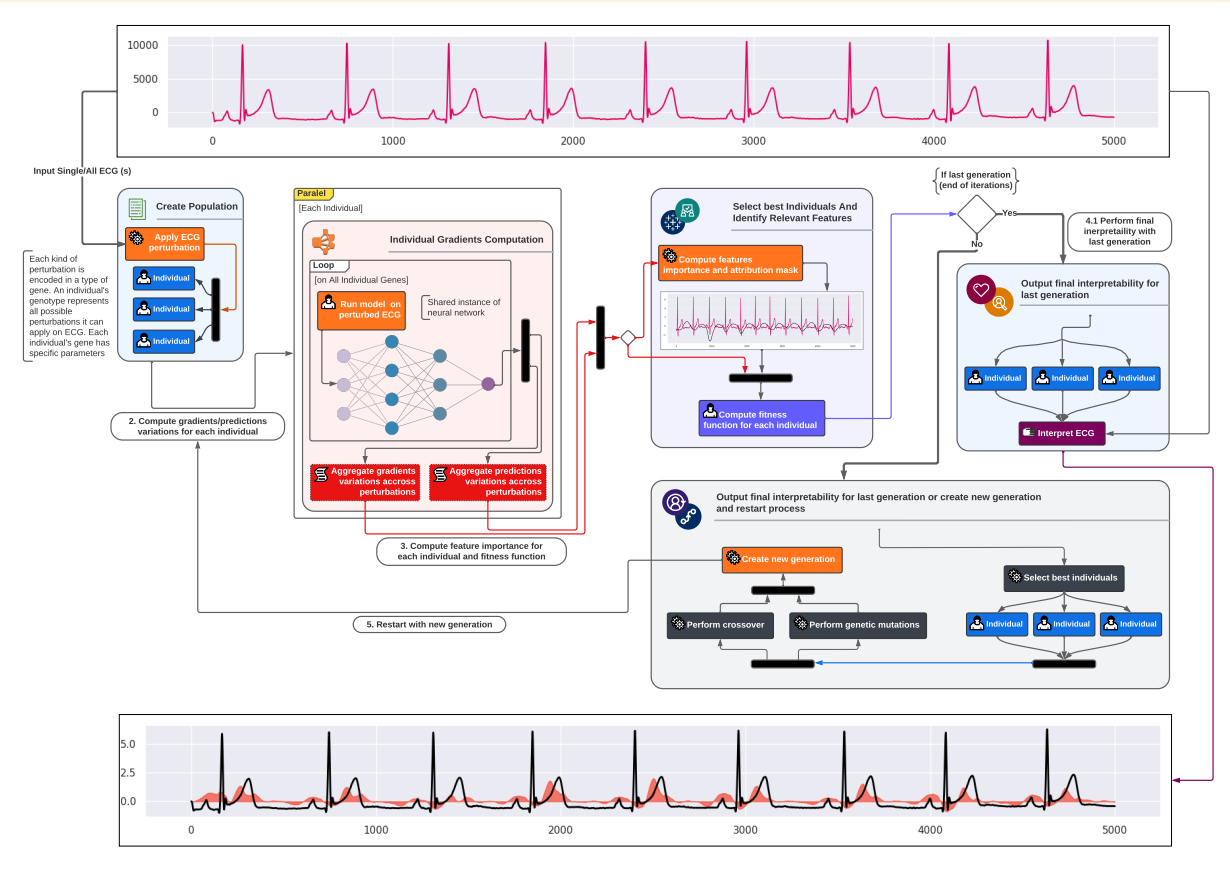


Figure 23: ***The Evocclusion interpretability algorithm.***

parameters are encoded in the genotype and represent genes of the individual. For the fitness function to capture subtle changes in the model's output and gradients we interpolate the transformation path between the initial ECG and the prototype based resulting transformation in a parameterized number of steps. The more the steps better the fitness can capture gradients variations. Transformations can also span across all the signal. The transformation of an ECG requires temporarily removing the heart rhythm by normalizing all RR distance to 750 points so we can add the delta prototype then add back the heart rhythm. We also used zero-based occlusion on some part of the signal. In Figure 24 we computed the prototype of a given ECG from class basal (without Sotalol), to transform it to a sotT6, we added to it the delta prototype between basal and sotT6. The resulting ECG prototype shows distinctive features that can be observed on sotT6 signals. A visible prolongation of the T wave and slight prolongation of the P wave can be noticed.

To focus the transformation we repeated the same experiment however with a constraint on the T wave, the coordinates of the T onset and T offset is given by segmentation neural networks. An illustration of the result is given in Figure 25. In Figure 26 we present a complete ECG which transformed representation flips the model prediction to the opposite class. Each transformation is applied individually on the ECG resulting in many ECGs transformed for a given individual, separating transformed ECGs allows the method to capture contributions of each transformation. Table 5 describes the various types of transformations we currently considered.

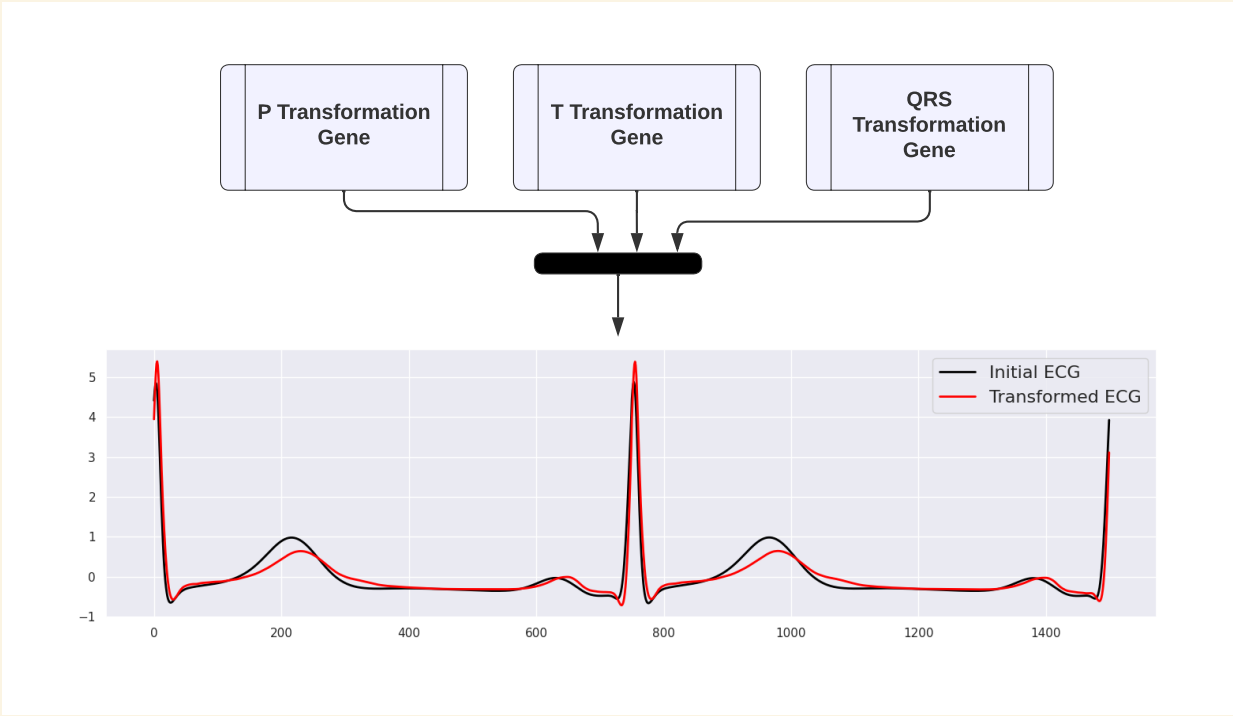


Figure 24: ***Class prototype transformation on ECG prototype****: we computed the prototype of a given ECG then added to it the delta prototype between class basal and class sotT6. We can notice a significant change in the T and P waves morphology after transformation typical of a sotT6 ECG.*

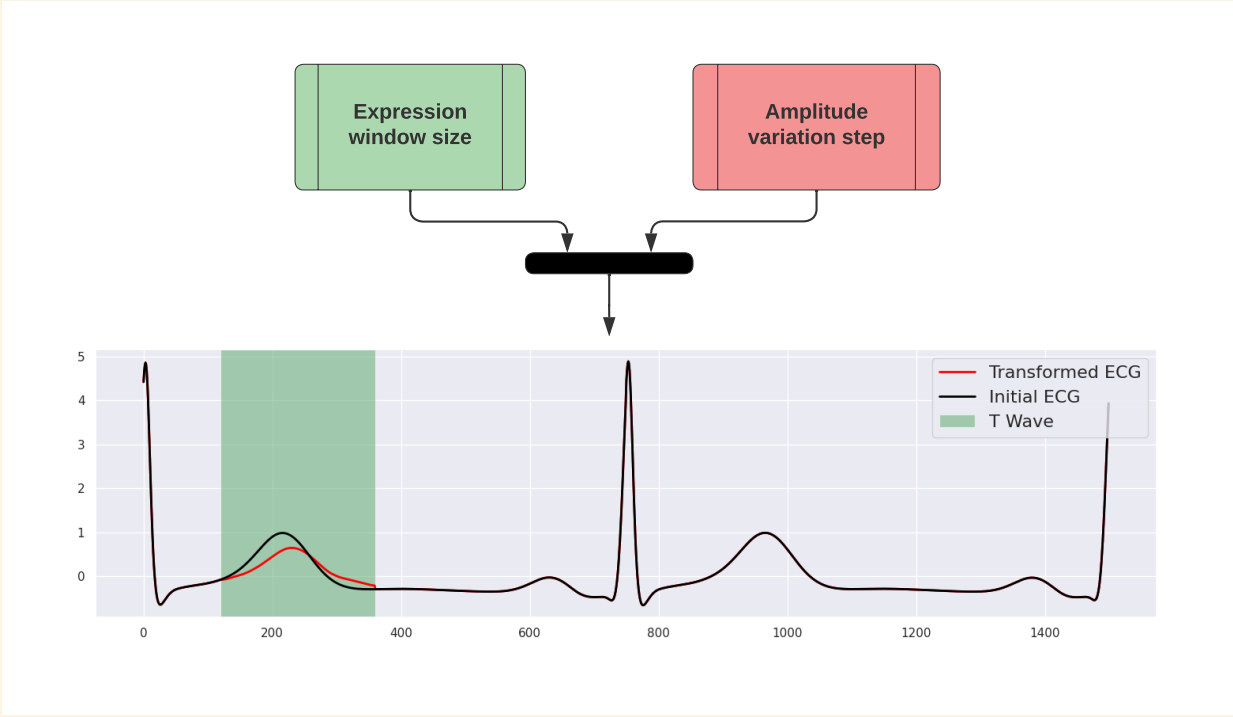


Figure 25: ***Class prototype transformation on ECG prototype on T Wave****: the same transformation in Figure 24 is constrained on the T Wave area.*

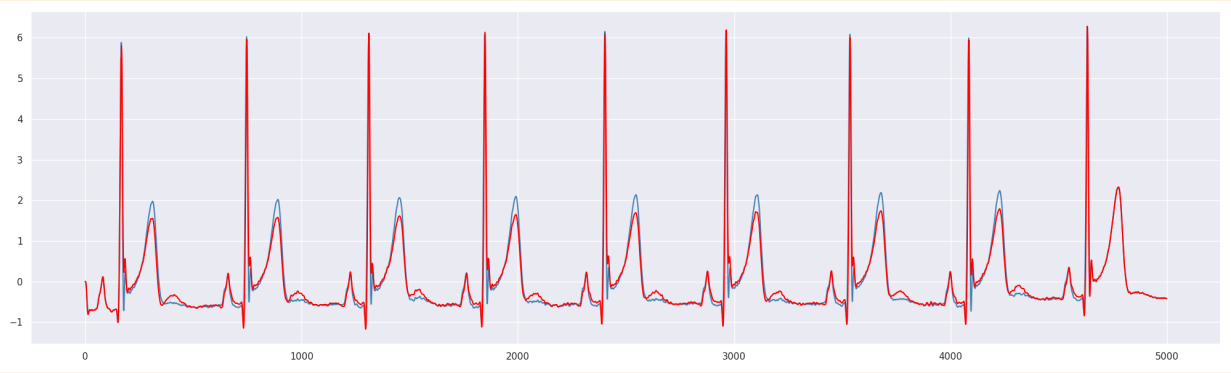


Figure 26: ***ECG transformed using prototype****: in blue the original ECG, in red one of the individual's transformations at which point the model decision changes towards the opposite class and gradients variation change direction.*

```
genome = [
    {
        "instances": {"min": 1, "max": 30},
        "function": "uniform_p_wave_prototype_transformer",
        "function_arguments": {
            "steps": {"min": 1, "max": 50},
            "prototypes_combination": {"min": 0, "max": 132}
        }
    },
    {
        "instances": {"min": 1, "max": 30},
        "function": "uniform_t_wave_prototype_transformer",
        "function_arguments": {
            "steps": {"min": 1, "max": 50},
            "prototypes_combination": {"min": 0, "max": 132}
        }
    },
    {
        "instances": {"min": 1, "max": 30},
        "function": "uniform_qrs_complex_prototype_transformer",
        "function_arguments": {
            "steps": {"min": 1, "max": 50},
            "prototypes_combination": {"min": 0, "max": 132}
        }
    },
    {
        "instances": {"min": 0, "max": 70},
        "function": "occlusion_transformer",
        "function_arguments": {
            "window_size": {"min": 1, "max": 50},
            "position": {"min": 0, "max": 4950}
        }
    }
]
```

Minimum and maximum number of instances of this gene

Data transformation function represented by the gene

Arguments of the transformation function

Gene type definition

For each part of the gene we compute its maximum binary representation based on the *max* value of each field (instances and function arguments)

For this specific gene type its binary length is:
– Number of instances: 5 bits
– Function arguments: 14 bits
  – steps: 6 bits
  – prototypes_combination: 8 bits
– Total: 14 bits * 30 instances = 420 bits

Genes position in the genetic code is fixed and determined by their position in the given genome

Figure 27: ***Genome, ECG transformations representation****: the possible transformations are stored in a JSON file fed to the algorithm which transforms it into binary representation*

Transformation Operation	Description	Parameters
base_prototype	Applies a prototype based transformation on all the signal	• *steps*: number of steps to interpolate the transformation path • *prototype_combination*: The prototype delta to add to the ECG
uniform_p_wave_prototype	Applies a prototype based transformation on all P waves of the signal using segmentation coordinates	
uniform_t_wave_prototype	Applies a prototype based transformation on all T waves of the signal using segmentation coordinates	
uniform_qrs_complex_prototype	Applies a prototype based transformation on all QRS complexes of the signal using segmentation coordinates	
occlusion	Applies a single occlusion window on the signal, by replacing corresponding points with zeros	• *window_size*: length of the occlusion window • *position*: start position of the window
uniform_{p_wave,t_wave,qrs_complex}_occlusion	Applies a single occlusion window on the P wave or T wave or QRS complex of all beats, by replacing corresponding points with zeros	• *window_size*: length of the occlusion window • *position*: start position of the window
single_beat_{p_wave,t_wave,qrs_complex}_occlusion	Applies a single occlusion window on the P wave or T wave or QRS complex of a specific beat, by replacing corresponding points with zeros	• *window_size*: length of the occlusion window • *position*: start position of the window • *beat*: beat id
single_beat_no_wave_occlusion	Applies a single occlusion window on part of a specific beat outside of P, T waves and QRS complex, by replacing corresponding points with zeros	• *window_size*: length of the occlusion window • *position*: start position of the window • *beat*: beat id

Table 5: Evocclusion ECG signal possible transformations: *we use prototype based transformations and classic occlusion based transformation on all the signal or at targeted areas such as the P or T wave, QRS complex or the rest of the beat signal. Beat related transformation operation's parameters window_size and position are expressed in percentages and are relative to the length of the target beat.*

Genes representation The binary representation employed in our genetic algorithm-based interpretability method serves as an efficient and compact way to encode the genotype of each individual in the population. In this representation, each gene corresponds to a specific transformation (perturbation) and its parameters, encoded as a binary string. This allows for a straightforward encoding

and decoding of the genotype, facilitating the implementation of genetic operations such as selection, crossover, and mutation. Furthermore, the binary representation offers a high degree of flexibility, as it can easily accommodate a wide range of transformations and parameter settings. By employing a binary representation in our method, we enable the genetic algorithm to effectively explore the search space of possible perturbations and identify those that provide the most meaningful insights into the model's decision process. All individual has a common fixed genotype length. A transformation and its parameters are the chromosomes of the genotype.

The transformations are represented using the following rules for all individuals:

- Each transformation type has a number of minimum and maximum instances set for all individual, each instance has its own parameters. Each individual has a given number of instances for a specific type of transformation, this allows to tweak the algorithm and test different scenarios.
- For each transformation we compute the number of bits required to represent its maximum number of possible instances
- Each transformation's parameter has a minimum and maximum value, a fixed possible value can also be set
- After computing the number of bits to represent the maximum number of instances, we compute the number of bits required to represent the all parameters and multiply it by the maximum number of instances. This allows all individuals to have a fixed genotype length. Each transformation starts with an activation bit which determines if it will be evaluated when computing the fitness.
- All individual have all possible transformation type but not all of them are activated depending on their genotype.
- Transformations are represented sequentially in the binary string, the order is important for parsing the binary string.
- At individual creation, a validity check function checks if the genotype is coherent, if a given transformation parameter is not valid it is regenerated until validity is reached. A validity dictionnary is provided for each possible transformation.

Figure 27 we present and explain the genotype representation in object format before generating a binary sample.

```
{
  "instances": { "min": 1, "max": 3 },
  "function": {"base_prototype_transformer",
  "function_arguments": {
    "steps": { "min": 1, "max": 3 },
    "prototypes_combination": { "min": 0, "max": 132 }
  }
}
```

Listing 3.1: Gene definition example

In this transformation genetic representation 3.1, the number of instances is represented using 2 bits, the function arguments for a given instance using 11 bits (*steps*: 3 bits, *prototypes_combination*: 8 bits). The total number of bits being $11bits * 2instances = 22bits$. The resulting transformation is given in Figure 28 where we generated a sample individual with genotype:

$$1110101100101011001011011000111000$$

.

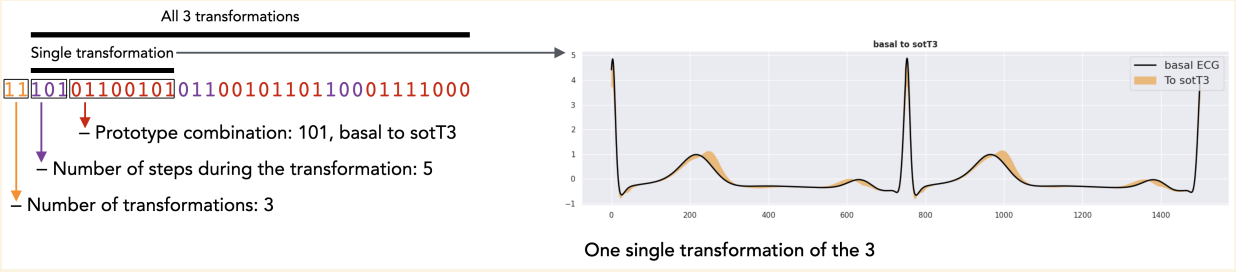


Figure 28: ***Example of a genetic transformation.***

Computing genes presence Genetic operations such as selection, crossover or mutation often rely on probability and randomness. However to improve convergence speed of the algorithm and focus on valuable solutions while conserving diversity, we compute the presence rate of each gene across n previous generations. The n previous generations are denoted as explorable history. The presence rate of each gene is computed by evaluating how often a given gene at given position is present in each individual genome for each previous generation. In this stage, the method iterates through the exploration of the stored evolution data of the populations. Each population is sorted according to individual fitness, and the top individuals (75th percentile by default) are selected as representative samples for the respective generation. For every gene in the top individuals, the method counts the number of occurrences (presence) and accumulates the fitness scores associated with the individuals carrying the gene. The total count of gene presence provides insight into the gene's effectiveness across generations, while the accumulation of fitness scores enables an assessment of the gene's historical contribution to overall fitness. In a second stage, an average fitness is computed by dividing the total accumulated fitness by the gene's presence count, essentially providing a measure of how much, on average, the gene contributes to an individual's fitness. A presence rate is also computed for each gene. It is a normalized measure of the gene's prevalence in the top rated individuals across generations. The presence rate is further adjusted using the maximum gene presence rate to ensure the value lies within the interval [0, 1], which allows for easier interpretation and usage. Upon calculating these two metrics, they are combined using a geometric mean to form a unified measure of the gene's success, where a high score indicates a gene that has both high average fitness and a high presence rate. This success rate is then used to update the gene's mutation rate. The update follows a dampening mechanism that ensures the mutation rate decreases if the gene has a high success rate and increases if the gene has a low success rate. This dynamic adjustment of mutation rates ensures that successful genes are preserved while less successful ones are subjected to a higher rate of mutation to explore new possibilities. This mutation probability and presence rate are used in crossover and mutation operations to provide a valuable trade-off between best solutions and exploration. The mutation rate is clamped, if the mutation rate is too high, the algorithm might constantly make changes and struggle to converge to an optimal solution. Conversely, if the mutation rate is too low, the algorithm might get stuck in local optima and not explore enough of the solution space.

Selection process Throughout the optimization process, the genetic algorithm iteratively updates the population by applying various genetic operations, such as selection, crossover, and mutation. These operations aim to improve the overall fitness of the population by retaining the best performing individuals, combining their genetic material to create offspring, and introducing random changes to explore new areas in the search space. As a result, the genetic algorithm evolves the population over time to identify the minimal perturbations that provide the most insight into the neural network's prediction process.

Crossover Crossover, also known as recombination, is the process of combining genetic material from two parent individuals to create one or more offspring. For example, considering two parent individuals with binary genotypes A = 110010 and B = 001101. A single point crossover operation

could be applied at the third position, resulting in the offspring genotypes A' = 110101 and B' = 001010. This operation promotes the exchange of genetic information between individuals, potentially generating offspring with higher fitness values. We used a crossover rate (Pc), a probability that a pair of individuals will undergo crossover during the reproduction process. A higher crossover rate encourages more exploration of the search space, while a lower rate favors exploitation of the current solutions. The Pc is a hyper-parameter of the algorithm. While this method can promote genetic diversity and exploration of the search space, it might be too naive for certain complex optimization problems. This method might lead to a break of a good combination of genes and could cause a drop in the quality of offspring. In our method, we used an adaptive crossover function aiming to address these challenges. The idea behind the adaptive crossover is to choose the genes for the offspring based not only on the parent's overall fitness, but also on the individual success rates of the genes. For each gene, a random number is generated as a crossover point. If this number is below a specified threshold, the gene for the offspring is chosen from the parent with the higher overall fitness. If the crossover point is above the threshold, the gene is chosen based on its success rate. This method has the potential to generate offspring that are not only similar to the fittest parents but also include successful genes from less fit parents. This introduces a balance between exploration and exploitation in the search space. Our adaptive crossover function therefore allows the algorithm to adapt more efficiently to the fitness landscape. Compared to single point crossover, our approach is more likely to maintain beneficial genetic structures and promote a guided exploration, potentially leading to faster convergence to a good solution. This approach is particularly beneficial in our complex search space, where the optimal solution is constructed by a delicate combination of various genes (transformations), and the disruption of these combinations can significantly decrease the fitness of the offspring. The threshold is denoted $adaptive\_crossover\_rate$

Mutation We also used mutation to introduce small random changes to an individual's genotype. This operation helps to maintain diversity within the population and prevents premature convergence. For example, consider an individual with a binary genotype C = 110010. A mutation operation could be applied to flip the second bit, resulting in the mutated genotype C' = 100010. Mutation rate (Pm) is the probability that a gene within an individual's genotype will undergo mutation. A higher mutation rate increases the level of exploration, while a lower rate reduces the likelihood of disrupting well performing solutions. Because this approach treats all genes equally in terms of their probability for mutation. This means that even genes that are performing well and contributing positively to an individual's fitness have the same chance of being mutated as genes that are performing poorly. It can slow the overall progression of the algorithm. Therefore we used an adaptive mutation rate for each gene instead of a universal mutation rate. This implies that genes that perform well and contribute positively to the fitness of an individual are less likely to undergo mutation, preserving optimal gene sequences and improving the convergence speed of the algorithm. In the conventional mutation example described above, each bit has the same probability, denoted as Pm, of being flipped. However, in our adaptive mutation function, the probability of mutation for each gene is inversely proportional to its success rate. Hence, for an individual with x genes, the mutation operation could alter the second gene with a probability of mutation unique to that gene's success rate. This adaptive mutation operation manages to strike a delicate balance between preserving high-performing genetic structures and promoting diversity in the population. While a higher mutation rate for less successful genes increases the level of exploration and helps in escaping local optima, a lower rate for successful genes minimizes disruption of well performing solutions, effectively enhancing exploitation.

Crossover and mutations are only applied withing a single transformation's parameters, we do not allow mutations spanning across different transformations to avoid inconsistency thus breaking the validity check which might result in the individual's genotype invalidity. During mutation and crossover some chromosome or genes for a given individual may be deactivated or activated resulting in different new transformations. For example the algorithm may progressively focus on specific part of the signal, this is also due to the selection process.

Selection algorithm The selection process is a critical step that guides the optimization of individuals in the population, ultimately leading to the discovery of minimal perturbations that flip the model's predictions. During the selection process, individuals are chosen from the current population based on their fitness scores, which represent the quality of the solution in terms of their proximity to the model's inter-class decision boundary and the minimal perturbation required to flip the prediction. Higher fitness scores indicate that an individual's perturbations are more meaningful and informative, making them more likely to be selected for reproduction. The selection process can be implemented using various techniques, such as tournament selection, roulette wheel selection, or rank-based selection, each with their own advantages and tradeoffs. Tournament selection, involves selecting a subset of individuals from the population and choosing the one with the highest fitness score, while roulette wheel selection assigns probabilities proportional to an individual's fitness score and selects individuals based on a random draw. In rank-based selection, individuals are sorted according to their fitness scores, and selection probabilities are assigned based on their ranks. The chosen individuals then undergo genetic operations, previously discussed, to produce offspring for the next generation. We choose the tournament selection algorithm for our method, as it offers a good balance between exploration and exploitation, which can be beneficial in finding minimal perturbations.

For a population of n individuals we performs n tournaments, the tournament size ts is a fixed hyper-parameter. During each iteration of tournament selection, a fixed number of individuals (the tournament size) are randomly selected from the population to compete against each other. The individual with the highest fitness score among the competitors is declared the winner and proceeds to the reproduction stage, where crossover and mutation operations are applied. The choice of tournament size has a direct impact on the selection pressure in the genetic algorithm. With a larger tournament size, the probability of selecting the best individuals in the population increases, leading to a higher selection pressure. This promotes exploitation of the best solutions, but may also reduce population diversity and increase the risk of premature convergence. Conversely, smaller tournament sizes result in lower selection pressure, as weaker individuals have a higher chance of being selected for reproduction. This helps maintain diversity in the population and encourages exploration of the solution space, but may slow down the convergence of the algorithm. The random selection process used in tournament selection ensures that each individual has a chance to participate in one or more tournaments, and the best individual from each tournament is selected for reproduction. The resulting winners of these tournaments could potentially be different, even if the same individuals participate in multiple tournaments, as their chances of winning will depend on the specific competitors in each tournament. Although this method can result in slow convergence rate, it's essential to strike a balance between convergence speed and diversity in the population. If the algorithm converges too quickly, it can get stuck in local optima and miss out on better solutions that might be discovered through further exploration of the solution space. Conversely, if the convergence is too slow, the algorithm might take an excessive amount of time to find a satisfactory solution.

This process enables the method to effectively explore the space of possible perturbations while simultaneously exploiting the most promising solutions. By using tournament selection, the genetic algorithm can efficiently search for individuals with minimal perturbations that flip the model's predictions, while also maintaining a diverse population to avoid premature convergence and explore different perturbation combinations.

Fitness function We evaluate the fitness of each individual based on their proximity to the model's inter-class decision boundary and how minimal the perturbation is that it causes a flip to the opposite class. For that, for a given individual with its set of transformed ECG input, we compute a distance between the target decision boundary and the prediction after applying the transformations to the input data. We also evaluate how far the transformed ECG is from the original one. This distance is minimized to force the algorithm to find smallest useful transformations We define the fitness function as:

Equation 3.1 Evocclusion fitness function

$$Fitness(I) = (\alpha * (1 - Proximity(I))) + (\beta * (1 - L2Distance(I))) \tag{3.1}$$

where:

- I is the individual to be evaluated.
- $Proximity$ is a function that measures the closeness ($distance\_score$ of the individual's perturbed input to the model's inter-class decision boundary. A lower value indicates that the perturbed input is closer to the decision boundary and thus the individual is more effective in finding minimal perturbations that flip the prediction. The decision boundary is a hyper-parameter, as the middle 0.5 might also indicates that the model is not able to classify the input in either class. The term is detailed in Algorithm 2, $\hat{ecgs}$ denotes the transformed ECGs.

Algorithm 2 Evocclusion: Algorithm of the $Proximity$ function used in the fitness

function PROXIMITY($model, ecg, \hat{ecgs}, decision\_boundary$)
  $proximity\_values \leftarrow []$
  for $\hat{ecgs}_i$ **in** $\hat{ecgs}$ **do**
    $prediction \leftarrow$MODEL($\hat{ecgs}_i$)
    $proximity \leftarrow$DISTANCE_SCORE($decision\_boundary - prediction$)
    $proximity_values \leftarrow$APPEND($proximity\_values, proximity$)
  end for
  return MEAN($proximity\_values$)
end function

The $distance\_score$ is computed as follow:

$$\text{distance\_score} = \begin{cases} \frac{\text{decision\_boundary} - \text{prediction}}{\text{decision\_boundary}} & \text{if prediction} < \text{decision\_boundary} \\ \frac{\text{prediction} - \text{decision\_boundary}}{1 - \text{decision\_boundary}} & \text{otherwise} \end{cases}$$

- $L2Distance$, this term calculates the L2 distance between the averaged perturbations and the original input. It aims to penalize individuals with perturbations that deviate too much from the original input. It encourages the algorithm to find perturbations that are closer to the original input, making the perturbed examples more interpretable and meaningful.
- α and β are weights to balance the contributions of perturbations importance, proximity and minimal perturbation to the overall fitness score. They are hyper-parameters.

As the fitness function is to be maximized, we subtract the proximity and minimal perturbation terms from 1. Therefore, individuals with higher proximity values, more important perturbations and less amount of perturbations will have higher fitness scores.

At the last generation, to get the interpretability from the individual's perturbations, we compute attribution maps similarly to the integrated gradients method. In fact, we capture the gradients variations of the neurons as we feed the final generation best individuals transformations to the network and apply the integrated gradients process. Because the overall computations is expensive, we implemented the whole method in Cython and C which is then compiled in binary modules. The method favors running on GPUs and optimizes the memory using model sharing technique and parallel programming.

3.4 Results

We ran several experimental scenarii in order to find adequate parameters. In this section we present two of our most informative scenarii and we focus on occlusion transformations. Both share the same parameters and genome. The first scenario involves running the algorithm on each ECG at different protocol times (i.e. classes). This process results in an adapted interpretability for each ECG rather than an interpretability related to the class. On the contrary, state-of-the-art methods such as saliency maps or occlusion provides class based interpretability as they apply same transformations (occlusion) to all ECGs uniformly. Table 6 presents the genome where as Table 7 describes the algorithm parameters configuration.

Transformer	**Min Instances**	**Max Instances**	**Initial Instances**	**Parameters**		
				Window Size %	**Relative Position %**	**Beat**
single_beat_p_wave_occlusion	1	200	200	[5 - 25]	[0 - 98] %	[0 - 20]
single_beat_t_wave_occlusion	1	200	200	[5 - 25]	[0 - 98] %	[0 - 20]
single_beat_qrs_complex_occlusion	1	200	200	[5 - 25]	[0 - 98] %	[0 - 20]
single_beat_no_wave_occlusion	1	200	200	[5 - 25]	[0 - 98] %	[0 - 20]

Table 6: Experiment genome: *we only use occlusion based transformation on individual beat. The transformations also target specific parts of the beat.*

The position of P, T waves and QRS complex are computed using our segmentation models. We optimized the code using C and Cython source code. The average computing time for 300 generations is 5 minutes compared to previously 3 hours with the initial implementation. The use of adaptive mutation rate per gene increased the convergence time of the method, while avoiding being stuck in local optima. In fact, the mutation rate has the ability to lower and increase over time, taking into consideration population performance in previous generations. Individuals of the first population are spawned with the maximum number of genes allowed in order to let the algorithm decide which genes are most useful to the task. On one ECG per training, the method gives high performance with a fitness generally around 95% with parameters α and β of the fitness being respectively 0.7 and 0.3. Table 8 presents fitness performances of the first scenario on single ECG at each protocol time. Evocclusion was able to find meaningful transformation to flip the prediction of the neural network close to the opposite class while minimizing the required transformations on the ECG signal (due to the L2 term in the fitness).

Name	Value	Description
Generations	300	Number of generations to iterate
Population size	150	Number of individuals in each population
History size	30	Number of previous generations to consider when computing gene success rate
Gene mutation rate moving average factor	0.7	Used when computing mutation rate of genes based on their success rate, if the value is high the newly computed mutation rate will have more impact, if it's low, previously computed mutation rate will have more impact
Gene mutation rate clamp	[0.01 - 0.8]	Upper and lower limits of genes mutation rate
Crossover method	swapping	The crossover method, we swap genes thus creating new offspring, the probability of swapping is weighted, if success_rate(gene1) > success_rate(gene2): swap_prob = [0.3, 0.7]
Prediction level target	Sot+ : 0.6, Sot- : 0.4	The prediction score we want to reach
Selection method	Tournament	Selection method for generating offspring
Tournament size	8	Size of tournaments
Elitism proportion	10%	Enables elitism, proportion of best individuals to keep without genetic modifications

Table 7: Experiment configuration: *genetic algorithm parameters setup.*

Protocol Time	Min Fitness	Max Fitness	Processing Time
Basal	0.6	0.90	5min
Inclusion	0.7	0.91	5min
SotT0	0.75	0.93	5min
SotT1	0.8	0.93	5min
SotT2	0.82	0.94	6min
SotT3	0.86	0.98	7min
SotT4	0.81	0.97	6min
SotT5	0.7	0.95	6min
SotT6	0.85	0.97	6min

Table 8: Experiment 1 - Fitness results: *fitness of best and worse individuals at the last generation for each ECG training. Protocol time basal is prior (up to 1h before) to injection of Sotalol, inclusion is right before injection, SotT0 is recorded just after injection and SotT1 to T6 are 1 to 6 hours after injection of the drug.*

Despite the good performances reached, the final interpretability results wouldn't be comparable

to other state-of-the-art methods as Evocclusion computes specific optimal transformations for each ECG where as other methods uses the same transformations for all ECGs. To make the results effectively comparable, we relaunched the same experiment as above but for each protocol time we used 100 ECGs instead of 1, the resulting best transformations (i.e. one for each class) are used on all ECGs. The proximity calculation *(in the fitness)* for each transformation is done in respect of the corresponding ECG. This approach allows capturing common transformations for each class instead of a single ECG thus allowing us to compare the results with state-of-the-art methods. We used the same genome and configuration. Table 9 presents the fitness of the last generation with 100 ECGs. A significant drop in performances can be noted specially for Sot- protocol times. These results can be justified by the differences in the waveform for each ECG and the fact that the model might not be looking at the same parts for each ECG. In Figure 29 we plot the last generation best individual occlusion windows genes on the signal.

Protocol Time	Min Fitness	Max Fitness	Mean Fitness	Std Fitness	Processing Time
Basal	0.25	0.43	0.29	0.06	1h
Inclusion	0.25	0.44	0.29	0.05	1h
SotT0	0.28	0.68	0.46	0.09	1h
SotT1	0.34	0.77	0.54	0.1	1h 30min
SotT2	0.29	0.74	0.43	0.1	1h 20min
SotT3	0.34	0.91	0.6	0.13	1h 40min
SotT4	0.26	0.61	0.37	0.09	1h 20min
SotT5	0.28	0.65	0.43	0.09	1h 25min
SotT6	0.31	0.79	0.53	0.11	1h 20min

Table 9: Experiment 2 - Fitness results: *fitness of best and worse individuals at the last generation for each ECG training. We observe a general performance drop specifically in Sot- protocol times where the method struggles finding common powerful transformations for the 100 ECGs.*

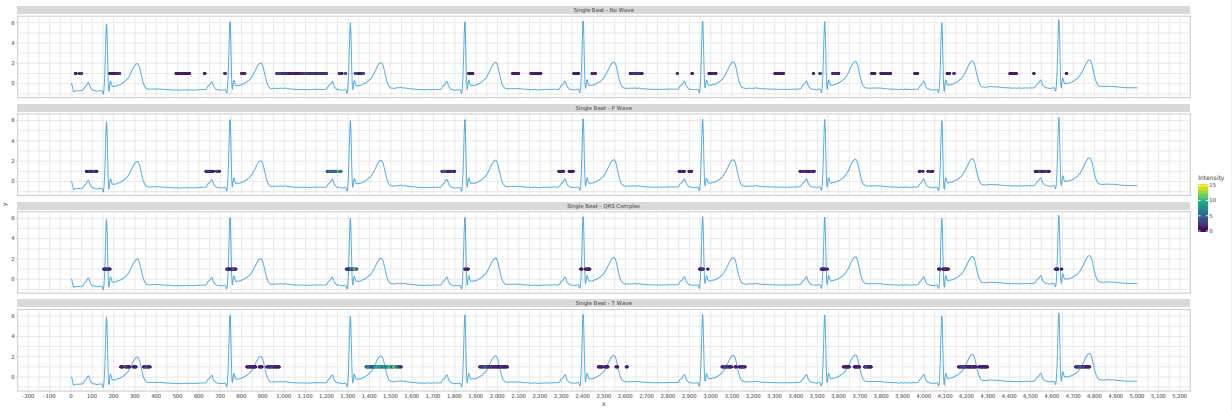


Figure 29: ***Experiment 2 - Occlusions genes representation - SotT3 protocol time****: a unique set of transformations is obtained for each protocol time. We can notice the genes are different across beats which might indicate the model's prediction is not based on all beats or same parts on each beat.*

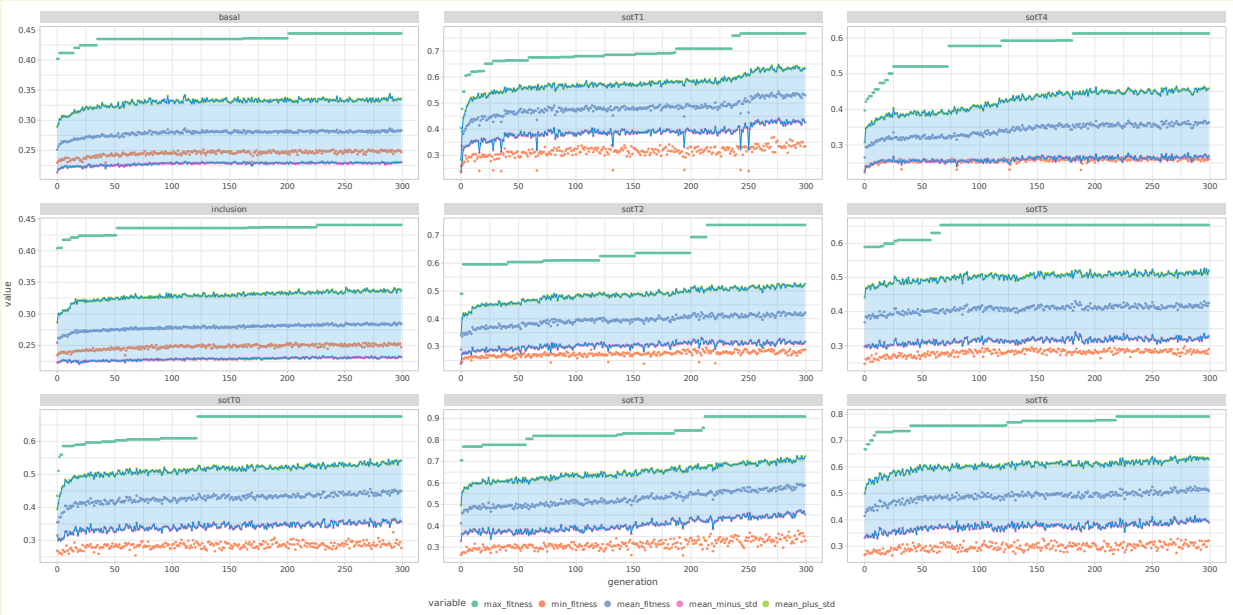


Figure 30: ***Experiment 2 - Fitness evolution****: evolution of the fitness over the 300 generations for each protocol time.*

In Figure 30 we plot the evolution of the fitness over generations, we can notice for each protocol time the fitness does not improve much over generations. In fact the method faces difficulties finding common power transformations for the 100 ECGs where as in single ECG mode the method rapidly finds powerful and meaningful transformations. Protocol time SotT3 presents the best fit performance. We also analysed the evolution of the proportions of activated genes for each gene type over generations for each protocol time, results are presented in Figure 31. The number of activated genes drops to approximately 100 for each gene type, their proportions are similar as shown in Figure 32.

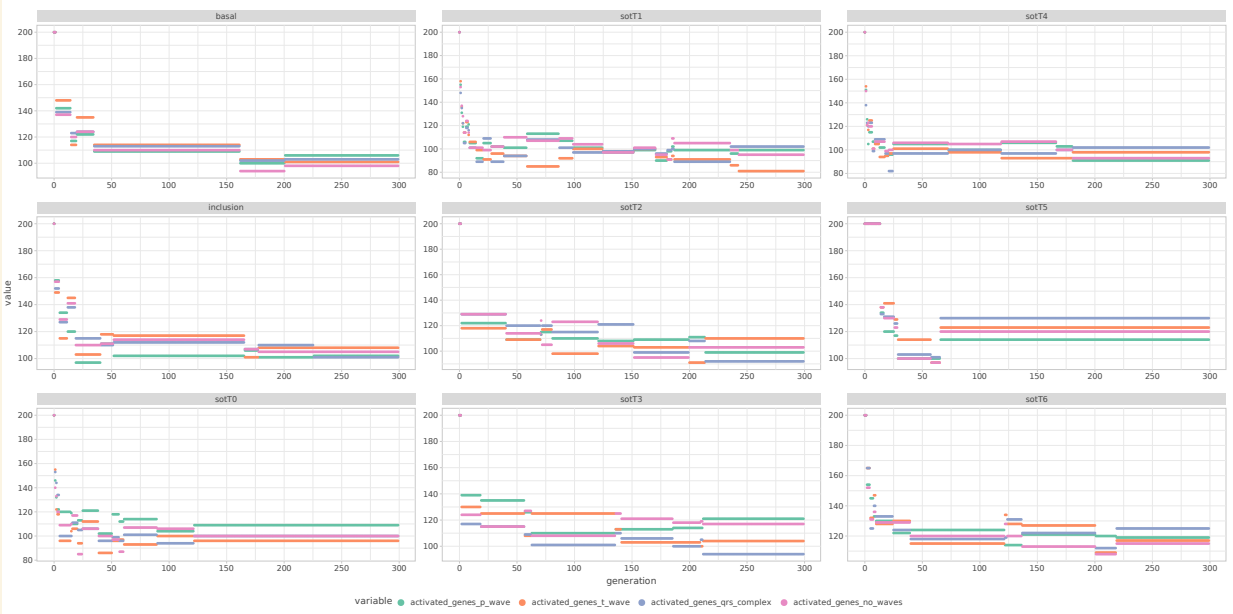


Figure 31: ***Experiment 2 - Activated genes evolution****: evolution of activated genes per category over the 300 generations for each protocol time.*

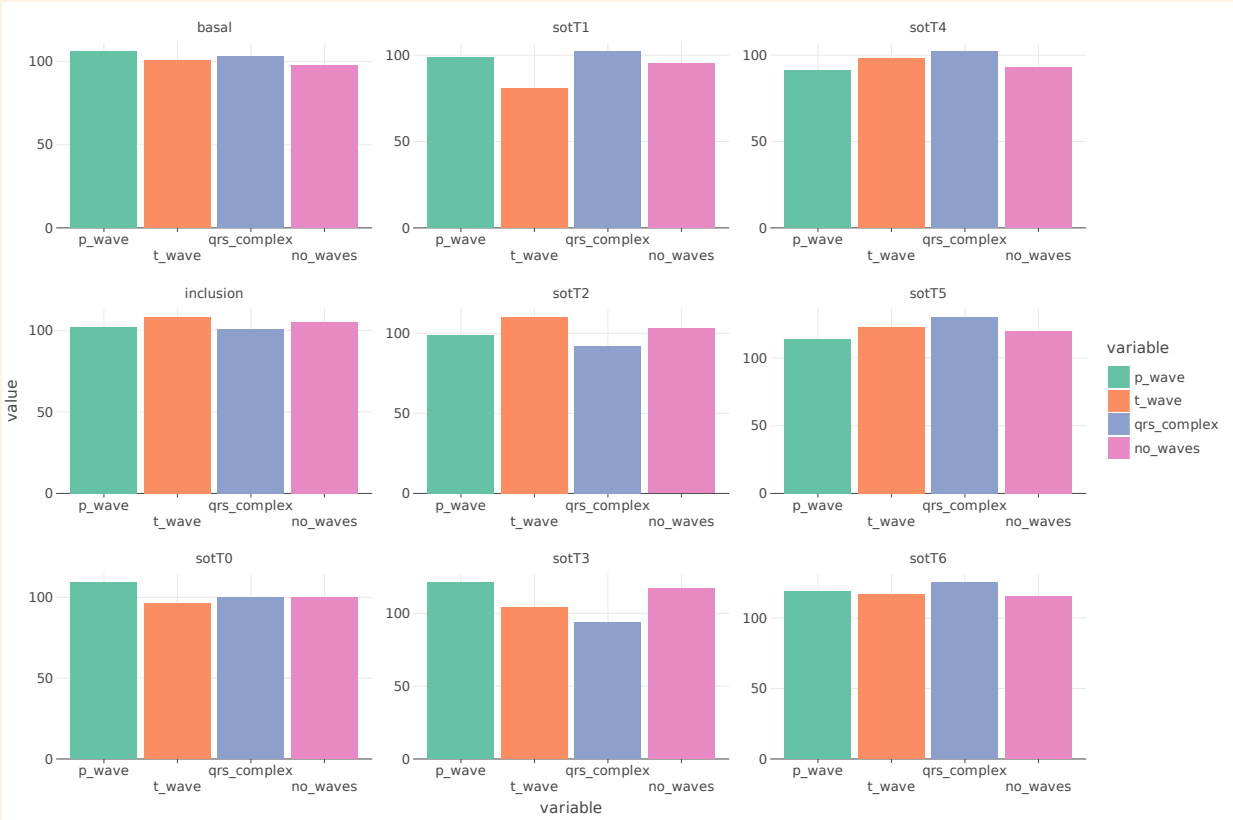


Figure 32: ***Experiment 2 - Activated genes proportions****: the number of activated genes at last generation for all protocol times are close for all categories.*

Feature importance and interpretability results To access the impact of the transformations identified by the method on the model's decision process we analysed the gradients of the neurons after feeding to the model the transformed ECGs. We did so by using the integrated gradients method for each category of genes at each protocol time which generated attribution maps. Results are presented in Figures 33, 34, 35, 36. The features importance vector is not standardized nor scaled to accurately compare the importance across protocol times thus justifying the absence of visible signal on some protocol times as the relative amplitude is low compared to others. When we compare these results to prototypes differences of each protocol time we can notice similarities particularly at the end the P Wave at SotT3. In fact, it has be shown on prototypes that the P Wave length changes significantly 3 hours after Sotalol injection which is also confirmed by Evocclusion. The start and end of the T wave also significantly impact the model's decision process as observed on the prototypes. The end of the QRS complex, specifically the S peak also changes across time with Sotalol as seen on prototypes, these variations are also spotted by Evocclusion. Parts of the ECG on the inter-beats signals are also important to the model's prediction. In our previous published work *(Chapter 2)* on predicting TdP and after applying standard occlusion interpertability we identified similar regions but with less precision, Evocclusion allows us to better emphasize important regions for the model's predictions.

Figure 33: ***Experiment 2 - Features importance P Wave****: features importance computed using integrated gradients method.*

Figure 34: ***Experiment 2 - Features importance T Wave****: features importance computed using integrated gradients method.*

Figure 35: ***Experiment 2 - Features importance QRS Complex****: features importance computed using integrated gradients method.*

Figure 36: ***Experiment 2 - Features importance No Wave****: features importance computed using integrated gradients method.*

These plots, primarily present the relative importance of some of the signal regions to the model, another form of interpretability is assessing the impact of identified transformations on the model's prediction rather on the gradients. To do so, we computed the delta between the original prediction without perturbations and the ones after transforming the ECG. Results are displayed in Figure 37. Parts of the ECG signal, which are overlapped with negative interpretability signal are considered as not contributing to the target class, similarly to the occlusion interpretability in our previous work (TdP risk prediction). In this plot we also kept the relative amplitude between protocol times, no scaling nor standardization are applied to the interpretability vector. We can clearly observe a negative to positive variation of the score across time. The target class we are interpreting is Sot- for basal, inclusion, SotT0 and Sot+ for the rest. In time basal and inclusion where the Sotalol has not been

administered yet, we can observe regions that are not contributing to the prediction, the same regions are known to contribute to the predictions of Sot+. This observation is confirmed in SotT1-6 scores, where the end of the P wave, and parts of the T wave and QRS complex are positive. This form of interpretability correlates with the previous features importance results we presented earlier as well as with the differences in prototypes across time. When we compare these results with standard occlusion technique, Evocclusion brings more details and precise annotation of the input data to better understand the model's decision process by exploring a multivariate universe of transformations of the data in order to find significant ones and from them derive relevant markers.

Figure 37: ***Experiment 2 - Interpretability using prediction scores variations****: we use identified transformations at last generation in the best individual and compute the model's prediction delta with original prediction score.*

Comparison with state-of-the-art methods We selected two state-of-the-art methods to compare with, saliency maps and standard occlusion, although there are various interpretability methods. These methods have been widely applied in medical diagnosis. Saliency maps, which visualize the gradients of model predictions with respect to inputs, are an intuitive way to highlight the significant features that contribute to the model's predictions. They provide a general understanding of model behavior and have been particularly successful in interpreting image based models. Standard occlusion, on the other hand, offers a more direct approach to interpretability. It systematically occludes parts of the input and measures the effect on the model's output, providing an empirical measure of feature importance. Both these methods have the benefit of simplicity and robustness, making them useful benchmarks for comparison with our approach, Evocclusion.

In our analysis, we computed saliency maps and performed standard occlusion on patient's ECGs. The comparison was conducted on two key aspects: time relative interpretability, as seen in Figure 38, and standardized interpretability, as shown in Figure 39. These results are extraction obtained on one patient's ECGs. Our proposed method, Evocclusion, displayed an intriguing discrepancy from the standard occlusion approach. Particularly, it emphasized areas physiologically associated with prolonged QT as negatively impacting the model's decision, whereas standard occlusion perceived these same zones as contributors to the prediction of Sot-. This divergence emphasizes the challenges inherent in understanding the interpretability results derived from standard occlusion. In contrast to Evocclusion and occlusion, saliency maps provide an indirect form of interpretability. They rely largely on the variations of gradients, which makes their interpretation less straightforward. While

saliency maps underscore the regions that are significantly crucial for the model's decision making process, they fail to provide a friendly understandable interpretive annotation whether a given ECG segment plays an important role in predicting the target class. Evocclusion rises above these limitations by providing a more comprehensible interpretability. Unlike standard occlusion, which operates on the principle of sequentially occluding a specific part of the signal, hence failing to account for the interdependence between different signal regions; Evocclusion explores a broad range of combination possibilities thus taking into consideration potential combined regions that collectively influence the model's prediction.

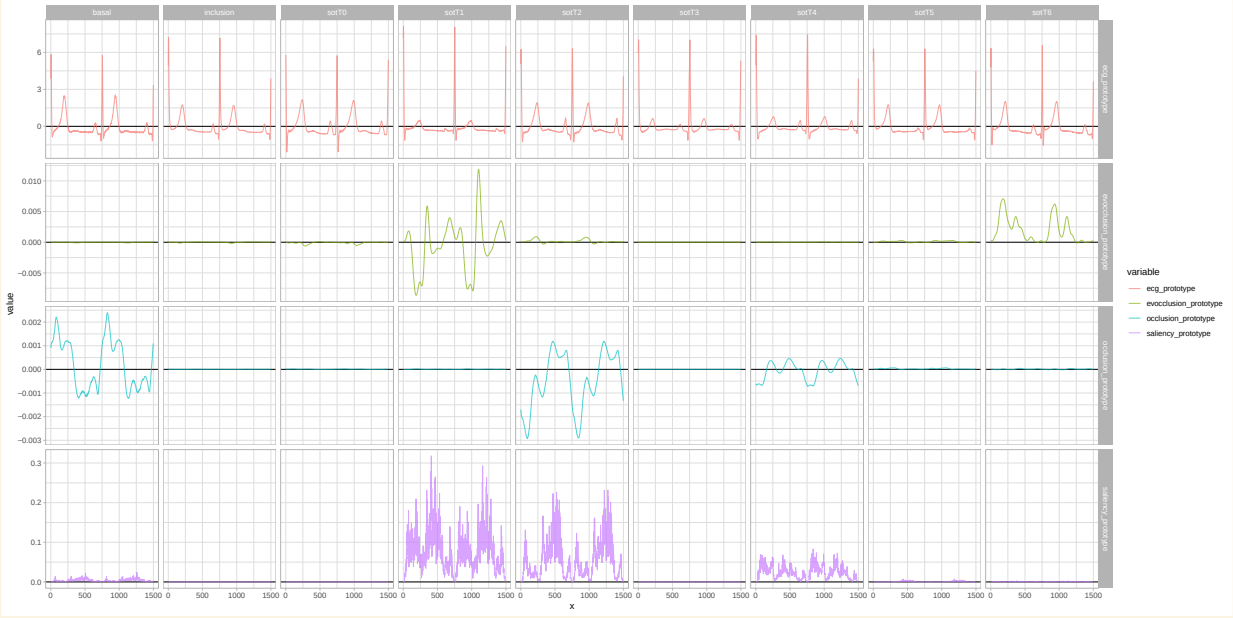


Figure 38: ***Evocclusion comaprison with SOTA methods - time relative****: we kept the interpretability values relative to protocol times to emphasize differences over time.*

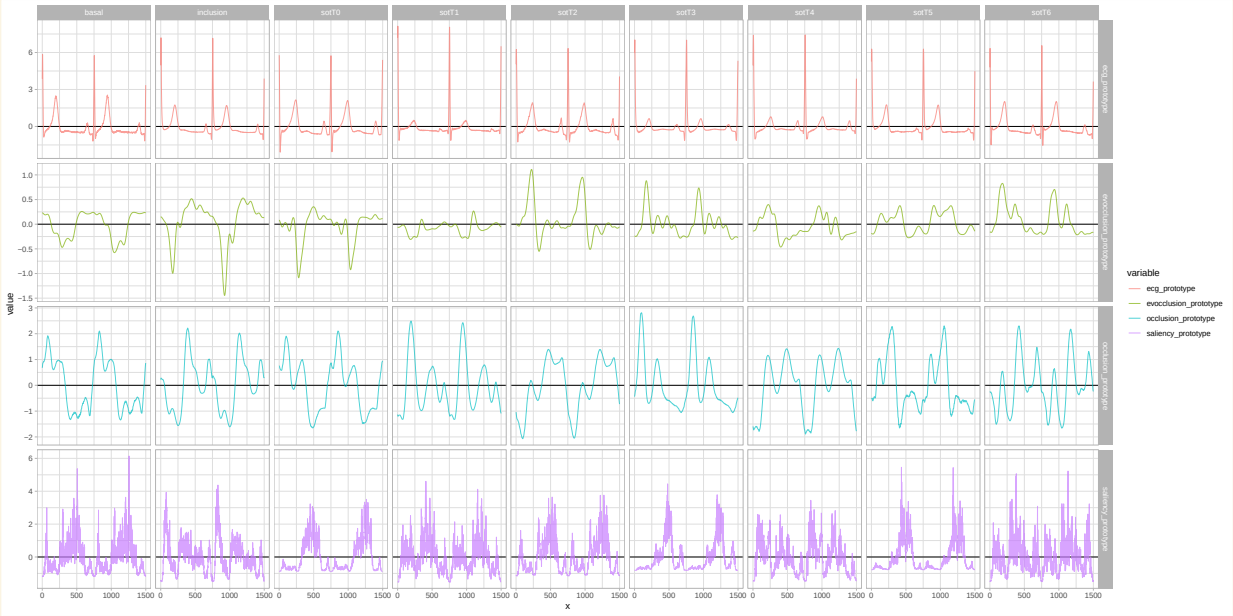


Figure 39: ***Evocclusion comaprison with SOTA methods - standardized****: we standardized each interpretability vector.*

3.5 Conclusion and Perspectives

Enhancing interpretability methods in machine learning and deep learning is crucial particularly in the clinical context. In fact, despite the progress and great performances of recent AI methods in the medical field, the trust level towards those methods from clinicians and practitioners is still low. This mainly comes from the lack of understanding of the model's prediction. Trust is a cornerstone for the acceptance and widespread application of AI models in the healthcare sector. By providing clear, comprehensible insights into their decision making process and by providing human friendly annotations of the input data, clinicians as well as researchers are more likely to have confidence in the outcomes and predictions generated by these models. This has far reaching implications for patient care, from more accurate diagnoses to personalized treatment plans. Moreover, enhancing interpretability can also lead to the discovery of new patterns or correlations in medical data that might not have been apparent with traditional analysis techniques.

The field of interpretability has rapidly improved over the past few years leading to a new domain, Explainable AI. Explainable AI, often referred to as XAI, aims to make complex machine learning algorithms more understandable, interpretable, and, in essence, explainable to a human audience. The goal of XAI is to create a system where both the output and the decision making process of AI models are clearly understandable by humans. This is particularly relevant in fields like healthcare where understanding the 'why' behind a model's decision is equally, if not more, important as the decision itself. In this study, we aim at improving the field of interpretability of machine learning models, specifically within the context of bio-signals,electrocardiograms. Our main contribution is the creation of a new innovative method of interpretability, Evocclusion, which has its foundation on state-of-the-art approaches and which we tailored to provide more understandable interpretability annotation of model's prediction.

Evocclusion has been designed as a significant improvement of traditionnal univariate occlusion method. Indeed, occlusion method usually relies on hiding sequentially one part of the data at a time and observing how the model's prediction varies. This method might not fully capture interdependencies among different regions of the data. As a result, it can overlook the combined influence of multiple regions on the model's prediction. Furthermore, most gradients-based methods, like saliency maps or integrated gradients highlight important regions influencing the model's decision by examining changes in gradients. Yet, this technique offers an indirect form of interpretability, leaving some questions about how particular parts of the data contribute to the prediction of the target class.

Our proposed method provides a multivariate alternative which bridges these gaps. It is based on adversarial attacks using data transformations. Because of the broad possible combinations of transformations, the method uses a genetic algorithm approach thus exploring this wide range and taking into account the features inter-dependencies. Evocclusion operates by applying different type of perturbations on the data and assessing how they affect the model's output. We have built tools to accurately identify relevant transformations on specific part of the data. We developed and used our denoising and segmentation tools to precisely delineate critical segments of the signal on ECG data. These methods allow Evocclusion to target these areas and focus on important parts of the signal while still exploring the rest of the ECG to capture meaningful and possible undiscovered patterns. One key aspect of the approach is the nature of the transformations applied to the data. Domain-specific transformations are important to better understand the model's output as they provide a better understanding of the output interpretability. In our study, we developed a data prototype based approach which provides accurate representation of a particular group/class of ECGs and per patients. These prototypes are used to transform the ECG in a clinically valid way. Although we presented occlusion transformations results in this chapter, we explored the prototypes based transformations which provided interesting and informative outputs which are still under analysis in close collaboration with cardiologists. However, the occlusion based transformations still provided accurate results which are understandable by clinicians, highlight features inter-dependencies and are clinically relevant.

Reflecting on these findings, we do recognize the need for continued evolution and refinement of

our method. Although it showed significant advancement, Evocclusion has its own limitations; one of which is the risk of out-of-distribution impact which might be caused by occlusion-based transformations. In fact, replacing occluded parts by zeros or mean value can lead to out of distribution scenario, therefore we focus our efforts in prototypes transformations which are provided more clinically understandable and interpretable results. This approach is currently under validation and improvement. Computing specific interpretability annotations for each ECG can be resources intensive. Therefore we optimized the algorithm and used low level programming language to reduce the consumption footprint while maintaining a short computing time. Finding optimal parameters for the algorithm also remains an important challenge.

Despite these challenges, the potential of Evocclusion is considerable, it can provide adaptive patient's data interpretability. Our vision for Evocclusion goes beyond ECGs, in fact, Evocclusion is part of bigger framework namely NeuralXplain. NeuralXplain is a framework we initiated in this study to provide interpretability and also explicability not only in the clinical field but in a more general way including images and other data types.

Chapter 4

Denoising and Restauration of Electrocardiograms with Autoencoders

Chapter Summary

4.1 Introduction

ECGs are obtained using electrodes, which are placed on different parts of the chest. Several visible segments can be found on an ECG. The right and left atria or upper chambers make the first wave called a "P wave", followed by a flat line when the electrical impulse goes to the bottom chambers. The right and left bottom chambers or ventricles generate the next wave called the "QRS complex" when they contract strongly to pump the blood in the periphery of the organism. The final wave or "T wave" represent the electrical recovery or return to a resting state of the ventricles. The diagnoses of several electrical cardiac conditions such as arrhythmia can be performed by experts by analysing ECG signals. For this, it is important to accurately identify individual components in the ECG. In current clinical practice, health practitioners, especially cardiologists, look for subtle abnormalities in wave morphology and the periodicity of repeating features measured "by hand" to provide diagnosis. New methods have been developed to automate several ECG processing tasks, including pathology detection, annotation, measurements and more recently, interpretability. These approaches are mostly based on neural networks. ECG signal is input in the NN, which perform operations to obtain different novel abstract representations of the signal where novel features are extracted. However, automated ECG analysis techniques as well as manual ones can be very challenging when it comes to noisy

ECG. For instance, noisy ECGs are extremely difficult to segment manually and automatically for waves (P, QRS, T) analysis. Indeed, ECGs are frequently recorded with noise, coming from difference in recording equipment, placement of electrodes, respiration, interference arising from sudden movements, muscle noise, ambient electrical interference, patient heterogeneity, etc. Approaches for denosing ECGs include digital filters or more recently deep learning based methods. Despite being generally effective for signal denoising, digital and wavelet filters are not particularly designed for ECG waveforms. They tend to fail especially on highly noised ECGs or deteriorated signals where part of the signal are missing or corrupted. Custom crafted neural networks were shown to be efficient on ECG denoising. However, they tend to remove or temper important features of the signal like the amplitude of the S peak or T wave which are important for detecting some pathologies or drug intake as [46] showed in their paper. In this study, we propose a novel and robust deep neural network for ECG denoising: **DeepFADE (Deep Frequency Amplitude Denoising Encoder)**. We designed a DenseNet-like [134] denoising autoencoder [135] (DAE), which cancels noise and reconstructs malformed waveforms of the signal while minimising the alteration of the relative amplitude of waves to the isoelectric line (baseline). We evaluated the approach on different level of noise (-6dB, 0dB, 6dB, 12dB, 18dB, 24dB) combined with baseline wander. Our method successfully denoised the signal with an output signal-to-noise ratio (SNR) of 16dB, which surpassing SOTA methods.

4.2 Related Works

ECG analysis that both focuses on the heartbeat morphology as well as its rhythm is a very powerful tool in diagnosing certain cardiac conditions. The most important regions in the heartbeat are the P wave, the QRS complex, which comprises the Q, R and S peaks and the T wave. Most ECG recordings are subject to different source of noise, which can interfere with the detection of classification of cardiac diseases. In 2021, Prifti et al. [46] proposed a novel DL approach based on a DenseNet [134] architecture to evaluate the occurrence risk of a particular type of arrhythmia, Torsade-de-Pointes (TdP) [136] [137] [46]. The method topped a 98% accuracy in predicting the footprint of a drug known to increase the risk of TdP. To train their model they used a relatively clean ECG database, however when tested on noisy ECGs the model's performance dropped due to the interference of the noise. When it comes to denoising ECG signals, there are several methods currently in use [138] [139]. One of the most popular methods is the use of digital filters, such as the Butterworth filter [140] and the median filter [141]. These filters are commonly used to remove high-frequency noise and baseline wander, respectively. Another popular method is the use of wavelet transform-based methods [142], such as the wavelet thresholding method and the empirical mode decomposition (EMD) method. These methods use the properties of wavelets to separate the noise from the signal. Recently, there has been an increasing interest in the use of deep learning-based methods for denoising ECG signals. One popular method is the use of deep neural networks (DNNs), such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs). These methods have been shown to be effective in removing various types of noise and preserving the important features of the ECG signals. Another popular method is the use of generative models, such as the generative adversarial network (GAN) [143] and the variational autoencoder (VAE). An autoencoder (AE) is a type of neural network that is used for dimensionality reduction and feature learning. It is composed of two main parts: an encoder network, which maps the input data to a lower-dimensional representation, and a decoder network, which maps the lower-dimensional representation back to the original input space. The goal of training an autoencoder is to learn a set of weights for the encoder and decoder networks such that the input data can be accurately reconstructed from the lower-dimensional representation. The structure of an autoencoder is illustrated in Figure 40. The VAE is a type of generative model based on a AE, that is used to learn a compact representation of the underlying probability distribution of a dataset, its structure is illustrated in Figure 41. The key difference between a VAE and a traditional autoencoder is that the VAE is trained to learn a probabilistic latent representation, rather than a deterministic one. This allows the VAE to generate new samples from the learned distribution by sampling from the latent space and passing the samples through the decoder network. These models have been

shown to be effective in removing noise while preserving the structure of the ECG signals. In their paper, [144] introduced a method to denoise ECGs based on a denoising autoencoder neural network. They trained their model on the Physionet MIT-BH NSTDB dataset which features ECGs with and without noise, and with real life noise artefact such as baseline wander, muscle artifact, and electrode motion. They successfully achieved a Signal to Noise Ratio signal reconstruction of approximately 15.9dB on the Physionet dataset. Despite the progress that has been made in denoising ECG signals, there are still several challenges that need to be addressed. One of the main challenges is the preservation of important features of the ECG signals, such as the QRS complex and the T-wave. Another challenge is the generalization of the denoising methods to different types of noise and different populations. To overcome these challenges we hypothesized and developed a new approach based on a neural network.

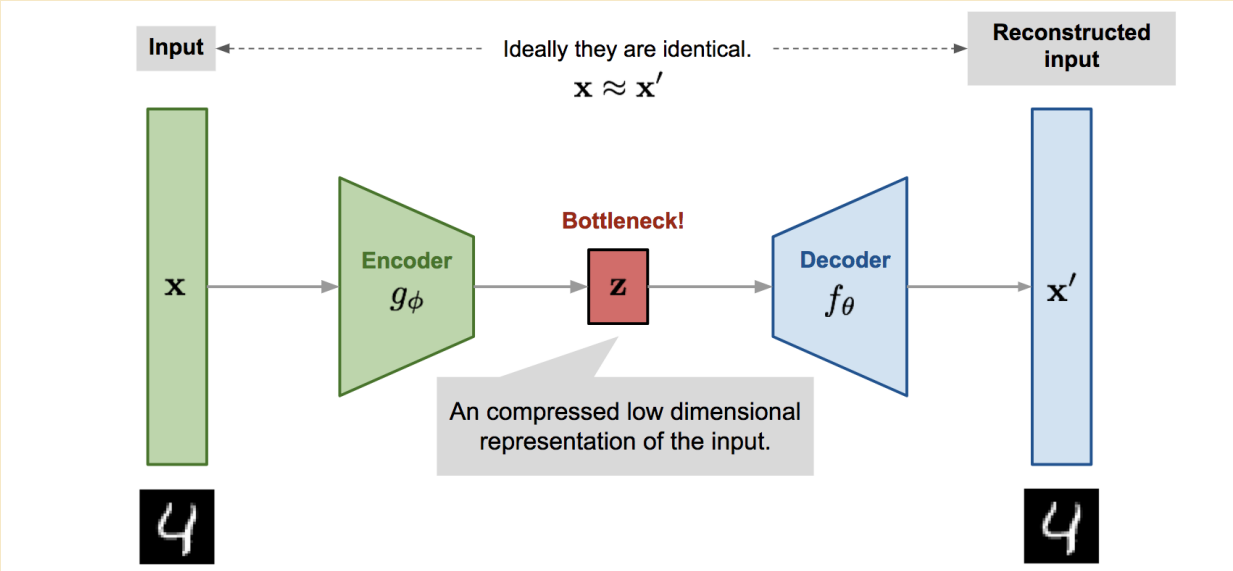


Figure 40: ***AutoEncoder architecture:*** *is a type of artificial neural network used to learn efficient, lower-dimensional representations of input data. It comprises an encoder that compresses the input into a latent space, and a decoder that reconstructs the input from this space. The goal is to minimize the difference between the original and reconstructed input. Autoencoders are useful for dimensionality reduction, denoising, and more. Image extracted from [4]*

4.3 Methodology

The method is based on an autoencoder (AE) architecture. AEs which are commonly used in denoising tasks are called Denoising AutoEncoders (DAE). DAEs learn feature vectors that characterize the input and are invariant to noise. Unlike vanilla AEs approaches, with DAEs, noise is added to the input data and the network learn to undo this corruption rather than just reconstructing the same input. The standard architecture of DAEs is illustrated in Figure 42. In this study, we implemented a DAE which takes as input an ECG denoted x, consisting of a vector of 5000 timepoints. We then corrupt the input with noise to $\tilde{x}$ and feed it to the DAE to compress it through convolution and average pooling layers to an abstract representation of 25 points. Embedding h is reconstructed back to 5000 timepoints while cancelling the noise and the output is denoted $\hat{x}$. To achieve a dimensionality upscale from 25 points to 5000 points, the decoder performs a reverse convolution operation known as deconvolution to approximate initial values previously processed in the symetric corresponding encoder layer without the initial noise. The DAE network is based on the DenseNet architecture [134].

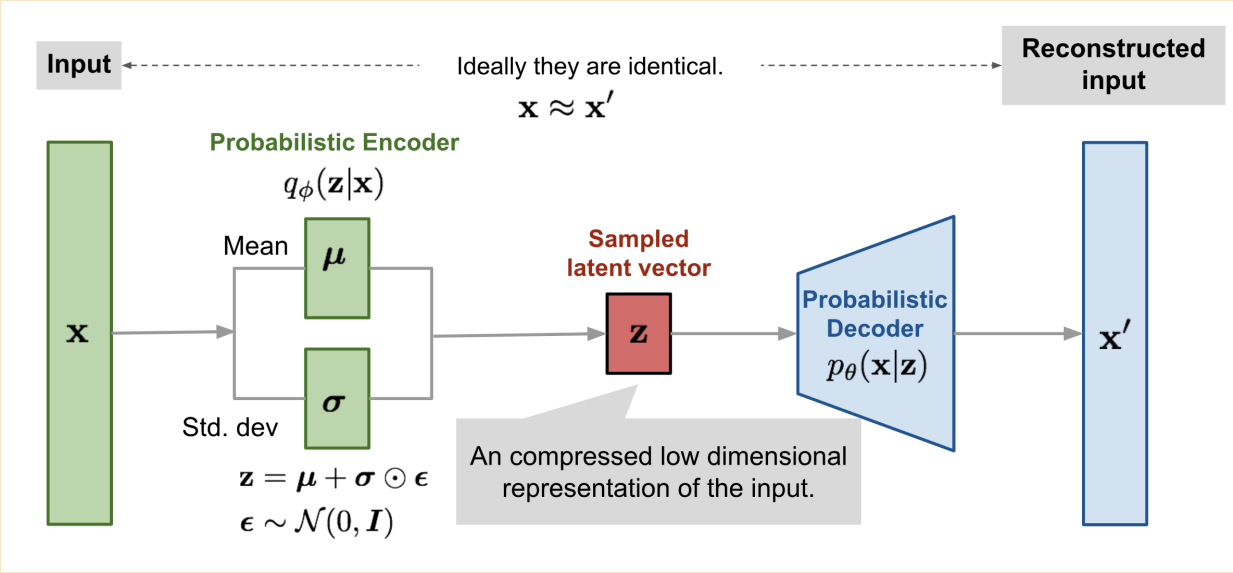


Figure 41: ***Variational AutoEncoder architecture:*** *is a type of autoencoder used in machine learning for generating new data. Unlike standard autoencoders, which map input data to a fixed point in a latent space, VAEs map input data to a distribution. This distribution allows VAEs to generate new data that resemble the training data, making them useful for generative modeling tasks. The "variational" part comes from their use of variational inference techniques in the encoding process. Image extracted from [4]*

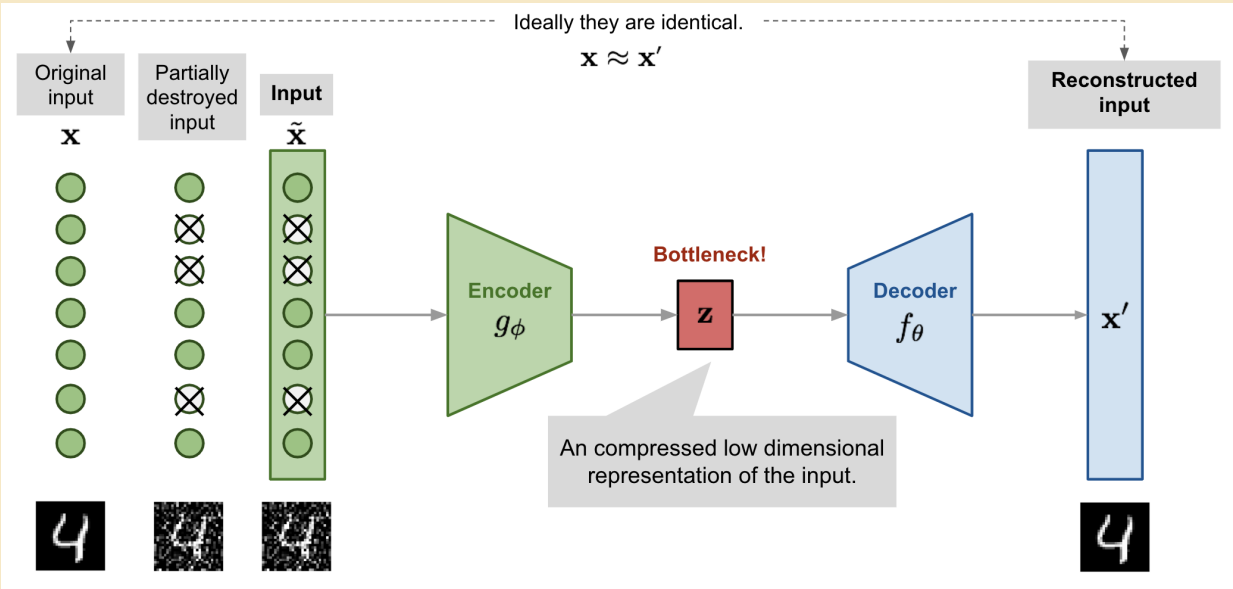


Figure 42: ***Denoising AutoEncoder architecture:*** *is a type of autoencoder used in machine learning, particularly for the task of denoising data. It works by intentionally introducing noise to its input data and then training itself to reconstruct the original, undistorted data. This process helps the DAE to learn the underlying structure of the data and improve its ability to ignore noise. In essence, a DAE's goal is to learn an identity function under the presence of noise, making them powerful tools for denoising and feature extraction tasks. Image extracted from [4]*

4.3.1 The DAE DenseNet Architecture

Conventional DAEs generally use linear stacked convolutional layers with down-sampling for the encoder, then symmetric linear stacked convolutional layers with up-scaling or deconvolution for the

decoder part. Although this approach has been proven to have great performances and being relatively simple and less complex, it usually comes with a trade-off in the network size. In fact, with linear stacked convolutional layers, the number of parameters of the network increases significantly. On the other hand, DenseNet architecture overcomes this problem by connecting all previous layers densely together. Similarly to what ResNet [48] architecture does with skip connections, in a DenseNet architecture, each layer receives inputs from all the preceding ones and passes its own output to all subsequent layers. The consequence is that the final output layer has direct information from every signal layer since the input. This approach allows us to shrink the network by reducing the number of layers and reusing all layers throughout the network. However, one should note that despite having a smaller network, its complexity significantly increases with dense connected layers.

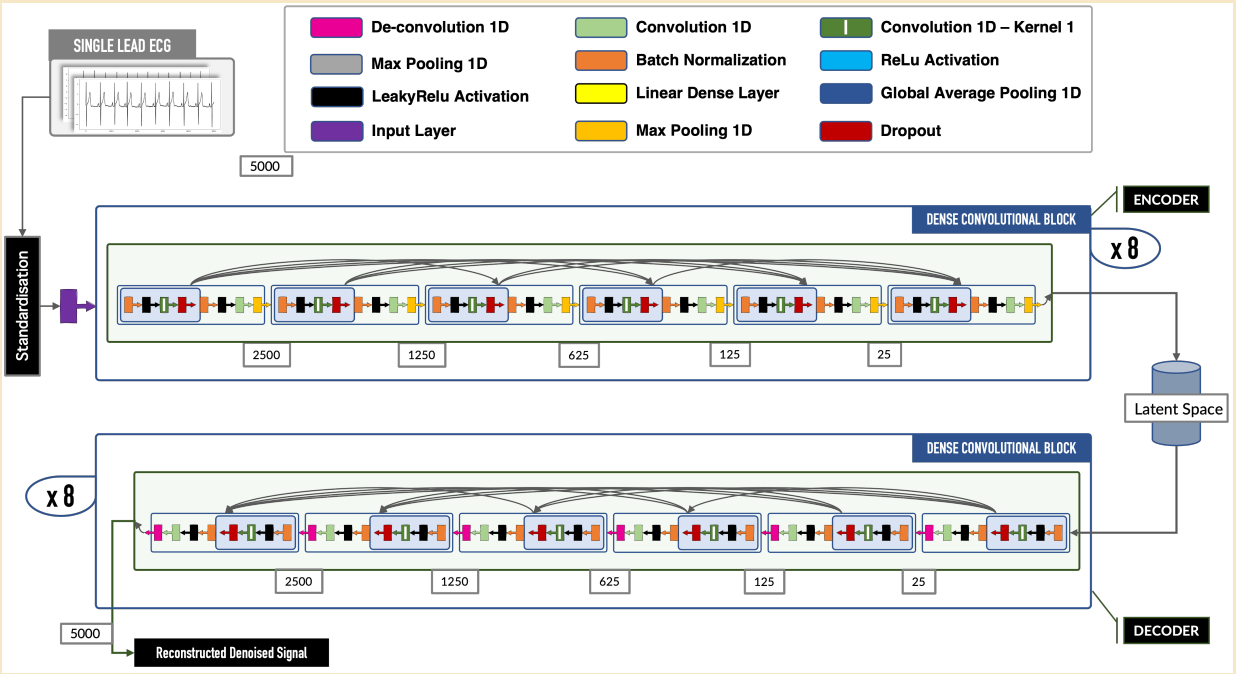


Figure 43: ***DeepFADE neural network architecture***

In this study, we designed a DenseNet Denoising Autoencoder (43) which compresses the ECG and reconstructs it while cancelling the noise. We primarily used the ELU activation function instead of conventional ReLU to avoid the "dying neurons" effect. The latter occurs when the ReLU neuron encounters negative values and the function gradient becomes zero, so, gradient descent algorithm will no longer update its weights and therefore, the neuron will always output zero. Whereas ELU has a small positive gradient *(not flat slope at 0 like ReLU)*, therefore for negative values the function will return $-\alpha x$ instead of zero, where x is the negative input. The negative scale factor is a hyper-parameter and is determined before training. The structure of the encoder part begins with a convolutional layer followed by a batch-normalization and ELU activation [49]. During this first step, higher features representations are extracted. The next step consists of eight successive densely connected convolutional blocks *(DenseBlocks)*. A DenseBlock is made of several convolutional sub-blocks called *DenseLayers*. Each DenseLayer begins with a bottleneck block of four layers *(a batch-normalization, followed by a ELU activation, a 1x1 convolution and a dropout layer)*. This bottleneck reduces the filter space dimensionality, decreasing the number of feature maps whilst retaining their salient features. The bottleneck technique helps reduce the number of parameters and increases computational efficiency. After the bottleneck comes another batch-normalization layer, a ELU activation, a classic convolution and another dropout layer. The dropout rate is a hyper-parameter and is determined before training. DenseLayers are densely connected, meaning that each DenseLayer takes as input all previous DenseLayers outputs. Between each DenseBlock lies a transition block with a bottleneck, followed by a ELU activation and an average pooling 1D layer which takes the average of

points, reducing the dimensionality of the previous output. The transition block aims to down-sample the data flow through the network and interconnect the DenseBlocks. The signal is progressively down-sampled from 5000 points to 25 points.

The decoder has a symmetric structure to the encoder where down-sampling functions are replaced with deconvolution layers and the signal is reconstructed to its original dimension as of 5000 points without the noise. The output layer is a convolution layer without batch-normalization nor activation function. The total number of trainable parameters of the model was 10,441,440. We used the Adam algorithm for the network optimizer [145].

4.3.2 Training Process

To train the model we used clean ECGs of 10s sampled at 500Hz with 5000 timepoints and applied to them several perturbations, We applied combinations of 6 noise levels (-6dB, 0dB, 6dB, 12dB, 18dB, 24dB), with and without baseline drift, real life artefacts (muscle artefact, powerline noise and electrode movement) extracted from the Physionet MIT-BH NSTDB dataset. We also set to 0 random window position on random signals, with a max window width of 500 timepoints.

Despite having ECGs with not much high frequencies noise, many of them have low frequencies noise thus baseline wander. To ensure the model is trained on clean signals, we applied on the ECGs a median filter (window size=200) to estimate its baseline, which was later substracted to the initial signal. This process is called signal detrending. We hypothesized that both detrending and lead differences would impact DAE's performance. To test these hypotheses, we designed four different versions of the model: *DeepFADE Lead II*: trained only on lead II without detrending (dpf_d2_dtr-); *DeepFADE Lead II Detrended*: trained only on lead II with detrending (dpf_d2_dtr+); *DeepFADE All Leads*: trained on all leads without detrending (dpf_all_dtr-); *DeepFADE All Leads Detrended*: trained on all leads with detrending (dpf_all_dtr+). Figure 44 illustrates an example of detrended ECG with its computed baseline.

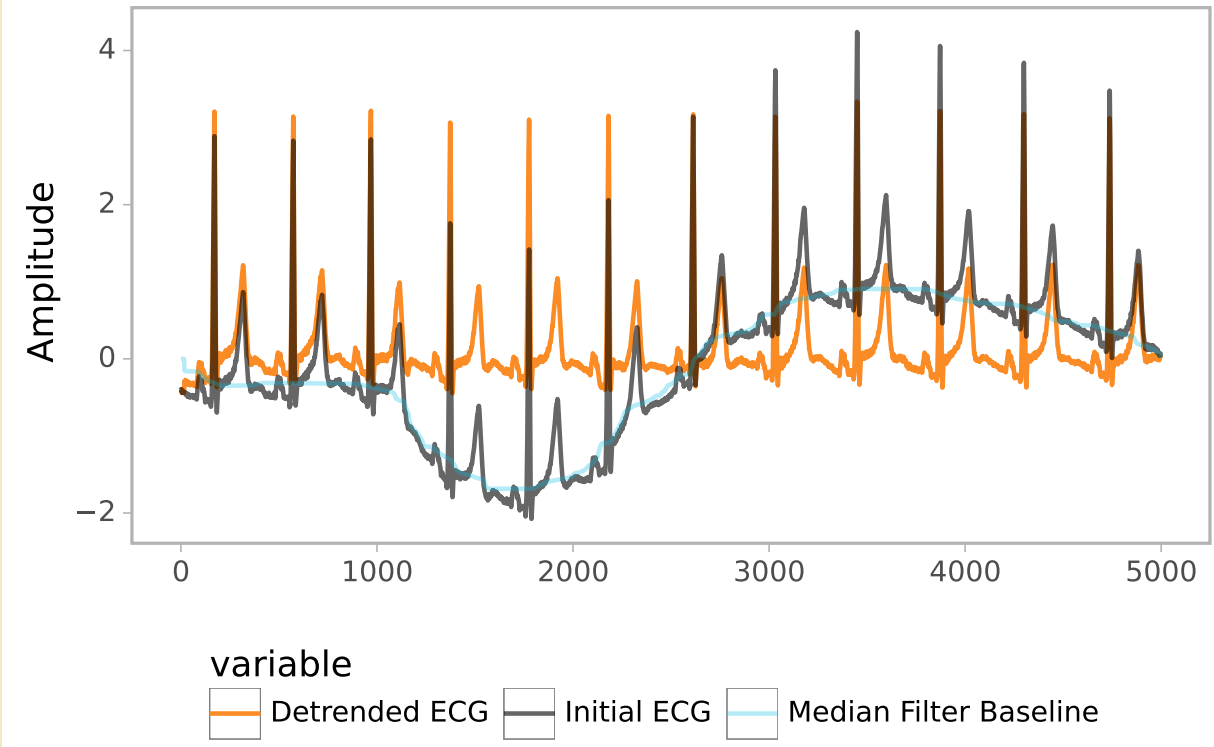


Figure 44: ***Detrending ECG Signal****: The gray line depicts the original ECG. The turquoise line is the baseline as identified with the median filter. The orange line depicts the detrended signal.*

4.3.3 Loss Functions & Metrics

To reduce waveform and amplitude alteration during the reconstruction of the signal in the decoder part, we trained the model to, alongside signal denoising, compute the baseline of the output signal. Therefore, each model have two outputs, one for the reconstructured signal and one for the predicted baseline, both outputs are then optimized. Learning to predict the baseline forces the models to learn the bottom shape of the signal which is important to avoid relative amplitude alteration, for example amplitude of the T wave relatively to the R peak. The error of the reconstruction of the signal is computed using the Soft-DTW loss function [146]. The Soft-DTW loss function is a variant of the Dynamic Time Warping (DTW) algorithm [147], which is a method for measuring the similarity between two sequences. The standard DTW algorithm is a measure of the dissimilarity between two sequences, by computing the minimum cumulative distance between them. Soft-DTW is a smooth approximation of the DTW distance, which is designed to be differentiable, and thus can be used in gradient-based optimization methods. This allows it to be used in training machine learning models. State-of-the-art DNN methods commonly use the Mean Squared Error (MSE). However, in time series data, the MSE loss may not be the best choice because it doesn't take into account the temporal structure of the data. The Soft-DTW loss function, on the other hand, is specifically designed for time series data and can capture the temporal dependencies between data points. One of the main advantages of using the Soft-DTW loss for DAE is that it allows the model to learn a more robust and accurate representation of the original time series data. The Soft-DTW loss is able to penalize large temporal shifts, which can be caused by noise, and thus the model will learn to ignore these shifts. Additionally, the Soft-DTW loss can also handle missing data, which allows the models to reconstruct missing beats of the signal. To optimize the baseline output of the models we used the MSE loss. The final loss is defined as:

$$Loss = (\lambda * SoftDTW(s, \hat{s})) + MSE(b, \hat{b}) \tag{4.1}$$

Where s and $\hat{s}$ are respectively the initial signal without noise addition and the predicted denoised signal,b and $\hat{b}$ are respectively the computed baseline *(with the median filter)* of the initial signal without noise addition and the predicted baseline and λ the hyper-parameter coefficient of the fist loss. To evaluate the quality of the reconstructed signal and to compare with state-of-the-art methods, we used the Signal to Noise Ratio (SNR). It is the ratio of the level of the desired signal to the level of background noise, usually measured in decibels (dB). A higher SNR indicates a better quality signal with less noise, while a lower SNR indicates a poorer quality signal with more noise. The SNR equation is defined as:

$$SNR = 10 * \log_{10}\left[\frac{\sum_{i=1}^{N} \hat{x}_i^2}{\sum_{i=1}^{N} (\hat{x}_i - x_i)^2}\right] \tag{4.2}$$

We also used a new metric of ours the Noise Reduction Robustness NRR. It measures the difference between a reference ECG signal, obtained by denoising a non-perturbed signal using the models, and the denoising of all ECG perturbations combinations of the same initial signal. This metric shows how close are the model's prediction to the reference signal, a low difference across different noise levels and perturbations indicates the model succeeds in denoising different type of signal alterations. The NRR is defined as follow:

$$NRR = |(M(signal) - M(P(signal, n, a, b))| \tag{4.3}$$

Where: M is the DeepFADE model, P is the perturbation function, n is the Gaussian noise level id dB to be applied, a is the real life artefact to be applied if provided and b conditions whether or not to apply random baseline drift to the signal.

4.4 Experimental Framework And Results

Models were trained three Physionet public ECG databases (ECGDMMLD, ECGRDVQ and MIT-BH NSTDB) and one private database. All databases have relatively clean ECGs except one MIT-BH

Experimentation			
	Training	Validation	Evaluation
Samples	(70%) 7562520	(10%) 598312	(10%) 599096
Holdout			
Samples	(10%) 596744		
Samples per Dataset			
Dataset	Raw	Altered	
Generepol	120952	5805696	
Physionet ECGDMMLD	31000	1488000	
Physionet ECGRDVQ	38016	1824768	
Physionet NSTDB	720	47520	

Table 10: Datasets partitions and number of samples per dataset with and without noise addition.

NSTDB. We performed a hyper-optimization to find optimal hyper-parameters, no cross-validation was conducted as the hyper-optimization process covered it. The main neural network architecture was implemented with Pytorch 1.13.1+cu117 with Python 3.10.6. Computation were performed on a HPC cluster consisting of : 6 GPU Nvidia A100 with 80 GB of memory each, a total of 224 CPU cores of 2.20 GHz and a total of 1TB of RAM. We implemented a parallel and distributed model training process across the cluster in Cython.

4.4.1 Datasets

We used Physionet databases [6] ECGDMMLD [148], ECGRDVQ [149] and MIT-BH NSTDB [150] and a private ECG dataset (Generepol [151]) for training, validating, evaluating and testing all 4 models. We used respectively 10s 3875x8 leads and 4752x8 leads ECGs of The ECGDMMLD and ECGRDVQ. The MIT-BH NSTDB database is an extraction of the MIT-H database. It contains 2x2 leads ECGs of 12 half-hour ECG recordings and 3 half-hour recordings of noise typical in ambulatory ECG recordings. The three noise records were assembled from the recordings by selecting intervals that contained predominantly baseline wander (in record 'bw'), muscle (EMG) artifact (in record 'ma'), and electrode motion artifact (in record 'em'). Electrode motion artifact is generally considered the most troublesome, since it can mimic the appearance of ectopic beats and cannot be removed easily by simple filters, as can noise of other types. We re-sampled this dataset from 360Hz to 500Hz and divided them in small ECGs of 10s (5000 timepoints). We also added these real life artefacts to the three other datasets. The Generepol dataset contains 15119x8 leads ECGs of 10s at 500Hz. In total we have 190688 raw signals with all leads combined. We applied the following perturbations and obtained 9356672 ECG signals on all leads:

- Gaussian noise level: -6dB, 0dB, 6dB, 12dB, 18dB, 24dB
- Artificial baseline wander
- Real life artifacts: baseline wander, muscle artifact, electrode motion artifact
- Random occlusion by replacing parts of the signal with zeroes

Datasets were split by patients into smaller partitions for the training and validation processes, Table 12 details datasets partitions and number of samples per dataset.

Params	Description	Value
Dense Layers	Number of convolution blocks for each layer	8
Dense Blocks	Vector of avg-pooling and deconvolution steps corresponding to dense blocks.	[2,2,2,5]
Dropout Rate	Fixed dropout rate	0.2
Activation	Activation function used inside the network	ELU(0.1)
Kernel	Fixed kernel size for all convolutions	3
Compression	Compression ratio used in transition blocks	1.0)
Learning Rate	Adam optimizer initial learning rate	0.001

Table 11: Models Final Hyper-Parameters

4.4.2 Training & Hyper-Optimization

We performed a hyper-optimization process to find optimal combination of parameters, all 4 models have the same parameters. Table 11 relates final hyper-parameters used. All computations with hyper-parameters took around 30 days to complete. We used the grid search algorithm to test different combinations. The validation partition dataset was used while training to assess progression of models performance. We set the maximum number of epochs to 400. We used adaptive learning rate for the optimizer with a reduction factor of 0.5 and a patience of 10 epochs while monitoring validation loss. The evaluation partition dataset was used to evaluate best models candidates to elect the best one. Each model was trained on 2 GPUs with a batch size of 128 split equally on each GPU. Adam optimizer was used for all models.

4.5 Results and Discussion

We computed three SNR measure inspired from [144], SNR_{in} between the raw signal and the perturbed one, SNR_{out} between the predicted signal and the raw signal and SNR_{imp} the difference between them. $SNR_{imp} = SNR_{out} - SNR_{in}$. The models topped a SNR_{imp} of 16dB which is higher than most of state-of-the-art methods. The models achieved a signal reconstructuion Soft-DTW loss of 0.1 and 0.001 for the MSE loss on the baseline prediction. Figure 47.B shows the denoising output of a signal at different noise levels with dae model trained on lead dII without data prepossessing, where as Figure 46.A shows the model trained with signal detrending prepossessing. The same ECG is used in both figures. We also tested the multi-leads model trained without data prepossessing on noise level 12dB in Figure 45. We compared three state-of-the-art (SOTA) methods with our models except for one trained on all leads without data prepossessing as it performed poorly compared to the 3 other models. In Figure 48.A we compare the SNR_{imp} distribution across partitions for all methods. Our three models display higher SNR values on all partitions and for all noise levels, the model trained on lead dII without signal detrending prepossessing presents better signal reconstruction quality. This is confirmed in Figure 48.B where we compare the NRR_{mean} distribution with the same configuration. The NRR average remains relatively constant and small. This can be interpreted as the model (DeepFADE dII Undetrended) predicting denoised signal across different level of noise with very small variations, thus confirming the robustness of the model. The use of a DenseNet architecture trained with the Soft-DTW loss and by also learning the baseline of the signal significantly improved the approach of denoising ECG signals using deep learning neural networks. Low performances of the model trained on all leads can be justified by the high heterogeneity of the different leads waveform. Indeed each of them have a specific waveform. A perspective to improve the multi-leads model could be to use a deeper network which will be able to capture the leads wave-

form heterogeneity. Another perspective of improvement could be to combine the model architecture with Recurrent Neural Network (RNN). In fact, beyond the spatial representation, RNN allow the learning of the temporal aspect. These architectures, through a cyclic structure allow the output of certain nodes to affect the subsequent input of the same nodes, thus offering a dynamic temporal behavior. This architecture could capture the spatial representation of the signal as well as its temporal

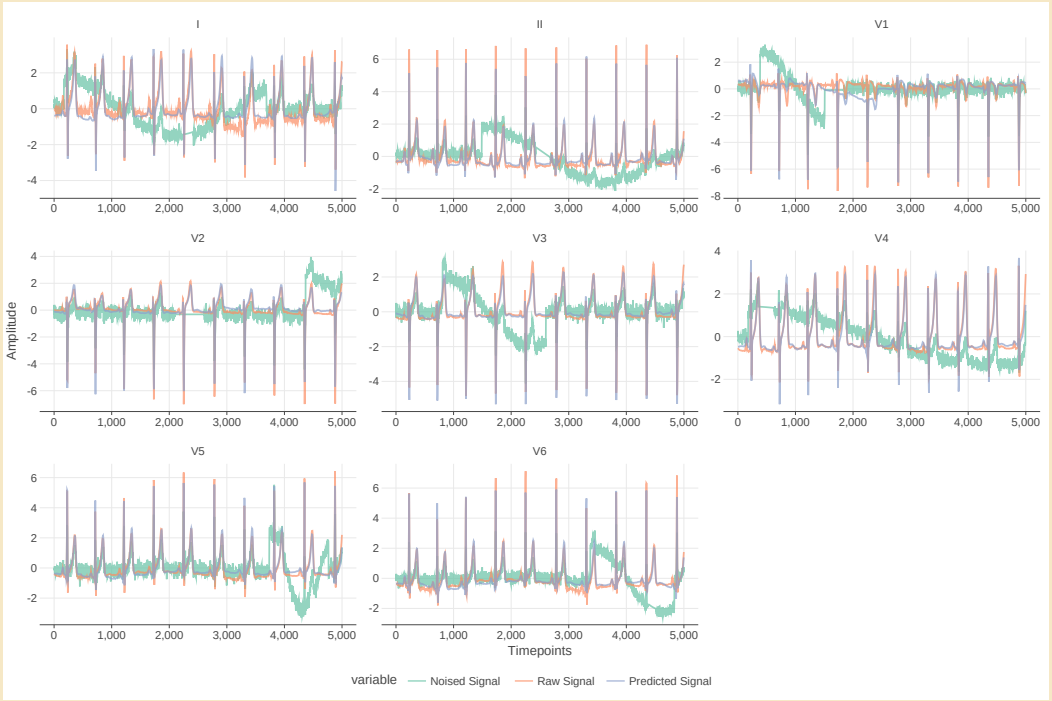


Figure 45: ***DeepFADE All Leads*** *- Prediction on Noise Level 12dB*

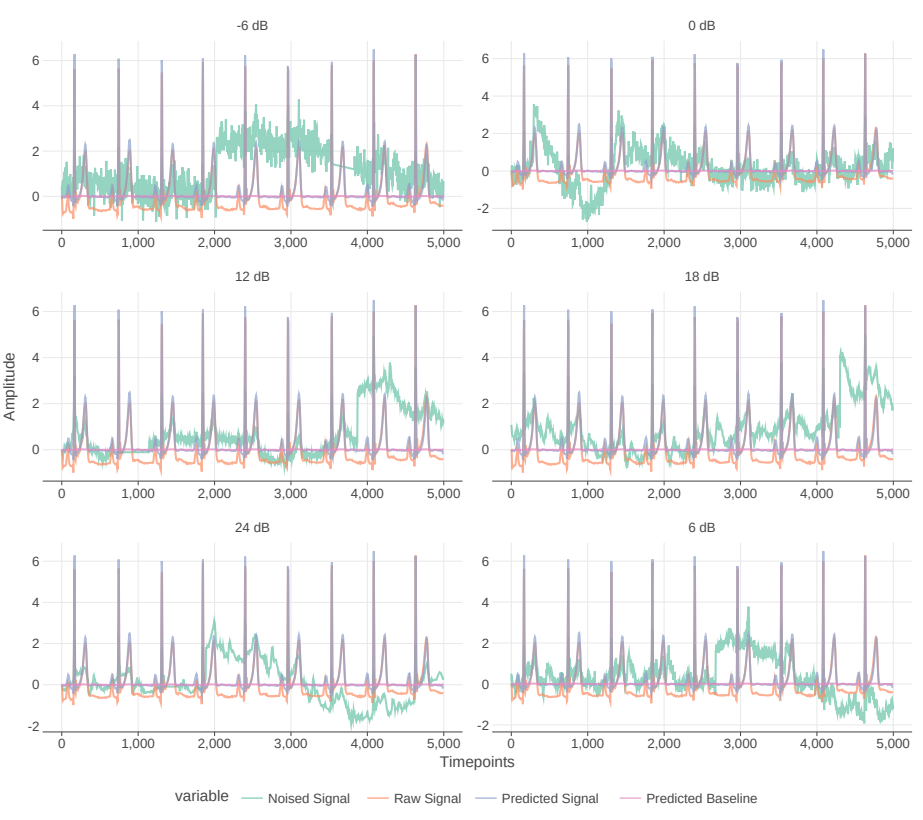


Figure 46: ***Denoising multiple levels of noise*** *with DeepFADE Lead II Detrended*

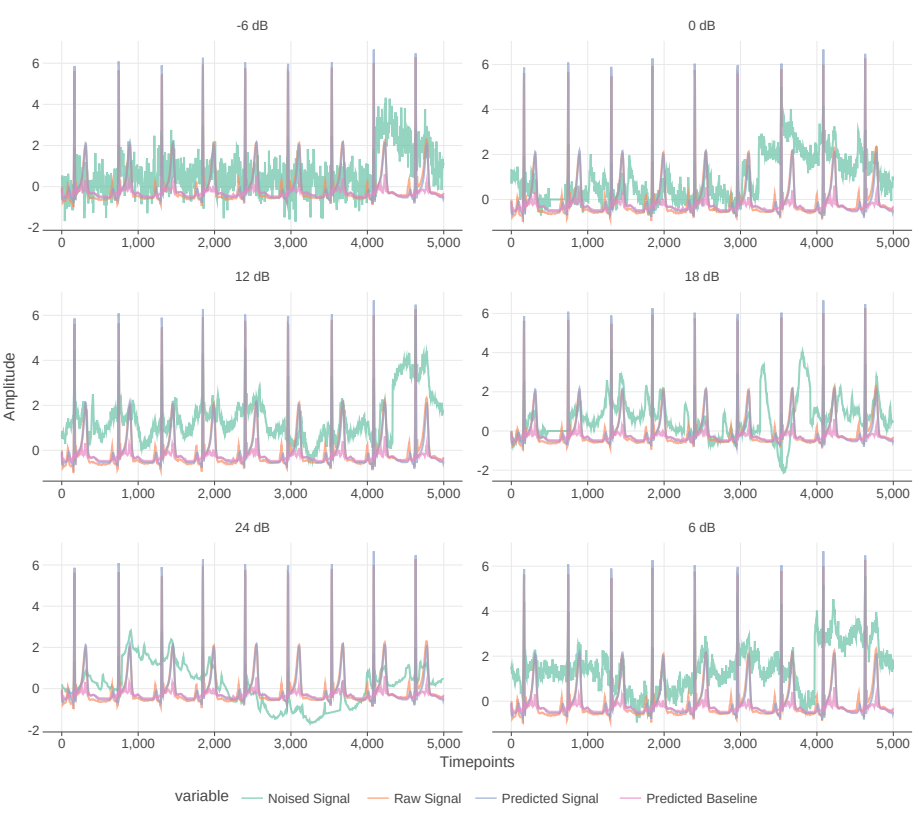


Figure 47: ***Denoising multiple levels of noise*** *with DeepFADE Lead II UNDetrended*

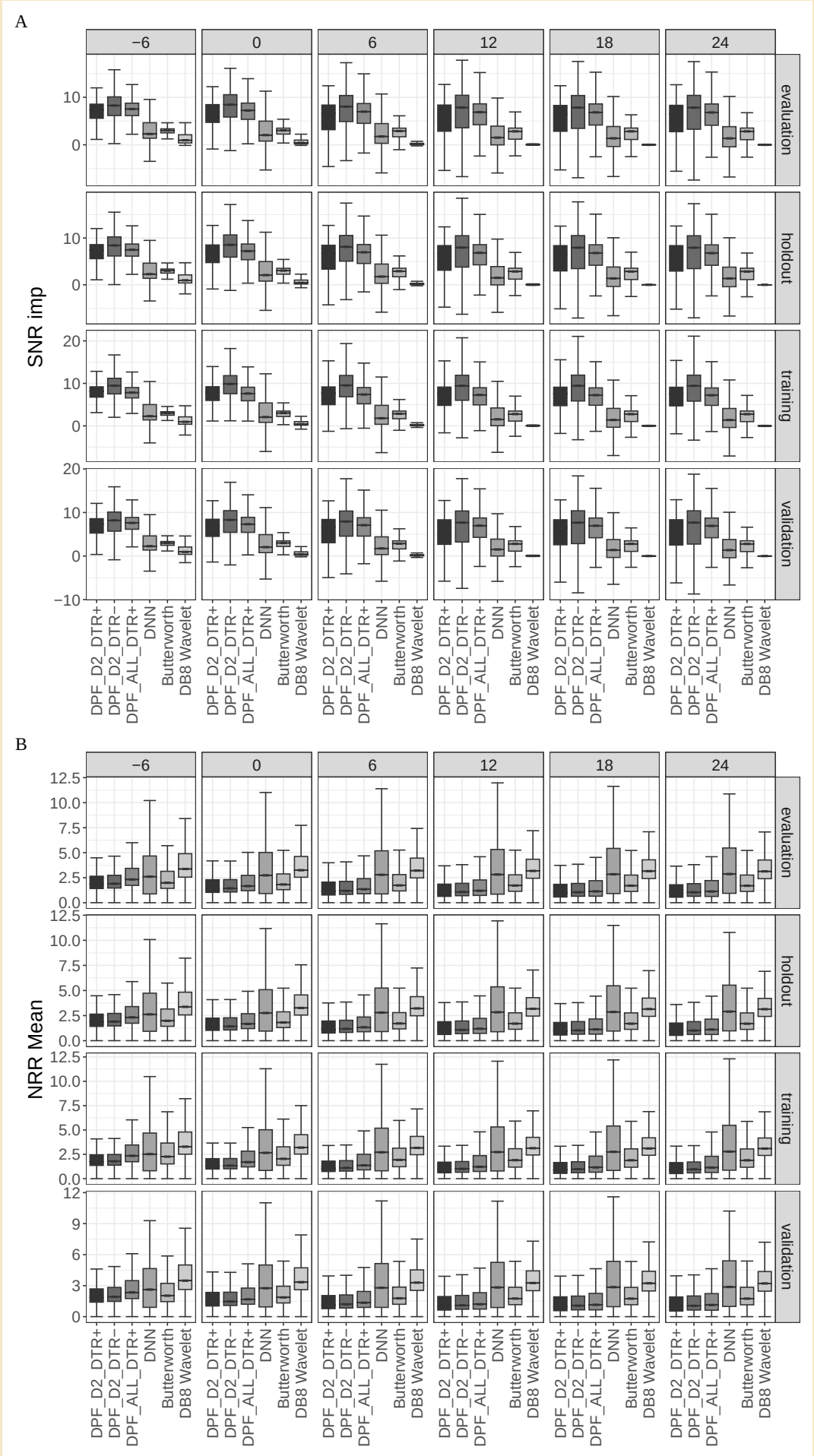


Figure 48: ***SNR_imp and mean NRR Distribution by Noise Level and Partition.***

4.6 Conclusion

In this study, we introduce an improved application of deep learning to the field of electrocardiogram signal processing: the Deep Frequency Amplitude Denoising Encoder (DeepFADE). This denoising autoencoder demonstrates considerable promise for the enhancement of ECG analysis by effectively eliminating noise and artifacts that can interefere with accurate cardiac diagnosis and monitoring and ECG waveform analysis.

ECG signals, which are complex and sensitive, are susceptible to various sources of noise and interference. This situation presents considerable challenges to both manual interpretation and automated ECG processing techniques. There have recent advancements in denoising methods. Traditional methods such as digital and wavelet filters, though reasonably effective, aren't particularly tailored for ECG waveforms, leading to sub-optimal results, especially with highly deteriorated signals. Even though neural networks have shown improvements, they can remove or attenuate crucial features of the ECG signal.

To overcome these challenges we designed our denoising autoencoder, DeepFADE, to accurately remove noise from the signal while preserving relevant features in the signal such as aspects of the QRS complex and P and T waves which are critical for cardiac disease diagnosis and monitoring. Our method outperformed the current state-of-the-art denoising methods in overall performance. The model was trained on artificially noised ECGs as well as real life noisy ECGs, to remove the noise and reconstruct parts of the signal that might have been deteriorated. However, our work also underlines an intrinsic challenge posed by this approach: assessing the relevance of the noise being removed. Although these variations are often classified as noise, they may carry significant information pertinent to the detection or classification of pathologies. This highlights the delicate balance between noise reduction and preserving potentially crucial signal features.

Despite this limitation, DeepFADE successfully removes and reconstruct the signal which improves further ECG analysis such as automatic segmentation, pathology detection and interpretability thereby enhancing the accuracy of diagnostic procedures and subsequent treatment decisions. To compare our methods with SOTAs we used the signal-to-noise ratio (SNR) which indicates the level of noise in a signal. We computed 3 values of SNR, SNR_{in} which indicates the level of noise between the raw signal and the noisy one, SNR_{out} which indicates the level of noise between the reconstructed signal and the raw initial signal and SNR_{imp} the difference between them. $SNR_{imp} = SNR_{out} - SNR_{in}$. In our dataset, most clean ECGs had an initial estimated SNR of 23dB. After denoising the signal, we obtained an SNR_{out} of 22dB which indicates the denoised signal is almost as clean as the original signal without noise addition. When we look at SNR_{imp}, we reached a noise level 16dB which is higher than SOTAs methods (around 15dB).

4.7 The Negative Impact of Denoising on Automated Classification of Electrocardiograms

To further assess how cleaning ECGs can affect classification models, we evaluated most State-of-the-art denoising methods and DeepFADE as well. Results show that the cleaning process effectively alters some of the ECG features thus resulting in a drop of performances of classification models. Results were published in the Deep Generative Models for Health Workshop NeurIPS 2023.

The Negative Impact of Denoising on Automated Classification of Electrocardiograms

Federica Granese*
UMMISCO
IRD, Sorbonne Université
Bondy, France
federica.granese@ird.fr

Ahmad Fall*
UMMISCO
IRD, Sorbonne Université
Bondy, France
Université Cheikh Anta Diop
Dakar, Sénégal
ahmad.fall@ird.fr

Alex Lence
UMMISCO
IRD, Sorbonne Université
Bondy, France
alex.lence@ird.fr

Joe-Elie Salem
Vanderbilt University Medical Center, Nashville, TN, USA
Clinical Investigation Center Paris-Est, INSERM, UNICO-GRECO Cardio-Oncology Program
Pitié-Salpêtrière University Hospital,Sorbonne Université, Paris, France
joe-elie.salem@aphp.fr

Jean-Daniel Zucker
UMMISCO
IRD, Sorbonne Université, Bondy, France
INSERIM, NutriOmique, AP-HP
Hôpital Pitié-Salpêtrière
Paris, France
jean-daniel.zucker@ird.fr

Edi Prifti
UMMISCO
IRD, Sorbonne Université, Bondy, France
INSERIM, NutriOmique, AP-HP
Hôpital Pitié-Salpêtrière
Paris, France
edi.prifti@ird.fr

Abstract

We present an evaluation of recent state-of-the-art electrocardiogram denoising methods and assess their impact on the performance of convolutional deep learning-based classifiers, with a focus on the risk prediction of Torsade-de-Pointes arrhythmia. Our findings indicate that the traditional approach of evaluating denoising methods independently of the application is insufficient. This is particularly the case for applications where the signals are used for phenotype prediction. We observed that when classifiers are fed denoised data instead of raw data, their performance significantly deteriorates, with a decline of up to 40 percentage points in accuracy and up to 27 percentage points in AUROC when a misclassification detection method is further applied, underscoring a notable reduction in model reliability. These findings highlight the importance of considering the downstream impact of denoising on automated classification tasks and shed light on the complexities of trustworthiness in the context of healthcare applications.

1 Introduction

An electrocardiogram (ECG) is a century-old test used to assess cardiac health. It involves placing two or more electrodes at specific locations on the chest, arms, and legs and recording the

*Equal contribution

1st Workshop on Deep Generative Models for Health at NeurIPS 2023.

heart's electrical signals through a central unit [1]. The ECG waveform (as in Figure 1) bears important information and subtle variations can be indicative of a large spectrum of diseases.

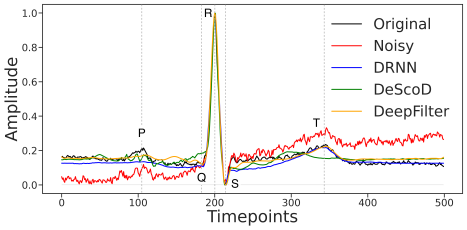


Figure 1: Example of a (heartbeat) recorded by an ECG along with waves annotation and denoised outputs.

However, recording such signals can be often susceptible to noise from various sources, including electrode loose contact, patient movement, and muscle contractions [2, 3]. This can alter the waveform resulting in challenging analyses and subsequent diagnoses. Broadly speaking, when we refer to *denoising of a signal*, we are alluding to the process of recovering the raw signal from the noisy one. Denoising ECG signals have been extensively studied in the literature and numerous methods have been proposed. These methods stemmed from different fields, including signal processing [4] as well as more recently deep learning [2, 5, 6, 7]. The evaluation of the proposed methods typically treats denoising as an independent task, while comparing the dissimilarity between the 'clean' signal and its noisy counterpart.

In this paper, we argue that assessing the denoising performance independently of the final downstream tasks is insufficient and can have important unforeseen consequences. Indeed, signals that may appear visually similar to the human eye can be fundamentally distinct when viewed from a neural network's perspective [8]. Specifically, in this work, we connect denoising with the prediction of the risk of a life-threatening type of arrhythmia known as Torsades-de-pointes (TdP) for which Prifti et al. [9] previously developed a high-performing deep convolutional neural network model. This model was trained on ECGs recorded from healthy individuals before and after the intake of Sotalol, a drug known to increase the risk of TdP.

Starting from the intuition that an effective denoiser should retain as much valuable information as possible while reducing noise in the data, we analyzed five recent state-of-the-art (SOTA) denoisers on a real-life ECG dataset `Generepol` [10]. We show that while providing good results in the denoising task, the application of these methods impact negatively the performance of the TdP risk classification model. Indeed, **the results reveal a significant decrease in the classifier's accuracy when the denoised ECG is classified as compared to the original ECG**. Additionally, **an examination of the distributions of correctly and incorrectly classified samples reveals a decrease in the model's confidence when classifying denoised data**, particularly concerning correctly classified samples. We performed the evaluation on a variety of denoisers, which include a *diffusion model* [6], a *deep recurrent neural network* (DRNN) [2], an *autoencoder* [7], and a *convolutional neural network* (CNN) [5]. Additionally, we considered a signal processing technique i.e., *wavelet transform* [4].

2 Problem formalization

The ECG denoising problem can be seen as a specific instance of the total-variation (TV) denoising problem [11] in the context of one-dimensional signals. In particular, we are given a *noisy signal* $\widetilde{\mathbf{x}} \in \mathbb{R}^d$, with $d > 1$, defined as $\widetilde{\mathbf{x}} \triangleq \mathbf{x} + \boldsymbol{\delta}$ where $\boldsymbol{\delta} \in \mathbb{R}^d$ is some unknown perturbation, and $\mathbf{x} \in \mathbb{R}^d$ is the *original signal*. The goal is to recover the underlying *denoised signal* $\widehat{\mathbf{x}} \in \mathbb{R}^d$ that best approximates the original signal $\mathbf{x}$. However, finding an accurate approximation of $\mathbf{x}$ from $\widetilde{\mathbf{x}}$ is generally ill-posed [12]. Therefore, the TV denoising problem is typically formulated as [13, 12]: $\min_{\widehat{\mathbf{x}} \in \mathbb{R}^d} F(\widehat{\mathbf{x}}, \widetilde{\mathbf{x}}) + \lambda R(\widehat{\mathbf{x}})$, where $F : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ represents the *data fidelity term*, $R : \mathbb{R}^d \to \mathbb{R}$ is the *regularization term* that narrows down the space of candidate solutions, and $\lambda > 0$ is the *regularization parameter* controlling the trade-off between permissible noise and the regularity imposed by R (e.g., the smoothness of the underlying distribution).

A *denoiser* is expected to solve the minimization by learning $\widehat{\mathbf{X}} \sim q_\psi(\widehat{\mathbf{X}}|\widetilde{\mathbf{X}})$ where ψ are the parameters to learn. In the following paper, we will investigate how SOTA denoisers behave when denoised signals are given as input for a predefined classification task. Therefore, throughout the paper, we refer to the *target classifier* as $p_\theta(\widehat{Y}|\mathbf{X})$ where $\widehat{Y}$ is the random variable representing the classifier's inference and θ are the learned parameters. Its induced hard decision is defined as $f_\theta : \mathcal{X} \to \mathcal{Y}$ s.t. $f_\theta(\mathbf{x}) \triangleq \arg\max_{y \in \mathcal{Y}} p_\theta(y|\mathbf{x})$, where $\mathcal{X} \subset \mathbb{R}^d$ is the feature space corresponding

Table 1: The 'Noised signal' column presents the results for the signal following perturbation and prior to any denoising. The results are expressed in terms of `mean±std` across all heartbeats.

	Noised signal	DeScoD [6]	DRNN [2]	DeepFilter [5]
SSD	7.62±9.45	2.35±4.39	4.00±4.25	**1.45**±2.95
MAD	0.23±0.12	0.13±0.07	0.23±0.08	**0.12**±0.05
PRDN	84.41±52.82	46.89±26.14	66.69±20.05	**37.09**±18.25
SNR	7.98±6.45	12.31±4.06	8.56±3.14	**14.10**±3.41
CosS	0.93±0.06	0.96±0.04	0.93±0.04	**0.98**±0.02

to the set of original ECGs that have not been subjected to any perturbations and $\mathcal{Y} = \{1, \dots, C\}$ represent the concept of the label space related to some task of interest, such as the detection of the risk for developing some form of arrhythmia. Formally, given $\mathbf{x} \in \mathcal{X}$ we will be interested in studying whether $f_\theta(\mathbf{x}) = f_\theta(\widehat{\mathbf{x}}) \equiv f_\theta(g_\psi(\widetilde{\mathbf{x}}))$ where $g_\psi(\widetilde{\mathbf{x}})$ is the sampling of $q_\psi(\widehat{\mathbf{x}}|\widetilde{\mathbf{x}})$.

3 Evaluation framework and results

3.1 Evaluation metrics for the standalone denoising task

We consider the distortion metrics used in the most recent literature [5, 6] to assess the capabilities of the SOTA methods for signal denoising. These metrics come from the ECG compression field [14, 15] to measure the reconstruction error between the original signal and the one obtained after compression and then decompression (encoding, decoding).

We employ the **sum of square distances** (the lower the better)

$$\text{SSD}(\mathbf{x}, \widehat{\mathbf{x}}) \triangleq \sum_{i=1}^{d} \left(\mathbf{x}(i) - \widehat{\mathbf{x}}(i)\right)^2; \tag{1}$$

the **absolute maximum distance** (the lower the better)

$$\text{MAD}(\mathbf{x}, \widehat{\mathbf{x}}) \triangleq \max_{i \in \{0,\dots,d\}} |\, \mathbf{x}(i) - \widehat{\mathbf{x}}(i) \,|; \tag{2}$$

a normalized version of the **percentage root-mean-square difference** (the lower the better)

$$\text{PRDN}(\mathbf{x}, \widehat{\mathbf{x}}) \triangleq \sqrt{\frac{\text{SSD}(\mathbf{x}, \widehat{\mathbf{x}})}{\sum_{i=1}^{d} \left(\mathbf{x}(i) - \mu\right)^2}} \cdot 100, \tag{3}$$

where μ is the mean of the clean signal, i.e., $\mu = \frac{1}{d}\sum_{i=1}^{d} \mathbf{x}(i)$. Notice that the **signal-to-noise ratio** can be easily calculated from the percentage root-mean-square difference (PRD) without normalization [15] – that corresponds to eq. (3) with $\mu = 0$ – as follows

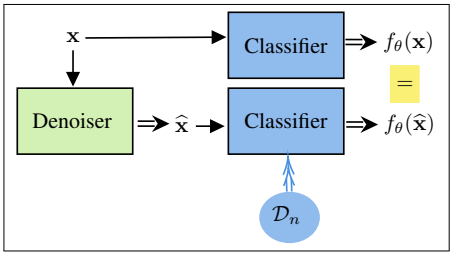
$$\text{SNR}(\mathbf{x}, \widehat{\mathbf{x}}) \triangleq 10 \cdot \log_{10}\left(\frac{\sum_{i=1}^{d} \left(\mathbf{x}(i)\right)^2}{\text{SSD}(\mathbf{x}, \widehat{\mathbf{x}})}\right) = 40 - 20 \cdot \log_{10}(\text{PRD} * 0.01). \tag{4}$$

Finally, we check the similarity between the two signals with **cosine similarity** (closer to 1 the better)

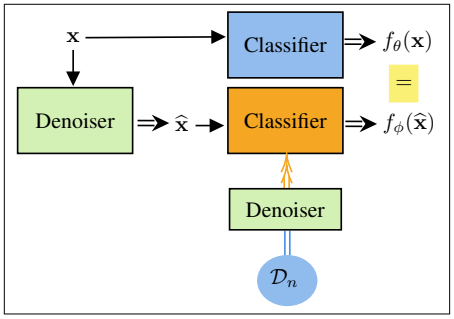
$$\text{CosS}(\mathbf{x}, \widehat{\mathbf{x}}) \triangleq \frac{\mathbf{x} \cdot \widehat{\mathbf{x}}}{||\mathbf{x}|| \cdot ||\widehat{\mathbf{x}}||} = \frac{\sum_{i=1}^{d} |\, \mathbf{x}(i) \cdot \widehat{\mathbf{x}}(i) \,|}{\sum_{i=1}^{d} |\mathbf{x}(i)| \cdot \sum_{i=1}^{d} |\widehat{\mathbf{x}}(i)|}. \tag{5}$$

3.2 Considered dataset, TdP risk prediction, and review of the related methods

`Generepol` dataset [10] and Torsades-de-Pointes (TdP) risk prediction [9]. We use the `Generepol` [10] dataset consisting of 10-seconds 8-lead ECGs sampled at 500Hz. For the experiments, we focus on lead II, commonly used to record the rhythm strip [16]. Since all the SOTA are tuned to work on *heartbeats*, i.e., on one single pulsation of the heart at a time, we segmented the original signals into chunks of 1s (500-points), centered around the R peaks. Finally, the training



(a) 'Original setting': Evaluation of the denoiser and classifier pre-trained on original ECGs.



(b) 'Denoised setting': Evaluation of the denoiser and classifier trained on denoised ECGs.

Figure 2: In Figure 2a, the classifier f_θ is trained on the original training set $\mathcal{D}_n = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ for TdP risk prediction. On the other hand, in fig. 2b, the classifier f_ϕ, where ϕ are the learned parameters, is trained with the original training dataset but denoised using the method under consideration. In both cases, $\mathbf{x}$ and the obtained denoised version $\widehat{\mathbf{x}}$ represent a sample from the testing set.

set consisted of 30009 Sot+ and 32448 Sot- ECGs heartbeats, the validation set has 3659 Sot+ and 4188 Sot- ECG heartbeats, and the testing set comprised 7221 Sot+ and 7543 Sot- ECG heartbeats. Specifically, we use Sot- and Sot+ to refer to ECGs recorded in healthy individuals respectively before and after the intake of 80mg Sotalol, a drug known to strongly increase the risk of developing Torsade-de-Pointes events. We consider the CNN model originally developed by Prifti et al. [9] for TdP risk prediction and we retrained it to work on single heartbeats. This DenseNet model with six blocks (each having eight dense convolutional layers) was trained for 100 epochs using Adam optimizer, learning rate of 0.001, dropout rate of 0.2.

DeepFilter [5]. The model consists of six Multi-Kernel Linear And Non-Linear (MKLANL) filter modules. Each module contains two groups of four convolutional layers, where each layer is followed by a linear activation function or by a rectified linear unit (ReLU) depending on the group's type. The training loss is a combination of the sum of the squared distance and the maximum absolute distance between the clean ECG and the denoised one (cf. section 3.1). The idea is therefore to learn "smart" filters in order to discriminate between the desired ECG signal and the undesired noise.

DeScoD-ECG [6] is a novel approach that utilizes a conditional-score diffusion model. The generative model begins with Gaussian white noise and proceeds to iteratively reconstruct the signal through a fixed Markov Chain. Each step of the reconstruction involves Gaussian translation, which is conditioned on the previous step's reconstructions and the noisy ECG observations.

DRNN [2]. A DRNN was proposed here as a denoiser. Initially, the signal undergoes processing through a recurrent layer comprising Long Short-Term Memory (LSTM) [17] units. Then, the signal is further processed through a specified number of dense layers with ReLU activation. The final layer is linear and responsible for aggregating the outputs from the preceding layer by summation.

3.3 Results

Due to space constraints, the results on FCN+DAE [7] and Wavelet transform [4] are presented in Appendices A.1 and A.2.1. We refer to Appendix A.2.2 for the results conducted on an additional publicly available dataset (`PTB-XL` [18]). We relegate to Appendix A.1, a deeper exploration of TdP risk classification.

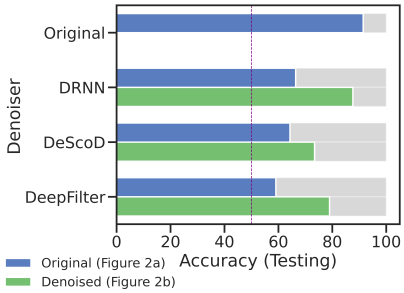


Figure 3: The vertical dashed line is at 50% accuracy.

Evaluation of the denoising task standalone. We conducted simulations of real-life noise (different levels of) *baseline wander* [4] inspired by original data obtained from the Physionet MIT-BIH NSTDB dataset [19, 20]. We used this data to assess the robustness of the evaluated methods when denoising the `Generepol` dataset. While the comprehensive discussion is relegated in Appendix A.2.1, we provide in Table 1 the summary of the results.

Effect of the denoising on TdP risk prediction. Given the TdP model trained on data without prior denoising, our objective was to assess the model's performance when the testing data is denoised using one of the SOTA denoisers. To achieve this, we begin with the original testing set from `Generepol`, which had not been subjected to any additional noise. Subsequently, we applied these denoisers to clean the signals before feeding them to the TdP risk classifier (see Figure 2a). **It is important to note that an effective denoiser should be capable of removing noise while preserving essential signal information and relevant features**. We provide in Figure 3 the accuracies of the model w.r.t. the different denoisers. The label 'Original' refers to the performance on the testing data, without any denoising applied to it. Interestingly, we observe a decrease in performance across all cases. This suggests that, despite the favorable results shown by the denoisers in Table 1, they are, in fact, removing valuable information from the signal, essential for the classification task.

We further explored the effect of denoising on the classification task by applying prior denoising to all partitions within `Generepol`. Consequently, we trained a new model for each denoiser (see Figure 2b). The updated accuracies are in Figure 3. Although the new models did not replicate the original performance, we observed a slight improvement compared to the previous setting. We finally analyzed the distribution of correctly and wrongly classified samples in Figure 4. Notably, neural networks are recognized for being overly *confident* in their predictions [21] (we call *confidence* the probability associated with the model's predicted class for a given sample). However, regardless of the denoisers used, the models tend to be less confident with the denoised data. **This can pose risks, as model confidence is commonly used for subsequent tasks related to model reliability**, such as *misclassification detection* [21, 22]. One technique used in computer vision for this task involves estimating the probability of classification error starting from the softmax outputted by the classifier [21]. Given an input sample, if this score exceeds a predefined threshold, the prediction is considered wrong, otherwise correct. The results, in Appendix A.1, show that alterations in the posterior distribution of the classifier have a detrimental effect on the method's ability to distinguish correctly and incorrectly classified samples with 40 percentage-point increase in False positive rate at 95% True positive rate (FPR, shortly) and 27-point reduction in AUROC.

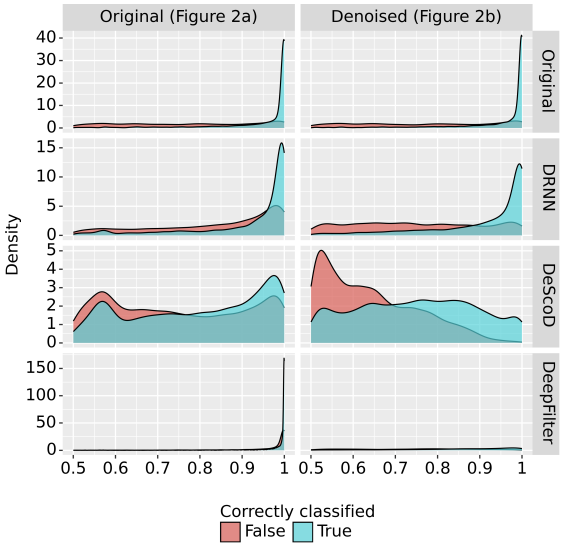


Figure 4: We split samples according to their true labels in blue and red. We examine the predicted class probability for each sample.

4 Conclusions

We examined ECG denoising methods and their effect on the automated classification of arrhythmia risk prediction. Our findings reveal that assessing denoising methods without considering downstream classification tasks yields overly optimistic results. We observed a reduction in classifier accuracy (up to 40 percentage points) when provided with denoised data compared to the original data. Further experiments have shown how alterations in the posterior distributions of the classifier on denoised data can have a detrimental impact on the misclassification detection task. These results stress the serious implications of denoising signals for the reliability of automated tasks, most notably within essential sectors such as healthcare, where even minor inaccuracies can have life-threatening outcomes.

Acknowledgments and Disclosure of Funding

This study was supported by the ANR-20-CE17-0022 DeepECG4U funding from the French National Research Agency.

References

[1] Electrocardiogram. https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/electrocardiogram.

[2] Karol Antczak. Deep recurrent neural networks for ecg signal denoising. *arXiv preprint arXiv:1807.11551*, 2018.

[3] Sarang L Joshi, Rambabu A Vatti, and Rupali V Tornekar. A survey on ecg signal denoising techniques. In *2013 International Conference on Communication Systems and Network Technologies*, pages 60–64. IEEE, 2013.

[4] Rahul Kher et al. Signal processing techniques for removing noise from ecg signals. *J. Biomed. Eng. Res*, 3(101):1–9, 2019.

[5] Francisco P Romero, David C Piñol, and Carlos R Vázquez-Seisdedos. Deepfilter: An ecg baseline wander removal filter using deep learning techniques. *Biomedical Signal Processing and Control*, 70:102992, 2021.

[6] Huayu Li, Gregory Ditzler, Janet Roveda, and Ao Li. Descod-ecg: Deep score-based diffusion model for ecg baseline wander and noise removal. *IEEE Journal of Biomedical and Health Informatics*, 2023.

[7] Hsin-Tien Chiang, Yi-Yen Hsieh, Szu-Wei Fu, Kuo-Hsuan Hung, Yu Tsao, and Shao-Yi Chien. Noise reduction in ecg signals using fully convolutional denoising autoencoders. *Ieee Access*, 7:60806–60813, 2019.

[8] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[9] Edi Prifti, Ahmad Fall, Giovanni Davogustto, Alfredo Pulini, Isabelle Denjoy, Christian Funck-Brentano, Yasmin Khan, Alexandre Durand-Salmon, Fabio Badilini, Quinn S Wells, et al. Deep learning analysis of electrocardiogram for risk prediction of drug-induced arrhythmias and diagnosis of long qt syndrome. *European Heart Journal*, 42(38):3948–3961, 2021.

[10] Joe-Elie Salem, Marine Germain, Jean-Sébastien Hulot, Pascal Voiriot, Bruno Lebourgeois, Jean Waldura, David-Alexandre Tregouet, Beny Charbit, and Christian Funck-Brentano. Genome wide analysis of sotalol-induced ikr inhibition during ventricular repolarization, "generepol study": Lack of common variants with large effect sizes. *PLOS ONE*, 12(8):1–16, 08 2017.

[11] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[12] Samuel Vaiter, Charles-Alban Deledalle, Gabriel Peyré, Charles Dossal, and Jalal Fadili. Local behavior of sparse analysis regularization: Applications to risk estimation. *Applied and Computational Harmonic Analysis*, 35(3):433–451, 2013.

[13] Mila Nikolova. Minimizers of cost-functions involving nonsmooth data-fidelity terms. application to the processing of outliers. *SIAM Journal on Numerical Analysis*, 40(3):965–994, 2002.

[14] M Sabarimalai Manikandan and Samarendra Dandapat. Ecg distortion measures and their effectiveness. In *2008 First International Conference on Emerging Trends in Engineering and Technology*, pages 705–710. IEEE, 2008.

[15] Andrea Němcová, Radovan Smíšek, Lucie Maršánová, Lukáš Smital, and Martin Vítek. A comparative analysis of methods for evaluation of ecg signal quality after compression. *BioMed research international*, 2018, 2018.

[16] Steve Meek and Francis Morris. Introduction. i—leads, rate, rhythm, and cardiac axis. *Bmj*, 324(7334):415–418, 2002.

[17] Alex Graves and Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pages 37–45, 2012.

[18] Patrick Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Dieter Kreiseler, Fatima I Lunze, Wojciech Samek, and Tobias Schaeffter. Ptb-xl, a large publicly available electrocardiography dataset. *Scientific data*, 7(1):154, 2020.

[19] George B Moody, W Muldrow, and Roger G Mark. A noise stress test for arrhythmia detectors. *Computers in cardiology*, 11(3):381–384, 1984.

[20] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

[21] Federica Granese, Marco Romanelli, Daniele Gorla, Catuscia Palamidessi, and Pablo Piantanida. Doctor: A simple method for detecting misclassification errors. *Advances in Neural Information Processing Systems*, 34:5669–5681, 2021.

[22] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

[23] Ingrid Daubechies. *Ten lectures on wavelets*. SIAM, 1992.

[24] SZ Mahamoodabadi, A Ahmedian, and MD Abolhasani. Ecg feature extraction using daubechies wavelet. In *Proceedings of the fifth IASTED International Conference VISUALIZATION, IMAGING and IMAGE PROCESSING, September*, pages 7–9, 2005.

[25] Yee Guan Yap and A John Camm. Drug induced qt prolongation and torsades de pointes. *Heart*, 89(11):1363–1372, 2003.

[26] Esseim Sharma, Brian McCauley, Wasiq Sheikh, Anshul Parulkar, and Antony Chu. Torsades de pointes secondary to sotalol: Predictable but not always preventable. *Journal of the American College of Cardiology*, 73(9S1):2789–2789, 2019.

[27] Eduardo Dadalto, Marco Romanelli, Georg Pichler, and Pablo Piantanida. A data-driven measure of relative uncertainty for misclassification detection. *arXiv preprint arXiv:2306.01710*, 2023.

[28] Fei Zhu, Zhen Cheng, Xu-Yao Zhang, and Cheng-Lin Liu. Openmix: Exploring outlier samples for misclassification detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12074–12083, 2023.

Table 2: The 'Noised signal' column presents the results for the signal prior to any denoising and following perturbation. The results are expressed in terms of `mean±std` across all heartbeats.

	Noised signal	FCN+DAE [7]	DeepFilter [5]	Wavelet [4]
SSD	7.62±9.45	27.78±16.89	**1.45**±2.95	7.23±9.39
MAD	0.23±0.12	0.54±0.08	**0.12**±0.05	0.21±0.12
PRDN	84.41±52.82	184.48±65.63	**37.09**±18.25	81.81±52.51
SNR	7.98±6.45	-0.08±3.55	**14.10**±3.41	8.26±6.22
CosS	0.93±0.06	0.86±0.07	**0.98**±0.02	0.93±0.07

A Appendix

A.1 Supplementary results of section 3.2

FCN+DAE [7]. The authors introduce a novel denoising algorithm utilizing a 13-layer Fully Convolutional Network (FCN) based Denoiser Autoencoder (DAE), where the decoder's objective is to reconstruct a signal based on the low-dimensional features generated by the encoder.

Wavelet transforms [4]. Wavelet transforms is a well-known technique used in signal processing for analyzing signals by cutting them up into different frequency components [23]. Because of the nature of the ECG signals, we look at the *Daubechies wavelets* family as they are similar in shape to the QRS complex, and their energy spectrum is concentrated around low frequencies [24]. The wavelet transforms approach leverages the signal's energy at different scales to effectively separate the noise from the ECG signals [6]. Nonetheless, it is worth noting, as highlighted in the literature [6, 5], that this method tends to be effective primarily in cases where the ECG signal is not severely corrupted, often struggling when confronted with high-amplitude noise.

Torsades-de-Pointes (TdP) risk prediction [9]. Torsades-de-pointes (TdP) is a life-threatening arrhythmia, which can be congenital or drug-induced, and it is associated with long QT intervals. Given the potential for sudden death associated with this condition [25], its study has garnered significant interest within the scientific community. Notably, in the study conducted using the `Generepol` dataset, the subjects were recorded ECGs both before and 1, 2, 3, and 4 hours after receiving an oral dose of 80mg of *Sotalol*, an anti-arrhythmic drug known to be associated with TdP [26].

A.2 Supplementary results of section 3.3

Publicly available code at `https://git.ummisco.fr/open/2023-denoising_impact`.

A.2.1 Evaluation of the standalone denoising task

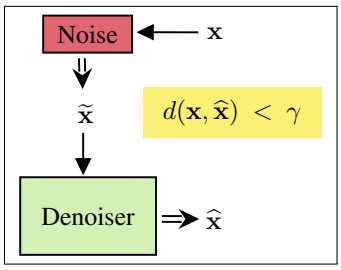


Figure 5: We denote with d any of the metric presented in section 3.1 and with $\gamma \in \mathbb{R}$ the corresponding threshold parameter.

We analyze how the methods described in section 3.2 perform when it comes to denoising on the `Generepol` dataset. In Figure 5 we show the evaluation pipeline. All the denoisers requiring a training phase have been trained as in [5]. In particular, the noise we consider throughout this paper is *basal wander* (or *basal drift*) with variable intensities, i.e., the effect where the base axis (x-axis) of a signal appears to 'wander' or move up and down rather than be straight [4].

To maintain consistency with other SOTA methods, we used the noise obtained from the Physionet MIT-BIH Noise Stress Test Database (MIT-BIH NSTDB) [19][20]. The dataset contains three types of noise including *baseline wander*, noise signals are recorded alongside clean ECGs. We superimposed the `Generepol` signals to

calibrate the amount of noise signals using the *nst* tool provided by Physionet. The amount of noise added is estimated in decibels dB. We added noise signals at 18dB and 24dB.

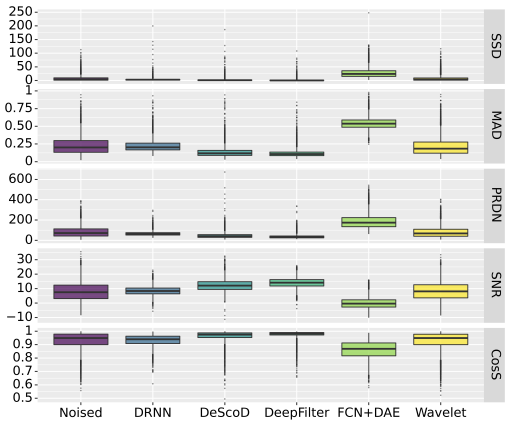
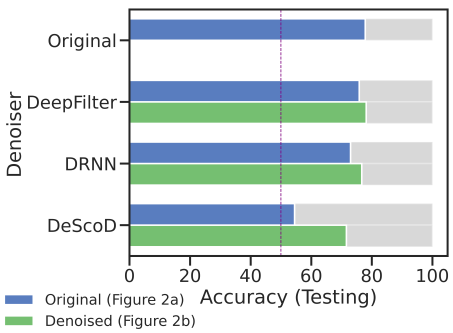


Figure 6: We remind that for `SSD` eq. (1), `MAD` eq. (2), and `PRDN` eq. (3) the lower the better; for `SNR` eq. (4) the higher the better; `CosS` eq. (5) closer to 1 the better.

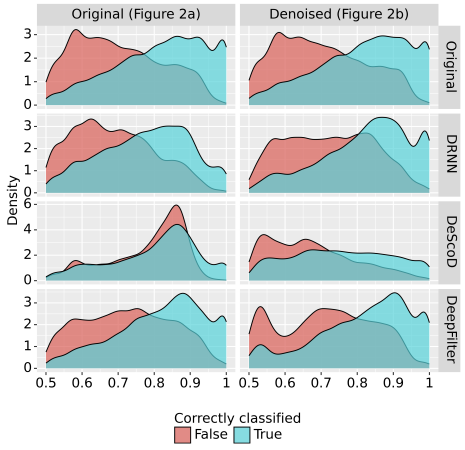
As we can see from Figure 6, even when evaluated solely for the denoising task on `Generepol`, the denoisers exhibit results consistent with those reported in their original papers. The poor performance of FCN+DAE may be attributed to a possible shift in signal reconstruction. It is worth noting that the model generates heartbeats with 512 points, even when provided with input heartbeats of only 500 points. To mitigate this discrepancy, we performed additional signal down-sampling. Finally, Wavelet transform yielded results closely resembling the original signal with noise Table 2. While this demonstrates its effectiveness when the signal has minimal noise, it offers limited denoising capabilities when confronted with higher levels of noise. Due to the unstable performances of the FCN+DAE and Wavelet, we have excluded these methods from the subsequent analysis with the TdP classifier.

A.2.2 Effect of the denoising on TdP risk prediction

We extend our analysis to a publicly available dataset `PTB-XL` [18] containing 10-second 12-lead ECGs sampled at 500Hz labeled as *normal*, *myocardial infarction*, *ST/T change*, *conduction disturbance* and *hypertrophy*. A binary classification task was created by putting all samples labeled as not normal in a single category. The final dataset consists of 96533 *normal*, class 0, and 82747 *abnormal*, class 1. Therefore we trained the same DenseNet architecture as for the TdP risk prediction in `Generepol`. We show in fig. 7a the results in terms of accuracy, and in fig. 7b the distribution of the predicted class for correctly and incorrectly classified samples.



(a) Accuracy



(b) Distributions of correctly and wrongly classified samples.

Figure 7: `PTB-XL` results. We remind that with 'Original' we indicate the setting in Figure 2a; with 'Denoised' we indicate the setting in Figure 2b.

A.2.3 Implication on misclassification detection

Table 3: Training TdP on original data.

Generepol	AUROC↑	FPR$\downarrow_{95\%}$
Original	**84.68**	**53.21**
DRNN	68.92	80.48
DeepFilter	64.81	82.03
DeScoD	58.04	92.55
PTB-XL	AUROC↑	FPR$\downarrow_{95\%}$
Original	**73.77**	**74.17**
DRNN	69.76	80.29
DeepFilter	70.26	78.66
DeScoD	55.84	82.43

Table 4: Training TdP on denoised data.

Generepol	AUROC↑	FPR$\downarrow_{95\%}$
Original	**84.68**	**53.21**
DRNN	79.91	62.23
DeepFilter	72.09	74.69
DeScoD	71.66	74.52
PTB-XL	AUROC↑	FPR$\downarrow_{95\%}$
Original	**73.77**	**74.17**
DRNN	71.67	75.32
DeepFilter	70.51	75.46
DeScoD	67.53	82.41

Misclassification detection is a hot topic in Machine Learning (ML) safety, focusing on identifying instances with potentially incorrect model predictions [21, 27, 28]. DOCTOR [21] is a simple method that aims to identify whether the prediction of a classifier should (or should not) be trusted so that, consequently, it would be possible to accept it or reject it. We conducted simulations using DOCTOR on the TdP risk classifiers in order to study the effect of denoising when the misclassification task is involved. In particular, we chose such a method to be applied to any pre-trained model, and it did not require prior information about the underlying dataset. The detector was defined as

$$D_\alpha(\mathbf{x}, \gamma) = \begin{cases} 1, & \text{if Gini}(\mathbf{x}) \geq \gamma' \cdot (1 - \text{Gini}(\mathbf{x})) \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

where $\text{Gini}(\mathbf{x}) \stackrel{\text{def}}{=} \sum_{y \in \mathcal{Y}} p_\theta(y|\mathbf{x})(1 - p_\theta(y|\mathbf{x}))$ is the probability of incorrectly classifying the feature $\mathbf{x}$ if it was randomly labeled according to the model distribution. Therefore, the higher $\text{Gini}(\mathbf{x})$, the higher the probability of $f_\theta(\mathbf{x})$ is of being wrong. The performances of the detector are evaluated in terms of False positive rate at 95% of True positive rate (FPR$\downarrow_{95\%}$), the lower the better, and AUROC (AUROC↑), the higher the better. We refer to [21] for a complete discussion.

In Table 4 and **??** we provide the summary of the results. Interestingly, even when the TdP models are trained on denoised data DOCTOR is not able anymore to distinguish the correctly classified samples from the incorrect ones due to the alteration in the posterior distribution of the models.

Chapter 5

Segmentation and Quality Evaluation of Electrocardiograms Signals

Chapter Summary

5.1 Introduction

ECGs are essential for the detection, diagnosis, and monitoring of various cardiac diseases, including arrhythmias, myocardial infarction, and heart failure. A critical step in the analysis of ECG signals is the accurate identification of characteristic waveform components such as the P wave, QRS complex, and T wave. This process, known as ECG segmentation, enables medical professionals to make well informed decisions regarding patient care and management. Identification of ECG features includes detection of QRS complexes where the R peak is the most commonly detected, P and T waves, followed by an analysis of their shapes, amplitudes and relative positions to QRS complexes. The process of detecting positions of P, T waves and QRS complexes is called segmentation of the ECG signal. While ECG segmentation has been a topic of extensive research for decades, several challenges persist. Traditional ECG analysis methods, such as rule-based algorithms, template matching, and wavelet transform, have shown limitations in handling noisy signals, morphological variations, and irregular heart rhythms. Such variations come from patients heterogeneity, difference in recording equipment, placement of electrodes, respiration, interference arising from sudden movements, muscle noise, ambient electrical interference, etc. Noisy ECGs are extremely difficult to segment automatically. Moreover, some ECG leads may not contain all common features, for example, P wave or Q peak may be missing. This task can be even more challenging in ECGs recorded from patients with pathologies or under drug treatments. Additionally, these methods often require manual finetuning and expert knowledge, which can lead to increased analysis time and potential errors. In current clinical practice, health practitioners, especially cardiologists, look for subtle abnormalities in wave morphology and the periodicity of repeating features measured "by hand" to provide diagnosis.

Besides requiring a significant amount of time and effort, not all practitioners are able to measure these features correctly. For instance it has been shown that only 50% of general practitioners and 80% of rhythomologists can measure the QT distance correctly [152]. Such measurements can be very hard to perform by hand in large datasets. The increasing prevalence of cardiovascular diseases, coupled with the rapid development of wearable and remote monitoring technologies, has resulted in a growing need for accurate, robust, and automated ECG segmentation techniques. These techniques must be capable of efficiently handling diverse and large scale ECG data in real world scenarios. Therefore, research efforts have shifted toward leveraging advanced machine learning and artificial intelligence methodologies, with neural networks emerging as a promising approach.

Automatic approaches for segmentation include wavelet transformation [153][154], classical machine learning algorithms [155] or digital signal processing techniques [156]. Classical machine learning techniques require the identification of specific features from the ECG as inputs and can predict position of target areas on the signal. These approaches rely on feature engineering, which is an essential step to transform the raw ECG data into a suitable internal representation from which the model is able to extract important regions. However, in most cases feature engineering with human intervention is very challenging, sometimes lacking precision and extremely time consuming. To overcome ML limitations such as feature engineering, deep learning (DL) approaches are able to automatically extract hidden important features through various steps of data transformations and representations. They often outperform many state-of-the-art (SOTA) techniques based on manual feature engineering and ML.

In this study, we propose an original deep learning framework for ECG segmentation. We designed UNet [157] like neural networks coupled with our denoising autoencoder, DeepFADE, which cancels noise and reconstructs malformed waveform of the signal to extract the following segments:

- P wave with coordinates of onset and offset
- QRS complexes with coordinates of Q, R and S peaks
- T wave with coordinates of onset and offset

Finally, the ECG data along with the segmentation information is used to compute a quality score, which is applied to filter out low quality signal, leaving the user with high quality segmented ECG data. We evaluated the performance of the models on two public datasets: the *PhysioNet QT database (QTDB) [158][6] and Lobachevsky University Electrocardiography Database (LUDB) [159][160][6].* **F1-Scores for P waves, QRS complexes and T waves detection display very high performances**, respectively 98.89%, 99.99% and 99.87%, which surpasses SOTA methods. To assess the performance of our quality score evaluation algorithm, we run the framework pipeline on the *PhysioNet/-Computing in Cardiology Challenge 2011* [6] and compared quality scores.

5.2 Related Works

ECG analysis that both focuses on the heartbeat morphology as well its rhythm is a very powerful tool in diagnosing certain cardiac conditions. The most important regions in the heartbeat are the P wave, which is usually between 120-200ms is typically small in time and amplitude, and positive relatively to the isoelectric line. The P wave is then followed by the QRS complex, which comprises the Q, R and S peaks. The most important component in the QRS complex is the R peak, which is always present on all leads. However, some peaks, like the Q peak, may be harder to detect or invisible in some leads. The QRS complex usually has a duration between 80-112ms [161]. R peak is the common way to delineate the heartbeat by centering the peak in a window of -400ms to +600ms equivalent 1-second containing 500 time-points. [161]. The QRS complex is followed by the T wave and typically last between 350ms to 430ms [161]. Longer duration for any of the regions

may lead to heart rhythm disruption or result in cardiac disorders. Extracting those regions from the heartbeat is important to develop automated diagnosis tools, localized analyses or interpretability algorithm. In 2021, Prifti et al. [46] proposed a novel DL approach based on a DenseNet [134] architecture to evaluate the occurrence risk of a particular type of arrhythmia, Torsade-de-Pointes (TdP) [136][137][46]. The method topped a 98% accuracy in predicting the footprint of a drug known to increase the risk of TdP. They went further in interpreting the model's output using an approach based on input perturbation (i.e. occlusion) to compute a feature importance score. To summarize the feature importance score in relation to a prototype heartbeat of the studied condition [162][46], the authors used a segmentation approach on the ECGs and overlapped the feature importance score. The method consisted on identifying the R peaks position and extracting a 1-second window for each heartbeat. The visualization of the areas of interests was very useful to field medical experts as it provided insights on how the model was performing classification.

Among the various SOTA methods for ECG segmentation, some rely on ML methods, which require manual feature extraction along with digital signal processing techniques [154][163]. In their survey, Roopa et al. [155] highlighted recent approaches for ECG segmentation, the main being ML techniques, Hidden Markov models (HMM), neural network methods and genetic algorithms. In their paper, Andreao et al. [153], introduced an approach based on HMM combined with wavelet transformation. Their framework required a feature extraction step through wavelet transformation to represent the original ECG signal in a scale-time space. Based on the extracted features they trained several HMM models to match ECG waveform patterns. Their average precision for P wave, QRS and T wave were 90.54%, 99.95% and 99% respectively.

Other methods based on Support Vector Machine (SVM) [164], Random Forest [165], Naive Bayes [166] or even rule-based methods have shown significant results in segmenting ECGs. Martinez et al [154] introduced a wavelet based method for segmenting ECG and achieved high performances of overall sensitivity of 99.66% and positive predictivity (PPV) of 99.56%.

More recently, automated methods based on NN have shown significant results in segmenting ECG. In 2021, Peimankar et al. [163] proposed a neural network based on long short-term memory (LSTM) [167]. They used a preprocessing step for noise reduction and achieved F1 score of 93.01%, 99.45% and 96.12% respectively for P wave, QRS complex and T wave. Another study using NN, Moskalenko et al. [168] proposed a method based on convolutional network (CNN) [169] with a UNet architecture [157] trained n the PhysioNet LUDB dataset [6][159][160]. They achieved 97.8% of F1-score for P wave, 99.5% for QRS complex and 99.9% for T wave.

One significant drawback of most segmentation methods is their tolerance to noise. Indeed, noisy ECGs with baseline drift are extremely difficult to segment automatically. Besides cancelling noise in ECGs, evaluating the quality of segmentation is crucial for analyses that strongly depend on it. Most popular methods rely on wavelet transformation. In 2018, Zhao et al. [5] proposed a method for evaluating the overall quality of the ECG based on heuristic fusion and fuzzy comprehensive evaluation of the Signal Quality Indexes (SQI). They used simple heuristic fusion to extract SQIs and determine the following SQIs: R peak detection match qSQI, QRS wave power spectrum distribution pSQI, kurtosis kSQI, and baseline relative power basSQI. Then, combined with Cauchy distribution, rectangular distribution and trapezoidal distribution, the membership function of SQIs was quantified, and the fuzzy vector was established. The bounded operator was selected for fuzzy synthesis, and the weighted membership function was used to perform the assessment and classification. Their method produced three different modalities to assess the ECG quality : "Excellent", "Barely acceptable" and "Unacceptable". Despite good performances, the quality estimation is global meaning an attribute is given to the whole ECG and not to each heartbeat.

5.3 Methodology

This section covers our proposed methodology for segmenting ECG and computing quality scores. We implemented an autoencoder (AE)-like architecture [170] to segment specific components of the signal P wave, QRS complex and T wave. The architecture is based on the Fully Convolutional

Network (FCN), UNet [157] architecture. To overcome noise interference, we trained and used a denoising autoencoder, our method DeepFADE, to significantly reduce it compared to the original ECG signal, but also to correct baseline drift and other anomalies from the signal. ECGs were standardized to avoid the problem of vanishing gradient and speed up training with faster convergence.

$$ECG_{standardized} = \frac{ECG - mean(ECG)}{std(ECG)} \tag{5.1}$$

Traditional segmentation NN require large datasets with heavy network architectures involving contracting convolution layers where downsampling is progressively applied. In their paper [171], Ciresan et al. introduced a NN consisting of successive convolutional, max-pooling and fully connected layers, which they trained to segment bio-medical images on 3 millions samples. Although they achieved good performances, the network required a very large number of samples, which are not always available for most studies. To counter this issue, in 2015 Ronnerberger et al. [157] published a new approach for segmentation using FCN. The network was based on an AE-like architecture and was designed to operate with smaller datasets. The structure of the network is similar to AEs with a first part that compresses the input (the contracting part), and the second part of the network, which is symmetric to the first part and up-samples specific features (the expansion part). Unlike AEs, the up-sampling process is performed by deconvolution [172] layers and skip connections are made between the contraction and expansion part. In fact, in the later, each layer takes as input previous layer as well as the corresponding symmetric layer in the contracting part. This connection between each symmetric layer of the two parts allows the expansion process to increase the resolution of the output and to :

- recover lost features and propagate context information to higher resolution layers
- capture precise localization of target areas during expansion
- assemble more precise output based on the information from the contracting part

By doing so, this method requires much less data for training. Beyond images, this method has already been used for ECG segmentation. Moskalenko et al. [168] used this architecture for segmenting ECGs. Despite having good performance, this method can lead to degraded performances when the noise on the signal increases significantly.

We capitalized on this work, which was the starting point of our approach. First we preprocessed the ECG for noise cancelling and baseline drift correction. Then we trained three NN dedicated to each component : P, T waves and QRS complex. Each NN is a lightweight UNet model trained for specific area segmentation which outputs precise localisation of components even with significant noise in the initial ECG. Each network as illustrated in Figure 49, is composed of :

- Contraction part
  - 5 successive convolution blocks witch each made of : [1D convolution, batch-normalization, Leaky ReLU activation, dropout] x 2, max-pooling
  - all convolutions have a kernel size of 5, padding of $kernel//2$ and strides of 1, filters start at 32 and grow after each blocks by a *growth_rate* hyper-parameter
  - the max-pooling steps are : 2, 2, 2, 5, 5. At the end of contraction the signal is encoded from 5000 points to 25 points
  - dropout
  - 2 successive convolution blocks
- Expansion part
  - symmetric blocks and layers as of contraction part

- max-pooling is replaced with deconvolution
- each layer takes as input a concatenation of previous input and symmetric contraction layer
- output is made of two consecutive convolution block with 8 filters and 2 filters, the final activation is the $Softmax$ function which gives a confidence score for each point of the mask whether it belongs to the target area or not. The final output for each model has the shape of $(x, 5000, 2)$, where x is the number of samples.

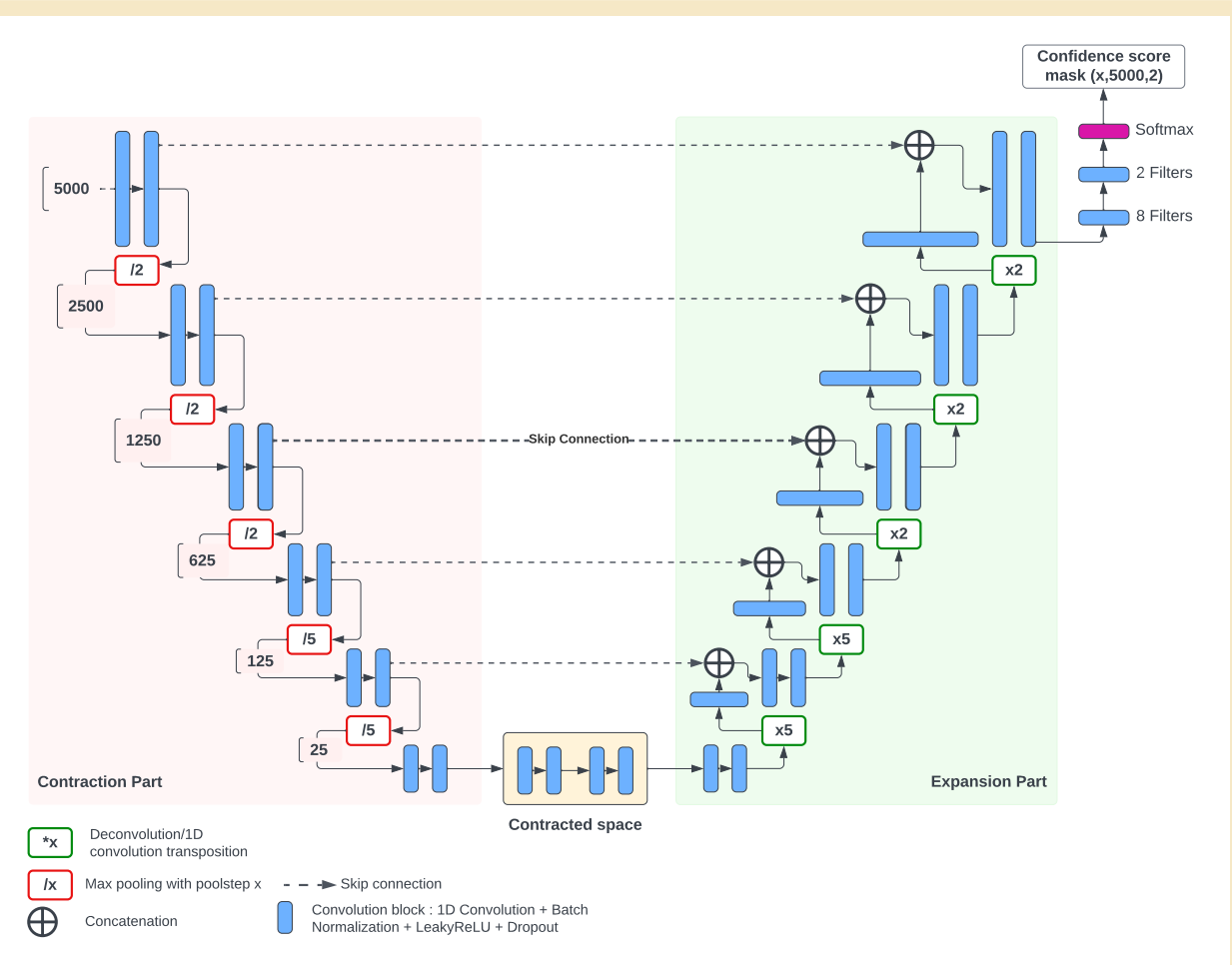


Figure 49: ***Architecture of the segmentation models****: the left red side corresponds to the contraction part where convolution layers and successive down-sampling operations are applied to an initial vector size of 5000 time-points which is contracted to 25 time-points. The contracted space is then expanded through the expansion part in the right side in green color through deconvolution layers which are outputs are concatenated to symmetric contraction layer.*

5.3.0.1 Training Process

Data used for training are ECGs where labels are binary masks in which *1* correspond to coordinates of P wave onset, P wave offset, Q, R, S, T wave onset and T wave offset and *0* otherwise. Each NN model outputs a mask of 5000 values interpreted as a confidence score of belonging to the target area.

5.3.0.2 Loss & Metric Functions

We used the *Dice Coefficient* (F1-Score) as accuracy metric. Dice Coefficient is a statistic method to evaluate similarity or overlapping between two samples. It's equation is given by :

$$Dice_{coefficient} = \frac{2 * \sum_i^N (Y_{mask} \cap \hat{Y}_{mask})}{\sum_i^N Y_{mask} + \sum_i^N \hat{Y}_{mask} + \epsilon} \tag{5.2}$$

Along with Dice Coefficient we used the Dice Loss, which is commonly used in the field of medical data segmentation and is given by :

$$Dice_{loss} = 1 - Dice_{coefficient}(Y_{mask}, \hat{Y}_{mask}) \tag{5.3}$$

5.3.0.3 Computing Quality Score Of The Heartbeat

Within our framework, we implemented an algorithm to detect outliers/misplacement in predicted mask. Its main goal is to discard false positive P and T wave onsets and offsets, Q and S peaks. We assume R peaks are correctly identified as shown in our results below. The algorithm is described as follows :

- Segment each beat by centering the R peak on a 500 point window (1s) with -400ms and +600ms from the R
- Compute average QRS and standard deviation values
- Compute average distances and positions for P and T wave (onsets and offsets) relatively to R peak on all beats
- For each beat we discard Q peak after R peak and S peak before R peak, we compare the remaining with previously computed average distance and discard the ones that doesn't fit
- For each beat we apply the same pattern for P and T wave coordinates to discard wrong coordinates

To compute a quality score of the ECG, an extension of the outlier discarding process was made. In fact, we calculated an average representation for T wave, P wave and QRS complex on each single ECG and on the whole dataset along with standard deviation. For each beat of the ECG we then interpolated the distance between QRS complexes, T waves, P waves. To avoid errors that might be induced by arrhythmia thus tampering with the distance between segments, we computed average distances based on all ECGs and also took into account relative distance between P, T waves and QRS complex. The algorithm output is a scoring vector of the same length as the ECG which values range from *0* to *1*. This score relies heavily on the segmentation outputs as it uses segment coordinates to compute distances. The global framework pipeline is illustrated in Figure 50.

5.4 Experimental Framework And Results

5.4.1 Datasets

We used Physionet databases QTDB [158][6] and LUDB [160] for training, validating, evaluating and testing all three models. The QTDB dataset contains 105 ECGs of fifteen minutes long recorded on two leads. They are all annotated with onset, offset of P, T waves and markers of QRS complex. Fifteen-minute recordings were divided into 10-second chunks. Both leads were used independently for a total number of 18900 samples. The LUDB dataset contains 200 10-second ECG recordings on 12 leads for a total of 2400 samples with single leads. Both datasets were structured into smaller partitions for the training and validation processes. Table 12 details datasets partitions. To evaluate the

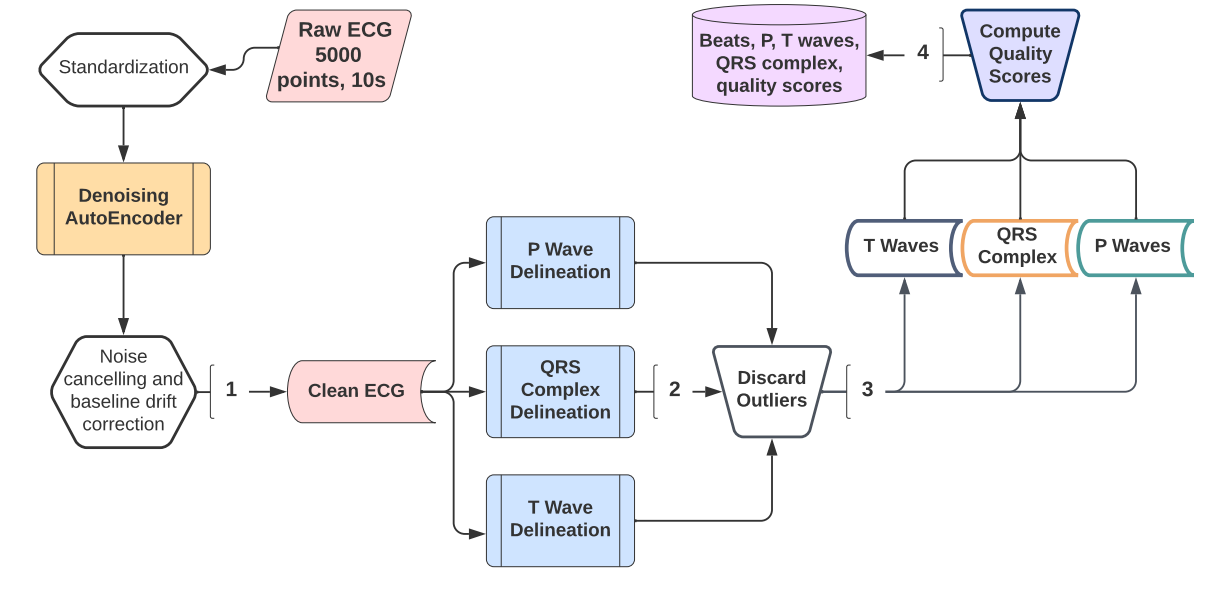


Figure 50: *ECG segmentation framework.*

performance of the quality scoring algorithm, we tested it on the *PhysioNet/Computing in Cardiology Challenge 2011* [6] dataset. It contains 10-second ECGs sampled at 500Hz, each ECG was reviewed by a group of 3 to 18 annotators working independently, they average grades were combined in 3 groups: acceptable, indeterminate and unacceptable. Approximately 70% of the collected records were assigned to group 1, 30% to group 3, and fewer than 1% to group 2. Due to extremely low number of samples, group 2 was discarded and only acceptable and unacceptable qualifications were considered.

P, T Wave, QRS Complex Segmentation				
	Holdout	**Experimentation**		
		Training	Validation	Evaluation
Samples	21300			
	(10%) 2130	(75%) 15975	(5%) 1065	(10%) 2130

Table 12: Datasets Partitions

5.4.2 Training & Hyper-Optimization

We performed a hyper-optimization process to find optimal combination of parameters specific to each of the three final models, which share the same architecture. Table 13 relates all hyper-parameters that were explored. We used the grid search algorithm to test different combinations. The validation partition dataset was used while training to assess progression of models performances. We set the maximum number of epochs to 400, although early stopping callback was used to monitor validation loss and stop training after 20 consecutive epochs without improvements. Adaptive learning rate was used for the optimizer with a reduction factor of 0.5 and a patience of 10 epochs while monitoring validation loss. The evaluation partition dataset was used to evaluate best model candidates and select the best ones. Adam optimizer was used for each model with a fixed batch size of 128 split across 2 GPUs.

Parameters	Description	Value
Initial filters	Number of initial filters for first convolution layer	16,32,64
Growth Rate	Rate by which convolutions filters are increased	2,12,24
Pool Steps	Vector of max-pooling steps. Each value correspond to a pool-size step	[2,2,2,5,5], [2,2,2,5], [2,2,5,5]
Dropout Rate	Fixed dropout rate	0, 0.2, 0.5
Kernel	Fixed kernel size for all convolutions	3, 5, 7
Activation	Fixed activation function	ReLU, LeakyReLU (alpha = 0.1)
Learning rate	Adam optimizer initial learning rate	0.01, 0.001, 0.0001

Table 13: Hyper-parameter optimization

Postprocessing: The model architecture outputs a mask of shape $(x, 5000, 2)$, where the first dimension is the number of samples, the second dimension corresponds to the time-points and the last dimension is a softmax value between 0 and 1. The first value corresponds to the score of the corresponding time-point not overlapping the target area whereas the second value corresponds to the score of the corresponding time-point overlapping the target area. Time-points of the mask, whose non-overlapping score with target area is > 0.5 are set to 0, so time-points which are not set to 0 corresponds to segmented area.

Evaluating ECG Quality Score: After training all three models, we computed a quality score per beat based on segmented data, specifically : Q,R,S peaks and P,T waves. Despite prepossessing ECGs with DeepFADE, some parts of the ECG were not successfully recovered or reconstructed. Therefore, we used quality score to discard malformed ECG beat with a threshold of 0.5.

Computational Resources And Code: All NN were implemented in both Tensorflow 2.x and Apache MXNet 1.9 with python 3.8.10. Computations were performed on an HPC cluster consisting of : 4 GPU Nvidia Tesla P100 with 12 GB of memory, 3 GPU Nvidia Tesla T4 with 16 GB of memory, 104 CPU cores of 2.20 GHz and a total of 504 GB of RAM. A parallel and distributed model training process across the cluster in was implemented in Cython. The code is available in a dedicated gitlab project.

5.4.3 Results and Discussion

The three best specific models scored the following F1 Score *($Dice_{coefficient}$)*: P Wave (98.89%), QRS Complex (99.99%) and T Wave (99.87%). These performances are computed prior to discarding malformed beats. Denoising and restructuring of ECG before segmenting improved even more performances as the models are less affected by noise interference, which may lead to misrepresentation of the ECG by the models. In fact the DAE, DeepFADE was able to denoise ECGs corrupted with noise of signal-to-noise-ratio (SNR) 25, denoised ECG SNR was 24.567. DeepFADE also successfully removed baseline drift and reconstructed parts of the signal that were malformed. To assess the contribution of preprocessing ECGs before segmenting, we added additional noise to ECGs and tested trained models. Table 14 details comparison between raw ECG and preprocessed ECGs where as Figure 51 illustrates a denoised and segmented ECG. After segmentation, quality scores are computed and compromised beats are discarded to ensure that further analyses only use accurately segmented and exploitable beats. Figure 52 presents a segmented ECG with quality scores.

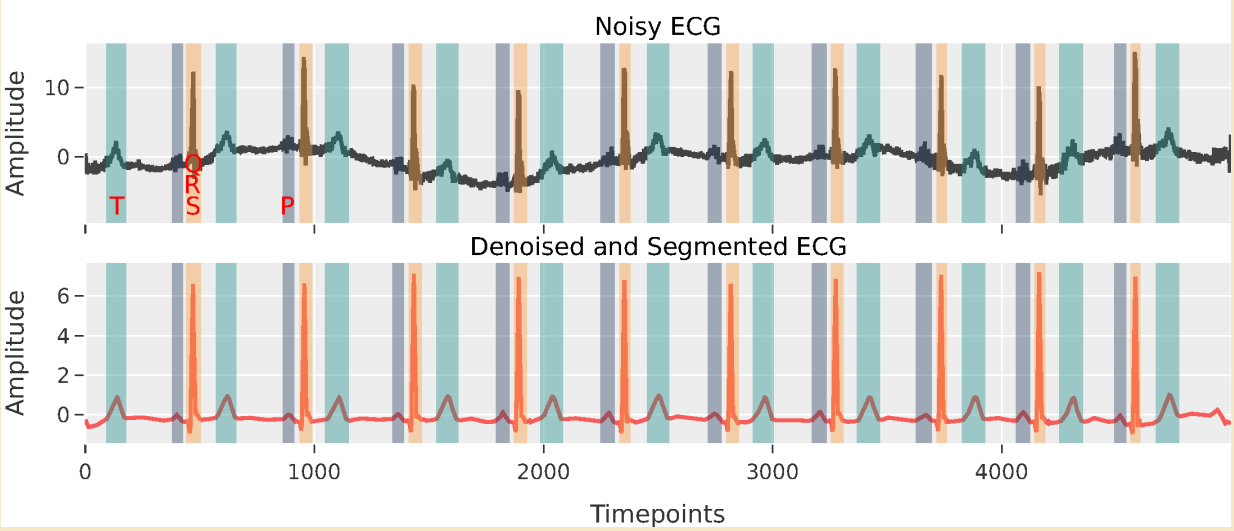


Figure 51: ***Corrected and segmented ECG with P, T waves and QRS complex.*** *The signal is pre-possessed through DeepFADE for noise cancelling and baseline drift removal. The clean ECG is then segmented and outliers are discarded to ensure precise results.*

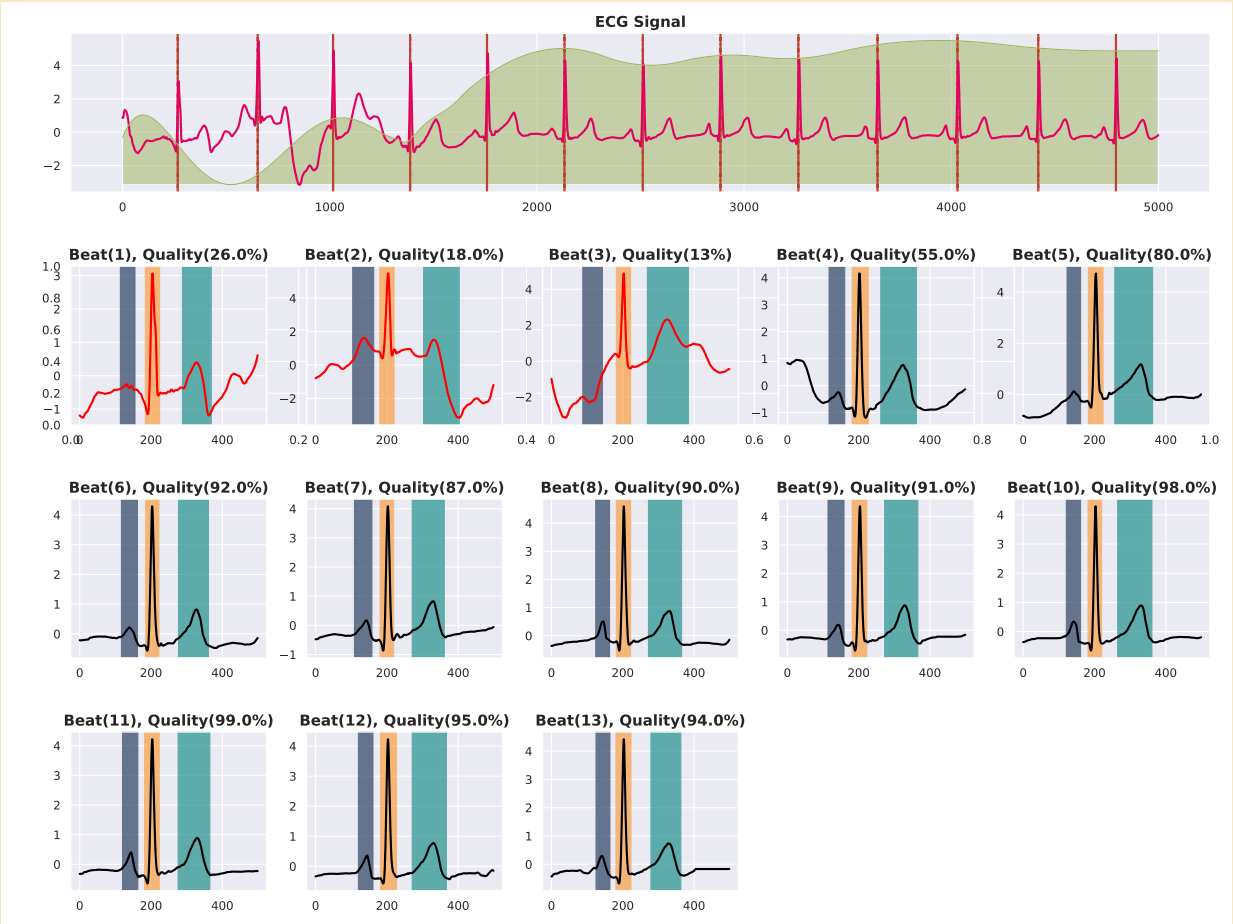


Figure 52: ***Segmentation with quality score per beat****: beat 1, 2 and 3 are colored in red as their quality score is low.*

		No DAE			DAE		
		P	**QRS**	**T**	**P**	**QRS**	**T**
Normal ECG	*F1 (%)*	98.08	**99.99**	99.58	**98.89**	**99.99**	**99.87**
	$Dice_{loss}$	0.020	0.0001	0.004	0.011	0.0001	0.001
Noisy ECG	*F1 (%)*	94.28	97.49	96.43	**98.75**	**99.87**	**99.65**
	$Dice_{loss}$	0.054	0.026	0.035	0.012	0.001	0.003

Table 14: Comparing segmentation models with DeepFADE and without

In Table 15, we compare our method to SOTA approaches with well known metrics: sensitivity or recall denoted Se, positive predictive value or precision denoted PPV and F1-Score. They are formulated as:

$$Se = \frac{TP}{TP + FN} \tag{5.4}$$

$$PPV = \frac{TP}{TP + FP} \tag{5.5}$$

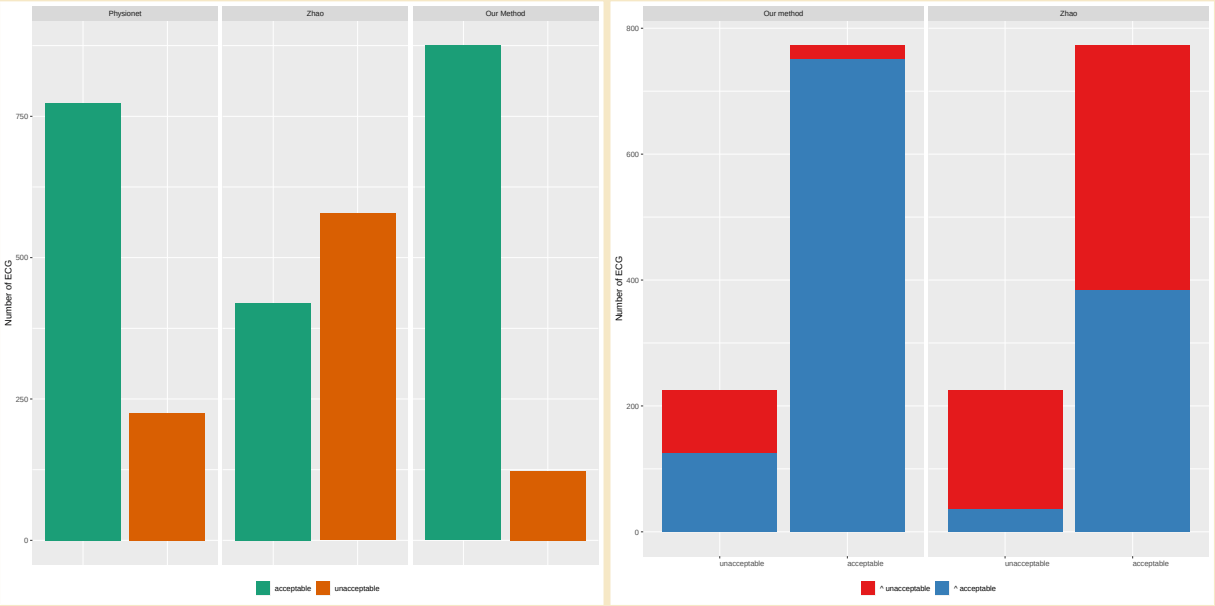
$$F1 = Dice_{coefficient} = \frac{2 * \sum_i^N (Y_{mask} \cap \hat{Y}_{mask})}{\sum_i^N Y_{mask} + \sum_i^N \hat{Y}_{mask} + \epsilon} = \frac{2 * SE * PPV}{SE + PPV} \tag{5.6}$$

	Sensitivity (%)			**Precision (%)**			**F1 Score (%)**		
Method	P	QRS	T	P	QRS	T	P	QRS	T
Mosalenko et al.[168]	98.03	100	99.22	97.71	99.93	99.41	97.87	99.97	99.56
Peimankar et al.[163]	96.53	99.70	98.75	89.74	99.19	95.44	93.01	99.45	96.12
Abrishami et al.[173]	90	95	92	92	94	90	91	94	91
Our method	**99.67**	**100**	**99.85**	**98.12**	**99.98**	**99.89**	**98.89**	**99.99**	**99.87**

Table 15: Comparing ECG Segmentation Methods

The best SOTA segmentation method which consists of a single model to segment P, T waves and QRS complex achieved respectively 97.8%, 99.5% and 99.9%. Their model was trained and tested of the PhysioNet LUDB dataset, which contains relatively clean ECG signals. Although their model performed very well, when corrupting the ECG with noise and added additional baseline drift, the F1 score dropped to: P Wave (89.1%), QRS Complex (92%) and T Wave (91%). Our approach achieves better performances on normal ECG and remain stable even on noisy ECGs.

Quality scores are evaluated on the Physionet dataset [6] and compared to method [5] on Figure 53. In Table 16 we notice better performances from our method however is prone to errors with low precision score and high false positive. However one must note that our approach overestimates quality as it is performed on clean ECGs preprocessed with DeepFADE. Further improvements would involve better support for abnormal rhythms as the method tends to be less precise on arrhythmia ECGs. NN could also be used along with the output of segmentation models to improve generalization on different ECG categories.



(a) **(b)**

Figure 53: ***Comparing our quality scores*** *with the Zhao [5] method on the Physionet/Computing in Cardiology Challenge 2011 dataset [6]*

Method	Sensitivity (%)	Precision (%)	F1 Score (%)
Our Quality Score Method	97.0	85.7	**91**
Zhao	49.6	91.4	64.3

Table 16: Comparing quality methods

5.5 Conclusion

In this chapter, we introduced a novel framework designed to enhance the analysis of electrocardiograms (ECGs) by providing accurate segmentation and quality score calculation. The framework aims to facilitate localized analyses on ECG signals, particularly where the examination of specific regions is crucial to interpret. To achieve this, our method incorporates several key components. The first component is our segmentation model, which outperforms state-of-the-art (SOTA) approaches in identifying crucial components of ECG signals. This advanced segmentation allows for more precise examinations of ECG signals and lays the groundwork for additional localized analyses. Next, we preprocessed the signals before segmentation with our denoising deep autoencoder (DAE), DeepFADE. DeepFADE is designed to effectively cancel noise and partially reconstruct corrupted waveform. By doing so, it ensures that the segmentation process remains consistently accurate even in the presence of noise or other distortions. To further enhance the utility of our framework, we have also developed an algorithm that computes a score vector for the entire ECG based on the segmentation results. This metric evaluates the quality of individual beats within the ECG and is specifically designed to avoid interference and errors in related analyses. By providing a reliable measure of ECG quality, this algorithm allows us to develop further methods focused on specific beats quality thus avoiding low quality beats that could induce non-negligeable bias. Our comprehensive framework combines advanced segmentation techniques, noise reduction through DeepFADE, and a quality score algorithm to deliver an enhanced ECG analysis. By providing accurate segmentation and quality assessment, this method sets the stage for further localized analyses on ECG signals and has the potential to improve patient care and outcomes.

Conclusion

ECGs, or electrocardiograms, are a valuable tool in medical diagnosis. This non-invasive test records the electrical activity of the heart over a period of time, helping to identify potential heart diseases. With the significant progress in the field of deep learning, automatic identification and classification of ECG pathologies have been considerably improved.

To understand how diagnoses are made from ECGs, it is important to first understand its basic structure. An ECG tracing primarily consists of P waves, QRS complexes, T waves, and occasionally, U waves, each representing a specific electrical event in the cardiac cycle. The P wave corresponds to the atrial depolarization, or the electrical activation of the atria that results in atrial contraction. Following the P wave, the QRS complex represents the ventricular depolarization, initiating ventricular contraction. The T wave subsequently corresponds to the ventricular repolarization, or the restoration of electrical charge, in preparation for the next cycle. The U wave, though not always seen, typically follows the T wave and may represent further ventricular repolarization.

The most commonly found pathologies in ECGs include arrhythmias, myocardial infarction, and hypertrophy (cardiomyopathies). Arrhythmias refer to abnormal heart rhythms and can be further categorized into numerous types such as atrial fibrillation, ventricular tachycardia, and sinus bradycardia, among others. Arrhythmias are characterized by irregularities in the heart's rate or rhythm, which can be reflected in ECGs through inconsistent intervals or abnormal waveforms. Myocardial infarction (MI), commonly known as a heart attack, involves a partial or complete blockage of the coronary arteries, depriving the heart muscle of oxygen and nutrients. ECGs can indicate MI by showing ST-segment elevation, Q-wave formation, or T-wave inversion. Cardiomyopathies, including left or right ventricular hypertrophy, exhibit changes in the amplitude and duration of the QRS complex. This is due to the increased muscle mass that comes with enlargement of the heart's chambers, which in turn, alters the path of the electrical signal.

ECGs are also invaluable in diagnosing electrolyte imbalances. For instance, hyperkalemia, or high potassium levels, can cause tall, peaked T waves, while hypokalemia, or low potassium levels, can result in flattened T waves and prominent U waves. The impact of ECGs on patient care and outcomes cannot be overstated. ECGs are quick, non-invasive, and relatively inexpensive, making them one of the first diagnostic tests done for patients with suspected heart disease. They can help guide further testing, inform treatment decisions, and monitor the effects of medications or interventions. Furthermore, ECGs provide a critical tool for real-time monitoring in critical care, perioperative, or emergency settings. Here, immediate detection of arrhythmias or myocardial ischemia can guide life-saving interventions. In community settings, the use of portable ECG devices is on the rise, particularly for monitoring arrhythmias. These devices allow for longer periods of recording, increasing the chances of catching sporadic events.

Despite the widespread usage and clear benefits of ECGs, they are not without their limitations. They are largely dependent on the skill and interpretation of the operator. Incorrect lead placement or misinterpretation of findings can lead to diagnostic errors. Deep learning, a subset of machine learning, excels in pattern recognition tasks, making it a promising tool in the automatic classification and detection of ECG pathologies, automatic segmentation, denoising and annotation.

Deep learning has demonstrated significant potential in various aspects of healthcare, including

the interpretation of electrocardiograms. By utilizing complex neural networks designed to model the way neurons interact in the brain, deep learning algorithms can process ECG data, learn from patterns, and accurately classify different cardiac conditions. The key advantage of deep learning in ECG interpretation lies in its ability to handle large volumes of data and detect subtle patterns that might be missed by the human eye. As such, these models can improve the accuracy, consistency, and speed of ECG interpretation, particularly when dealing with complex and rare arrhythmias.

Deep learning models commonly used in ECG interpretation include Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). CNNs are particularly useful for image and signal processing. They employ filters that move across the input data, in this case, ECG waveforms, to extract salient features. These features are then used to make predictions, such as classifying the ECG into normal or indicative of a specific cardiac pathology. On the other hand, RNNs, and their advanced type, Long Short-Term Memory (LSTM) networks, are effective at processing sequential data, making them ideal for time-series data like ECGs. RNNs have a 'memory' that captures information about what has been calculated so far, making them capable of recognizing patterns over time. LSTM networks, in particular, can learn and remember over long sequences, making them highly effective for ECG data where the context and sequence are important. Despite the promising performance of deep learning in ECG interpretation, there are challenges to be addressed. One key concern is the black-box nature of these models, which makes it difficult for clinicians to understand why a certain prediction was made. This lack of interpretability can hinder the adoption of deep learning tools in clinical practice. Efforts to enhance the interpretability of deep learning models include techniques like saliency maps and occlusion. These techniques visualize which parts of the ECG were most influential in the model's decision, thereby providing insight into the model's workings.

In our study, we aim at proposing and developing a new method for providing understandable interpretation of the model's decision process by finding relevant markers in the input signal. We focused on specific cardiac condition, Torsades de Pointes (TdP). TdP is a unique type of polymorphic ventricular tachycardia, characterized by a shifting sinusoidal wave form on the ECG. It has the potential to turn into more serious conditions such as ventricular fibrillation and sudden cardiac death. The name Torsades de Pointes, French for 'twisting of the peaks', is derived from the distinctive ECG appearance of this condition. The QRS complexes appear to twist around the isoelectric line, with a changing polarity and amplitude. TdP is frequently linked to an abnormal prolongation of the QT interval. This interval, measured from the beginning of the QRS complex to the end of the T wave on an ECG, represents the time taken for ventricular depolarization and repolarization. When this duration is extended, it signifies that the heart muscle is taking longer than normal to recharge between beats, creating an electrophysiological environment that can lead to the development of TdP. Because real life TdP cases data are rare, we used a surrogate approach. A particular drug, Sotalol known to prolong ventricular repolarization through IKr inhibition, which can lead to Torsade de Pointes (TdP), was used as a surrogate for IKr blockade, the major mechanism by which drugs cause QTc prolongation and predispose to TdP. The ECGs were recorded as triplicate constituting the Generepol cohort. The clinical protocol recordings consisted of several recordings before administration of the drug (basal, inclusion, SotT0) and other recordings after injection up to 6 hours (SotT1 to 6). In this dataset, each ECG contained 8 leads (I, II, V1-V6). The experiment was performed on healthy volunteers. From these data, we developed a neural network approach to predict the indirect risk score of developing TdP based on the presence of Sotalol biologic markers on the ECG. More specifically, tTe models were designed to predict Sot+ (having received Sotalol, as a surrogate for IKr blockade) and Sot- classes (normal ECG before Sotalol intake). Two categories of models were developed, one using all 8 leads and another one trained on single lead. Results showed that both categories reached high performances (> 92% of accuracy). To validate the models we tested them on additional cohorts, cLQT ECGs and real life drug induced TdP events (diTdP). The cLQT cohort contained ECGs from patients having congenital Long QT syndrome (1, 2 and 3). Congenital Long QT Syndrome (cLQT) comprises genetic cardiac disorders that can cause life threatening arrhythmias due to a prolonged QT interval on the ECG. cLQT1, the most common form, results from a mutation in the KCNQ1

gene affecting the Kv7.1 potassium channel. It often manifests symptoms during exercise or stress. cLQT2 arises from a mutation in the KCNH2 gene impacting the Kv11.1 potassium channel. cLQT3 is due to a mutation in the SCN5A gene that alters the Nav1.5 sodium channel, typically leading to arrhythmias during rest or sleep. Prolonged QT can be caused by cLQT thus possibly leading to TdP event. Individuals with cLQT2 are more prone to developing TdP. This risk stems from the underlying genetic mutation in the KCNH2 gene which encodes the Kv11.1 potassium channel. This mutation results in impaired repolarization, thus prolonging the QT interval on the ECG. The models were capable of discriminating ECGs from cLQT2 patients. This is particularly noteworthy as cLQT2 shares the same pathophysiological mechanism of IKr blockade with sotalol-induced LQTS. This result has potential clinical applications such as screening incoming patients for cLQTS and discrimination of types. The models were also tested on real TdP cases (drug induced) and showed good performances although it was lower than the ones obtained on the Generepol cohort.

To better understand the model's predictions and to find relevant markers in the ECGs responsible for the prediction, we applied interpretability tools to the models. Interpretability of AI models holds a pivotal role in understanding, explaining, and validating the workings and decisions of these models. It serves as the bridge between raw computational processes and human cognizance. This quality of a model is a measure of how well humans can understand the cause-effect relationships in the model's operation and outputs. It also includes the ability to predict the model's behavior based on changes in its input. In our context, this black box problem can have significant implications. Without interpretability, clinicians and researchers may hesitate to rely on these models, despite their accuracy, because they can't fully understand or explain the reasoning behind a given diagnosis. Trust in a diagnostic tool is crucial, particularly in healthcare, where decisions can significantly impact patient outcomes. Interpretability, in the context of ECGs means providing clinicians with an understandable correlation between the model's input (ECG signals) and output (diagnosis or prediction). In the case of TdP risk prediction, if the model determines that a patient's ECG indicates high risk, an interpretability tool could highlight the specific ECG segments or features that led to this conclusion. We started by testing state-of-the-art intepretability methods like occlusion. It is a type of perturbation-based method, where parts of the input ECG are systematically blocked or 'occluded' (replacing parts of the signal with zeroes or mean value) to see how the model's output changes by computing the delta between the initial prediction and the occluded ECG prediction. The basic idea is to determine which parts of the input data are most important for the model's decision by checking how much the output changes when each part is removed. When tested using the 8 leads model, the results were not understandable as the model combined all 8 leads together making it difficult to understand which parts of which lead contributed to the prediction. However, the single lead model trained on a specific lead of the ECG provided more understandable interpretability annotations. It highlighted segments of the ECG that are known to contribute to long QT diagnosis. The annotations also showed the progression over time of the importance of these segments. This novel work on TdP provides a new way of predicting TdP events combined with annotations to the clinicians. This approach increases trust in our methods. Despite these results, the interpretability annotation lacked precision and in some cases it was difficult to understand and interpret. Also, the technique used, occlusion, showed critical limitations. In fact, occlusion is limited to local explanations. It provides an understanding of the model's behavior on a specific instance (ECG), rather than a general understanding of the model by taking into considerations the particularities of multiple ECGs thus interpreting a general class or group of data. It helps understand which parts of a particular ECG were important for a model's decision, but it doesn't necessarily tell us about the model's decision making process in general. Furthermore, it doesn't account for context. Occlusion doesn't consider the context around the occluded region. In many models, the decision is based on the relationships between different parts of the input, not just on individual regions. By occluding part of the input, we disrupt this context, which might cause the model to behave differently than it would under normal circumstances. Other approaches, gradient-based methods like saliency maps or integrated gradients, while valuable, offer an indirect form of interpretability and often leave questions about how specific data parts contribute to the target class

prediction.

To tackle these challenges, we proposed and developed a new approach as an evolution of the occlusion technique, Evocclusion. Our novel method relies on multivariate transformations of the input data in order to take into account the relationships between different parts of the input. The nature of input transformation/perturbation is crucial as it can cause out-of-distribution situation where the model has not been trained on the resulting transformed input and therefore could predict randomly. We designed a prototype-based approach where we compute a representative ECG for each patient and each class. These prototypes have shed the light on the particularities of each group of ECG particularly in our TdP analysis on the Generepol cohort. Prototypes showed the prolongation and amplitude reduction of the T wave over time when Sotalol is administered and we also noticed change in the P wave and QRS complex specifically at the S peak. These variations have been validated by cardiologists and are known markers of prolonged QT. In Evocclusion, we use these prototypes to transform an ECG thus translating it into a different class, from basal to SotT6 and observe how this transformation impacts the model's prediction. We also used standard occlusion technique for transforming the ECG by replacing parts of the signal with zeroes or prototype's corresponding values. To explore the broad possibilities of transformations combinations, Evocclusion relies on a Genetic Algorithm. Genetic Algorithms (GAs) are search algorithms rooted in the principles of natural selection and genetics. They are designed to solve optimization problems, using a population of potential solutions, or 'individuals'. Each individual has encoded genes. The fitter individuals, measured by an objective function, have a higher probability of being selected for reproduction. The GA process begins with a randomly generated population. Each iteration or 'generation' involves selection, crossover, and mutation. During selection, individuals are chosen based on their fitness to contribute to the next generation. In the crossover phase, pairs of individuals combine their chromosomes to produce 'offspring', inheriting genes from each parent. Mutation involves random alterations in the offspring genes, adding diversity and avoiding local optimum solutions. GAs are highly useful in solving complex optimization and search problems, especially where the solution space is vast and poorly understood. In the case of Evocclusion, individual's genes encode transformations to be made on the input data. Each individual have multiple transformations, the GA evolves at finding powerful and meaningful transformations that flips the model's initial prediction closer to the opposite class while minimizing the required amount of transformations. This allows the method to search for combined transformations thus taking into consideration the inter-dependencies of features. The method is optimized to reduce its resources consumption footprint. This multivariate approach overcome the challenges of common state-of-the-art methods and provides a more understandable intepretrability annotation particularly when using prototype based transformations. The interpretability is produced in two forms. Firstly, a feature importance is computed based on gradients variations to assess the impact of best transformations on the model and secondly, the delta between initial prediction and transformed ECG prediction provides more accurate annotations of the input data. Results have shown significant performances and the ability to interpret not only single ECGs but a whole global class of ECGs, thus giving clinicians more insight on their data and the model's decision process and potentially leading to the discovery of new patterns. Actual work involves validating and improving prototype based transformations to find counterfactual that can explain a cause to effect relation on the input data. Validation process is not only performed on TdP dataset but on other studies as well and beyong ECGs including images. The potential and implication of this novel method are considerable as it can provide adapted patient's data interpretability and contribute to adaptive treatment. It can provide a better understanding of the patient's data over time and improve diagnosis. Evocclusion is also tailored to focus on some part on the input, in our case, ECGs. It focuses more on known areas of interest such as P, T waves and QRS complex. To achieve this, we developed tools to remove noise from the signal and accurately segment the signal.

Denoising and segmentation are two critical stages in the process of ECG signal processing and analysis. Their significance lies in the fact that they facilitate the extraction of useful information from

ECG signals, aiding in the detection of various heart conditions. ECG signals, which represent the electrical activity of the heart, often contain noise due to various internal and external sources. This noise can mask the underlying heart signal and make it challenging to interpret the data accurately. The primary sources of noise include power line interference, muscle contractions, respiration, and patient movement. Denoising aims at removing or reducing this noise while preserving the true heart signal. Several methods have been developed for ECG signal denoising. Among the most common are filtering techniques, such as low-pass, high-pass, and band-pass filters. These filters remove frequency components outside of a specified range, based on the fact that the ECG signal mainly lies within a particular frequency band, while most of the noise is outside this band. However, filtering techniques can sometimes distort the ECG signal, especially when the noise overlaps with the signal in the frequency domain. To overcome this, more advanced methods, such as wavelet transform and empirical mode decomposition (EMD), have been developed. Wavelet transform is a popular method due to its ability to represent data at different scales, providing a more flexible and robust denoising process. EMD, on the other hand, decomposes a signal into intrinsic mode functions (IMFs), and noise can be reduced by selectively reconstructing the signal with these IMFs. However these methods, most of the time are not adapted to the ECG waveform particularly when the signal is deteriorated. To improve noise reduction while minimizing the impact of feature removal from the signal, we developed a neural network model, DeepFADE. DeepFADE, which stands for Deep Frequency Amplitude Denoising Encoder aims at removing the noise from signal while reconstructing deteriorated parts and preserving at best the features present on the signal. A denoising autoencoder is a specific type of artificial neural network utilized for learning efficient codings of input data, while being robust to noise introduced into the input layer. This neural network falls under the broad category of autoencoders, which are used for unsupervised learning of efficient encodings. The architecture of a denoising autoencoder is quite similar to a standard autoencoder. Both consist of an encoder and a decoder. The encoder's task is to transform the input data into a compressed representation, and the decoder's job is to reconstruct the original data from this representation. However, the critical distinction is that a denoising autoencoder is trained to reconstruct the original input from a corrupted version. To create the corrupted version of input, noise is deliberately introduced to the input data. We used real life noise obtained from Physionet databases *(electrode movement, muscle artifact, baseline wander)* as well as artificial noise (Gaussian noise at different SNR levels, -6dB, 0dB, 6dB, 12dB, 18dB and 24dB, destruction of parts of the signal, baseline wander) to corrupt the ECG. DeepFADE was able to accurately reconstruct the signal while preserving at best features. However, this method still presents important challenges; one of which is assessing the relevance of removed noise. Although we consider it as noise, it could also be relevant features in the signal.

We primarily used DeepFADE to denoise signals in order to develop another accurate method based on neural networks to segment ECG waveform. Segmentation of ECG signals involves the division of these continuous time series into discrete segments that correspond to the different stages of a heartbeat. The standard heartbeat in an ECG signal is composed of a P wave, a QRS complex, and a T wave. Accurate segmentation of these components from an ECG signal is necessary for effective diagnostic and monitoring processes. Different methods and algorithms have been proposed for the segmentation of ECG signals. Traditional methods often involve thresholding and windowing techniques, or wavelet transforms to extract features. More recent advancements have seen the application of machine learning techniques to this problem. Deep learning methods have shown significant performances at segmenting ECGs. These models can automatically learn to extract useful features from raw ECG signals, without the need for manual feature engineering. The learnt features are then used to classify each point in the ECG signal into one of the wave categories (P wave, QRS complex, T wave, or baseline). Despite these advancements, challenges remain in the segmentation of ECG signals. Noise and artifacts, such as baseline wander and electromyographic noise, can distort the ECG signal and make segmentation difficult. Additionally, there is considerable variability in ECG signals between different individuals and even between different heartbeats from the same individual. This variability can make it challenging to develop a segmentation algorithm that performs well across all ECG signals. To handle these challenges we developed 3 neural network models to respectively

extract, P, T waves and QRS complex. Our method uses DeepFADE to remove noise from the signal and therefore is also trained on noisy signals. This process improves the performances of the method even with highly noised signals. Results have shown our approach surpass state-of-the-art methods.

Throughout our study we developed several novel methods around cardiac pathology diagnosis with a use case of TdP. We developed these methods in close collaboration with cardiologists. Our main contribution remains the new interpretability method, Evocclusion, which lays the path to the exploration of a vast universe of data. These methods have significantly contributed to new collaborations with partners in different countries, particularly the University of Vandarbilt in the United States. Moreover, we pioneered in 2020 a partnership with developing countries like Senegal specifically between UMMISCO and the cardiology department of the hospital Aristide le Dantec, our main goal being to translate our new methods to these countries and to adapt them to their local challenges. This adaption will also result in more robust methods adapted to different environments. They can be embedded in portable or wearable devices that provide enhanced live patient monitoring. To make our methods more accessible to a broader range of clinicians, practitioners and researchers, we initiated translationnal projects. In fact, I, with the collaboration of researchers actively participated in mentoring and supervising of engineering students to develop software applications/platforms that make our methods usable with an intuitive human-to-machine interface. These software platforms improve the impact of our novel approaches.

Bibliography

[1] Alexander Selvikvåg Lundervold and Arvid Lundervold. An overview of deep learning in medical imaging focusing on mri. *Zeitschrift für Medizinische Physik*, 29(2):102–127, 2019.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[3] Yu Zhang, Peter Tiňo, Ales Leonardis, and Ke Tang. A survey on neural network interpretability. *CoRR*, abs/2012.14261, 2020.

[4] Lilian Weng. From autoencoder to beta-vae. *lilianweng.github.io*, 2018.

[5] Zhidong Zhao and Yefei Zhang. Sqi quality evaluation mechanism of single-lead ecg signal based on simple heuristic fusion and fuzzy comprehensive evaluation. *Frontiers in Physiology*, 9:727, 2018.

[6] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

[7] Muthiah Vaduganathan, George A. Mensah, Justine Varieur Turco, Valentin Fuster, and Gregory A. Roth. The global burden of cardiovascular diseases and risk. *Journal of the American College of Cardiology*, 80(25):2361–2371, 2022.

[8] Thomas Gaziano, K Srinath Reddy, Fred Paccaud, Sue Horton, and Vivek Chaturvedi. Cardiovascular disease. *Disease Control Priorities in Developing Countries. 2nd edition*, 2006.

[9] Christopher P Cannon. Cardiovascular disease and modifiable cardiometabolic risk factors. *Clinical cornerstone*, 8(3):11–28, 2007.

[10] Sami Viskin. Long qt syndromes and torsade de pointes. *The Lancet*, 354(9190):1625–1633, 1999.

[11] Yee Guan Yap and A John Camm. Drug induced qt prolongation and torsades de pointes. *Heart*, 89(11):1363–1372, 2003.

[12] Ramesh M Gowda, Ijaz A Khan, Sabrina L Wilbur, Balendu C Vasavada, and Terrence J Sacchi. Torsade de pointes: the clinical considerations. *International journal of cardiology*, 96(1):1–6, 2004.

[13] Michael C Fishbein, Derek Maclean, and Peter R Maroko. The histopathologic evolution of myocardial infarction. *Chest*, 73(6):843–849, 1978.

[14] Kristian Thygesen, Joseph S Alpert, Harvey D White, Harvey D. White (New Zealand)* TASK FORCE MEMBERS: Chairpersons: Kristian Thygesen (Denmark), Joseph S. Alpert (USA)*, Fred S. Apple (USA) Marcello Galvani (Italy) Hugo A. Katus (Germany) L. Kristin Newby (USA) Jan Ravkilde (Denmark) Biomarker Group: Allan S. Jaffe, Coordinator (USA), Peter M. Clemmensen (Denmark) Mikael Dellborg (Sweden) Hanoch Hod (Israel) Pekka Porela (Finland) ECG Group: Bernard Chaitman, Co-ordinator (USA), Jeroen J. Bax (The Netherlands) George A. Beller (USA) Robert Bonow (USA) Ernst E. Van Der Wall (The Netherlands) Imaging Group: Richard Underwood, Coordinator (UK), William Wijns Coordinator (Belgium) T. Bruce Ferguson (USA) Philippe G. Steg (France) Barry F. Uretsky (USA) David O. Williams (USA) Intervention Group: Jean-Pierre Bassand, Co-ordinator (France), Elliott M. Antman (USA) Keith A. Fox (UK) Christian W. Hamm (Germany) E. Magnus Ohman (USA) Maarten L. Simoons (The Netherlands) Clinical Investigation Group: Paul W. Armstrong, Coordinator (Canada), Enrique P. Gurfinkel (Argentina) Jose-Luis Lopez-Sendon (Spain) Prem Pais (India) Shanti Mendis (Switzerland) Jun-Ren Zhu (China) Global Perspective Group: Philip A. Poole-Wilson, Coordinator (UK), et al. Universal definition of myocardial infarction. *circulation*, 116(22):2634–2653, 2007.

[15] Harvey D White and Derek P Chew. Acute myocardial infarction. *The Lancet*, 372(9638):570–584, 2008.

[16] Eedara Prabhakararao and Samarendra Dandapat. Myocardial infarction severity stages classification from ecg signals using attentional recurrent neural network. *IEEE Sensors Journal*, 20(15):8711–8720, 2020.

[17] Wendy Lim, Paula Holinski, Philip J Devereaux, Andrea Tkaczyk, Ellen McDonald, France Clarke, Ismael Qushmaq, Irene Terrenato, Holger Schunemann, Mark Crowther, et al. Detecting myocardial infarction in critical illness using screening troponin measurements and ecg recordings. *Critical Care*, 12:1–10, 2008.

[18] Gregory YH Lip and Hung-Fat Tse. Management of atrial fibrillation. *The Lancet*, 370(9587):604–618, 2007.

[19] Morteza Zabihi, Ali Bahrami Rad, Aggelos K Katsaggelos, Serkan Kiranyaz, Susanna Narkilahti, and Moncef Gabbouj. Detection of atrial fibrillation in ecg hand-held devices using a random forest classifier. In *2017 Computing in Cardiology (CinC)*, pages 1–4. IEEE, 2017.

[20] M Irene Ferrer. The sick sinus syndrome. *Circulation*, 47(3):635–641, 1973.

[21] Michael Semelka, Jerome Gera, and Saif Usman. Sick sinus syndrome: a review. *American family physician*, 87(10):691–696, 2013.

[22] Victor Adan and Loren A Crown. Diagnosis and treatment of sick sinus syndrome. *American family physician*, 67(8):1725–1732, 2003.

[23] Payam Safavi-Naeini and Mohammad Saeed. Pacemaker troubleshooting: Common clinical scenarios. *Tex Heart Inst J*, 43(5):415–418, October 2016.

[24] John Hampton and Joanna Hampton. *The ECG made easy e-book*. Elsevier Health Sciences, 2019.

[25] Dan M Roden. Torsade de pointes. *Clinical cardiology*, 16(9):683–686, 1993.

[26] Beny Charbit, Emmanuel Samain, Paul Merckx, and Christian Funck-Brentano. Qt interval measurement: evaluation of automatic qtc measurement and new simple method to calculate and interpret corrected qt interval. *The Journal of the American Society of Anesthesiologists*, 104(2):255–260, 2006.

[27] Irena Andršová, Katerina Hnatkova, Kateřina Helánová, Martina Šišáková, Tomáš Novotnỳ, Petr Kala, and Marek Malik. Problems with bazett qtc correction in paediatric screening of prolonged qtc interval. *BMC pediatrics*, 20:1–10, 2020.

[28] Charlotte J Haug and Jeffrey M Drazen. Artificial intelligence and machine learning in clinical medicine, 2023. *New England Journal of Medicine*, 388(13):1201–1208, 2023.

[29] Jeannette Lawrence. *Introduction to neural networks*. California Scientific Software, 1993.

[30] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[31] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

[32] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[33] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

[34] Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. *arXiv preprint arXiv:1902.09843*, 2019.

[35] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[36] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[37] Hojjat Salehinejad, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaee. Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*, 2017.

[38] Larry R Medsker and LC Jain. Recurrent neural networks. *Design and Applications*, 5:64–67, 2001.

[39] Ian J Goodfellow. On distinguishability criteria for estimating generative models. *arXiv preprint arXiv:1412.6515*, 2014.

[40] Terrance DeVries, Adriana Romero, Luis Pineda, Graham W Taylor, and Michal Drozdzal. On the evaluation of conditional gans. *arXiv preprint arXiv:1907.08175*, 2019.

[41] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[42] Olof Mogren. C-rnn-gan: Continuous recurrent neural networks with adversarial training, 2016.

[43] Fanling Huang and Yangdong Deng. Tcgan: Convolutional generative adversarial network for time series classification and clustering. *Available at SSRN 4197079*.

[44] David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. Xai—explainable artificial intelligence. *Science robotics*, 4(37):eaay7120, 2019.

[45] Feiyu Xu, Hans Uszkoreit, Yangzhou Du, Wei Fan, Dongyan Zhao, and Jun Zhu. Explainable ai: A brief survey on history, research areas, approaches and challenges. In *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II 8*, pages 563–574. Springer, 2019.

[46] Edi Prifti, Ahmad Fall, Giovanni Davogustto, Alfredo Pulini, Isabelle Denjoy, Christian Funck-Brentano, Yasmin Khan, Alexandre Durand-Salmon, Fabio Badilini, Quinn S Wells, et al. Deep learning analysis of electrocardiogram for risk prediction of drug-induced arrhythmias and diagnosis of long qt syndrome. *European Heart Journal*, 42(38):3948–3961, 2021.

[47] Joe-Elie Salem, Marine Germain, Jean-Sébastien Hulot, Pascal Voiriot, Bruno Lebourgeois, Jean Waldura, David-Alexandre Tregouet, Beny Charbit, and Christian Funck-Brentano. Genome wide analysis of sotalol-induced ikr inhibition during ventricular repolarization,"generepol study": Lack of common variants with large effect sizes. *PLoS One*, 12(8):e0181875, 2017.

[48] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[49] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[50] Hanwei Zhang, Felipe Torres, Ronan Sicre, Yannis Avrithis, and Stephane Ayache. Opti-cam: Optimizing saliency maps for interpretability. *arXiv preprint arXiv:2301.07002*, 2023.

[51] Jacob Bien and Robert Tibshirani. Prototype selection for interpretable classification. 2011.

[52] Yao Ming, Panpan Xu, Huamin Qu, and Liu Ren. Interpretable and steerable sequence learning via prototypes. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 903–913, New York, NY, USA, 2019. Association for Computing Machinery.

[53] Yupeng Wu and Cheng Lian. A novel dilated causal convolutional and transformer attention prototype network with cut feature map for ecg classification. In *2022 China Automation Congress (CAC)*, pages 981–986, 2022.

[54] Inês Neves, Duarte Folgado, Sara Santos, Marília Barandas, Andrea Campagner, Luca Ronzio, Federico Cabitza, and Hugo Gamboa. Interpretable heartbeat classification using local model-agnostic explanations on ecgs. *Computers in Biology and Medicine*, 133:104393, 2021.

[55] Alan H Gee, Diego Garcia-Olano, Joydeep Ghosh, and David Paydarfar. Explaining deep classification of time-series data with learned prototypes. In *CEUR workshop proceedings*, volume 2429, page 15. NIH Public Access, 2019.

[56] Yehualashet Megersa Ayano, Friedhelm Schwenker, Bisrat Derebssa Dufera, and Taye Girma Debelee. Interpretable machine learning techniques in ecg-based heart disease classification: A systematic review. *Diagnostics*, 13(1), 2023.

[57] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[58] Peiyu Li, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi. Motif-guided time series counterfactual explanations, 2022.

[59] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks, 2017.

[60] Awni Y. Hannun, Pranav Rajpurkar, Masoumeh Haghpanahi, Geoffrey H. Tison, Codie Bourn, Mintu P. Turakhia, and Andrew Y. Ng. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, 25(1):65–69, Jan 2019.

[61] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier, 2016.

[62] Qihang Yao, Ruxin Wang, Xiaomao Fan, Jikui Liu, and Ye Li. Multi-class arrhythmia detection from 12-lead varied-length ecg using attention-based time-incremental convolutional neural network. *Information Fusion*, 53:174–182, 2020.

[63] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On Pixel-Wise explanations for Non-Linear classifier decisions by Layer-Wise relevance propagation. *PLoS One*, 10(7):e0130140, July 2015.

[64] Nils Strodthoff and Claas Strodthoff. Detecting and interpreting myocardial infarction using fully convolutional neural networks. *Physiological Measurement*, 40(1):015001, jan 2019.

[65] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018.

[66] Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. Transforming auto-encoders. In Timo Honkela, Włodzisław Duch, Mark Girolami, and Samuel Kaski, editors, *Artificial Neural Networks and Machine Learning – ICANN 2011*, pages 44–51, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[67] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. Interpreting blackbox models via model extraction, 2019.

[68] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[69] Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(5):726–742, 2021.

[70] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization, 2015.

[71] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks. 2015.

[72] Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.

[73] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[74] Irene Sturm, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Interpretable deep neural networks for single-trial eeg classification. *Journal of neuroscience methods*, 274:141–145, 2016.

[75] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[76] Ramprasaath R Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? *arXiv preprint arXiv:1611.07450*, 2016.

[77] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. *Advances in neural information processing systems*, 29, 2016.

[78] Andy Brown, Aaron Tuor, Brian Hutchinson, and Nicole Nichols. Recurrent neural network attention mechanisms for interpretable system log anomaly detection. In *Proceedings of the first workshop on machine learning for computing systems*, pages 1–8, 2018.

[79] Xuan Liu, Xiaoguang Wang, and Stan Matwin. Improving the interpretability of deep neural networks with knowledge distillation. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 905–912. IEEE, 2018.

[80] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. *Advances in neural information processing systems*, 31, 2018.

[81] Yulong Wang, Hang Su, Bo Zhang, and Xiaolin Hu. Interpret neural networks by identifying critical data routing paths. In *proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8906–8914, 2018.

[82] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *International Conference on Machine Learning*, pages 2376–2384. PMLR, 2019.

[83] Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, and Hiroki Arimura. Distribution-aware counterfactual explanation by mixed-integer linear optimization. *Transactions of the Japanese Society for Artificial Intelligence*, 36(6):C–L44_1, 2021.

[84] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[85] Tong Wang. Gaining free or low-cost interpretability with interpretable partial substitute. In *International Conference on Machine Learning*, pages 6505–6514. PMLR, 2019.

[86] LiMin Fu. Rule learning by searching on adapted nets. In *AAAI*, volume 91, pages 590–595, 1991.

[87] Geoffrey G Towell and Jude W Shavlik. Extracting refined rules from knowledge-based neural networks. *Machine learning*, 13:71–101, 1993.

[88] Rudy Setiono and Huan Liu. Understanding neural networks via rule extraction. In *IJCAI*, volume 1, pages 480–485. Citeseer, 1995.

[89] Rudy Setiono and Huan Liu. Neurolinear: From neural networks to oblique decision rules. *Neurocomputing*, 17(1):1–24, 1997.

[90] Koichi Odajima, Yoichi Hayashi, Gong Tianxia, and Rudy Setiono. Greedy rule generation from discrete data and its use in neural network rule extraction. *Neural Networks*, 21(7):1020–1028, 2008.

[91] Richi Nayak. Generating rules with predicates, terms and variables from the pruned neural networks. *Neural Networks*, 22(4):405–414, 2009.

[92] José Manuel Benítez, Juan Luis Castro, and Ignacio Requena. Are artificial neural networks black boxes? *IEEE Transactions on neural networks*, 8(5):1156–1164, 1997.

[93] Juan L Castro, Carlos J Mantas, and José Manuel Benítez. Interpretation of artificial neural networks by means of fuzzy rules. *IEEE Transactions on Neural Networks*, 13(1):101–116, 2002.

[94] Mark Craven and Jude Shavlik. Extracting tree-structured representations of trained networks. *Advances in neural information processing systems*, 8, 1995.

[95] R Krishnan, G Sivakumar, and P Bhattacharya. Extracting decision trees from trained neural networks. *Pattern recognition*, 32(12), 1999.

[96] Olcay Boz. Extracting decision trees from trained neural networks. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 456–461, 2002.

[97] Tejaswini Pedapati, Avinash Balakrishnan, Karthikeyan Shanmugam, and Amit Dhurandhar. Learning global transparent models consistent with local contrastive explanations. *Advances in neural information processing systems*, 33:3592–3602, 2020.

[98] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[99] Feng Wang, Haijun Liu, and Jian Cheng. Visualizing deep neural network by alternately image blurring and deblurring. *Neural Networks*, 97:162–172, 2018.

[100] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196, 2015.

[101] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[102] Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *Advances in neural information processing systems*, 29, 2016.

[103] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11):e7, 2017.

[104] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549, 2017.

[105] Ruth Fong and Andrea Vedaldi. Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8730–8738, 2018.

[106] Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, Anthony Bau, and James Glass. What is one grain of sand in the desert? analyzing individual neurons in deep nlp models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6309–6317, 2019.

[107] Gregory Plumb, Denali Molitor, and Ameet S Talwalkar. Model agnostic supervised local explanations. *Advances in neural information processing systems*, 31, 2018.

[108] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

[109] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[110] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[111] Erik Strumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, 11:1–18, 2010.

[112] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[113] Marco Ancona, Cengiz Oztireli, and Markus Gross. Explaining deep neural networks with a polynomial time algorithm for shapley value approximation. In *International Conference on Machine Learning*, pages 272–281. PMLR, 2019.

[114] Tom Heskes, Evi Sijben, Ioan Gabriel Bucur, and Tom Claassen. Causal shapley values: Exploiting causal knowledge to explain individual predictions of complex models. *Advances in neural information processing systems*, 33:4778–4789, 2020.

[115] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pages 3429–3437, 2017.

[116] Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. *arXiv preprint arXiv:1702.04595*, 2017.

[117] Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, pages 883–892. PMLR, 2018.

[118] Shengjie Wang, Tianyi Zhou, and Jeff Bilmes. Bias also matters: Bias attribution for deep neural network explanation. In *International Conference on Machine Learning*, pages 6659–6667. PMLR, 2019.

[119] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017.

[120] Karthikeyan Natesan Ramamurthy, Bhanukiran Vinzamuri, Yunfeng Zhang, and Amit Dhurandhar. Model agnostic multilevel explanations. *Advances in neural information processing systems*, 33:5968–5979, 2020.

[121] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems*, 32, 2019.

[122] Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Unmasking clever hans predictors and assessing what machines really learn. *Nature communications*, 10(1):1096, 2019.

[123] Shaeke Salman, Seyedeh Neelufar Payrovnaziri, Xiuwen Liu, Pablo Rengifo-Moreno, and Zhe He. Deepconsensus: Consensus-based interpretable deep neural networks with application to mortality prediction. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[124] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

[125] Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. Representer point selection for explaining deep neural networks. *Advances in neural information processing systems*, 31, 2018.

[126] Mike Wu, Sonali Parbhoo, Michael Hughes, Ryan Kindle, Leo Celi, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez. Regional tree regularization for interpretability in deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6413–6421, 2020.

[127] Mike Wu, Michael Hughes, Sonali Parbhoo, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez. Beyond sparsity: Tree regularization of deep models for interpretability. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[128] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8827–8836, 2018.

[129] Gregory Plumb, Maruan Al-Shedivat, Ángel Alexander Cabrera, Adam Perer, Eric Xing, and Ameet Talwalkar. Regularizing black-box models for improved interpretability. *Advances in Neural Information Processing Systems*, 33:10526–10536, 2020.

[130] Ethan Weinberger, Joseph Janizek, and Su-In Lee. Learning deep attribution priors based on prior knowledge. *Advances in Neural Information Processing Systems*, 33:14034–14045, 2020.

[131] Maksymilian Wojtas and Ke Chen. Feature importance ranking for deep learning. *Advances in Neural Information Processing Systems*, 33:5105–5114, 2020.

[132] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[133] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019.

[134] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[135] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 1096–1103, New York, NY, USA, 2008. Association for Computing Machinery.

[136] W Haverkamp, M Hördt, X Chen, G Hindricks, S Willems, H Kottkamp, B Rotman, J Brunn, M Borggrefe, and G Breithardt. Torsade de pointes. *Zeitschrift fur Kardiologie*, 82(12):763–774, 1993.

[137] Carlo Napolitano, Silvia G Priori, and Peter J Schwartz. Torsade de pointes. *Drugs*, 47(1):51–65, 1994.

[138] Sarang L. Joshi, Rambabu A Vatti, and Rupali V. Tornekar. A survey on ecg signal denoising techniques. In *2013 International Conference on Communication Systems and Network Technologies*, pages 60–64, 2013.

[139] C. Haritha, M. Ganesan, and E. P. Sumesh. A survey on modern trends in ecg noise removal techniques. In *2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT)*, pages 1–7, 2016.

[140] Xunyu Zhang and Shumin Jiang. Application of fourier transform and butterworth filter in signal denoising. In *2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP)*, pages 1277–1281, 2021.

[141] Prashant Mani Tripathi, Ashish Kumar, Rama Komaragiri, and Manjeet Kumar. A review on computational methods for denoising and detecting ecg signals to detect cardiovascular diseases. *Archives of Computational Methods in Engineering*, pages 1–40, 2021.

[142] V Seena and Jerrin Yomas. A review on feature extraction and denoising of ecg signal using wavelet transform. In *2014 2nd international conference on devices, circuits and systems (ICDCS)*, pages 1–6. IEEE, 2014.

[143] Pratik Singh and Gayadhar Pradhan. A new ecg denoising framework using generative adversarial network. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 18(2):759–764, 2020.

[144] Hsin-Tien Chiang, Yi-Yen Hsieh, Szu-Wei Fu, Kuo-Hsuan Hung, Yu Tsao, and Shao-Yi Chien. Noise reduction in ecg signals using fully convolutional denoising autoencoders. *IEEE Access*, 7:60806–60813, 2019.

[145] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[146] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *International conference on machine learning*, pages 894–903. PMLR, 2017.

[147] Meinard Müller. Dynamic time warping. *Information retrieval for music and motion*, pages 69–84, 2007.

[148] L Johannesen, J Vicente, J W Mason, C Erato, C Sanabria, K Waite-Labott, M Hong, J Lin, P Guo, A Mutlib, J Wang, W J Crumb, K Blinova, D Chan, J Stohlman, J Florian, M Ugander, N Stockbridge, and D G Strauss. Late sodium current block for drug-induced long QT syndrome: Results from a prospective clinical trial. *Clin Pharmacol Ther*, 99(2):214–223, November 2015.

[149] L Johannesen, J Vicente, J W Mason, C Sanabria, K Waite-Labott, M Hong, P Guo, J Lin, J S Sørensen, L Galeotti, J Florian, M Ugander, N Stockbridge, and D G Strauss. Differentiating drug-induced multichannel block on the electrocardiogram: randomized study of dofetilide, quinidine, ranolazine, and verapamil. *Clin Pharmacol Ther*, 96(5):549–558, July 2014.

[150] George B Moody, W Muldrow, and Roger G Mark. A noise stress test for arrhythmia detectors. *Computers in cardiology*, 11(3):381–384, 1984.

[151] Joe-Elie Salem, Marine Germain, Jean-Sébastien Hulot, Pascal Voiriot, Bruno Lebourgeois, Jean Waldura, David-Alexandre Tregouet, Beny Charbit, and Christian Funck-Brentano. Genome wide analysis of sotalol-induced ikr inhibition during ventricular repolarization, "generepol study": Lack of common variants with large effect sizes. *PLOS ONE*, 12(8):1–16, 08 2017.

[152] S. Viskin et al. Inaccurate electrocardiographic interpretation of long QT: the majority of physicians cannot recognize a long QT when they see one. *Heart Rhythm*, pages 569–74, 2005.

[153] Rodrigo Varejão Andreão and Jérôme Boudy. Combining wavelet transform and hidden markov models for ecg segmentation. *EURASIP Journal on Advances in Signal Processing*, 2007:1–8, 2006.

[154] Juan Pablo Martínez, Rute Almeida, Salvador Olmos, Ana Paula Rocha, and Pablo Laguna. A wavelet-based ecg delineator: evaluation on standard databases. *IEEE Transactions on biomedical engineering*, 51(4):570–581, 2004.

[155] CK Roopa and BS Harish. A survey on various machine learning approaches for ecg analysis. *International Journal of Computer Applications*, 163(9):25–33, 2017.

[156] P. Kathirvel, M. Sabarimalai Manikandan, S. R. M. Prasanna, and K. P. Soman. An efficient r-peak detection based on new nonlinear transformation and first-order gaussian differentiator. *Cardiovascular Engineering and Technology*, 2(4):408–425, Dec 2011.

[157] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[158] Pablo Laguna, Roger G Mark, A Goldberg, and George B Moody. A database for evaluation of algorithms for measurement of qt and other waveform intervals in the ecg. In *Computers in cardiology 1997*, pages 673–676. IEEE, 1997.

[159] Alena Kalyakulina, Igor Yusipov, Viktor Moskalenko, Alexander Nikolskiy, Konstantin Kosonogov, Nikolai Zolotykh, and Mikhail Ivanchenko. Lobachevsky university electrocardiography database.

[160] Alena I Kalyakulina, Igor I Yusipov, Viktor A Moskalenko, Alexander V Nikolskiy, Konstantin A Kosonogov, Grigory V Osipov, Nikolai Yu Zolotykh, and Mikhail V Ivanchenko. Ludb: a new open-access validation tool for electrocardiogram delineation algorithms. *IEEE Access*, 8:186181–186190, 2020.

[161] The University of Nottingham cardiology teaching package. https://www.nottingham.ac.uk/nursing/practice/resources/cardiology/function/normal_duration.php.

[162] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning, 2018.

[163] Abdolrahman Peimankar and Sadasivan Puthusserypady. Dens-ecg: A deep learning approach for ecg signal delineation. *Expert systems with applications*, 165:113911, 2021.

[164] Alessia Mammone, Marco Turchi, and Nello Cristianini. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(3):283–289, 2009.

[165] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[166] David J Hand and Keming Yu. Idiot's bayes–not so stupid after all? *International statistical review*, 69(3):385–398, 2001.

[167] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[168] Viktor Moskalenko, Nikolai Zolotykh, and Grigory Osipov. Deep learning for ecg segmentation. In *international conference on Neuroinformatics*, pages 246–254. Springer, 2019.

[169] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*, pages 319–345. Springer, 1999.

[170] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning Internal Representations by Error Propagation*, page 318–362. MIT Press, Cambridge, MA, USA, 1986.

[171] Dan Ciresan, Alessandro Giusti, Luca Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. *Advances in neural information processing systems*, 25, 2012.

[172] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *2010 IEEE Computer Society Conference on computer vision and pattern recognition*, pages 2528–2535. IEEE, 2010.

[173] Hedayat Abrishami, Chia Han, Xuefu Zhou, Matthew Campbell, and Richard Czosek. Supervised ecg interval segmentation using lstm neural network. In *Proceedings of the International Conference on Bioinformatics & Computational Biology (BIOCOMP)*, pages 71–77. The Steering Committee of The World Congress in Computer Science, Computer . . . , 2018.