Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Projection-Based Photoplethysmography Signal Quality Assessment

Guilherme Chagas Suzuki

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Dr. Pedro Garcia Freitas

Brasília
2024

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Projection-Based Photoplethysmography Signal Quality Assessment

Guilherme Chagas Suzuki

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof. Dr. Pedro Garcia Freitas (Orientador)
CIC/UnB

Prof. Dr. João Luiz Azevedo de Carvalho
ENE/UnB

Prof. Dr. Eduardo Peixoto Fernandes da Silva
ENE/UnB

Prof. Dr. Guilherme Novaes Ramos
CIC/UnB

Prof. Dr. Marcelo Grandi Mandelli
Coordenador do Bacharelado em Ciência da Computação

Brasília, 17 de setembro de 2024

Dedication

I dedicate this work to the Holy Virgin Mary, to make this work hers, not mine.

Acknowledgments

Firstly, I thank the Father for allowing all of this thesis to be possible, the Son for giving a reason to work on it, and the Holy Spirit, who acted in the moments when I had been honest, resilient, and humble, virtues needed to produce results that honor the Holy Trinity. Secondly, I thank the Holy Virgin Mary for interceding for everyone's envolved true benefit. Thirdly, I thank my father Donato Sadao, my mother Maria da Assunção, my sister Alice, and my brother Gabriel for helping me through my existence and assisting the production of this thesis. Finally I thank my advisor, professor Pedro, who constantly guided me along the production of this work and even reserved entire days for assisting me. Additionally, I thank the chair members for dedicating attention to this work.

This work was supported by the Fundação de Apoio a Pesquisa do Distrito Federal (FAP-DF), by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), the Conselho Nacional de Desenvolvimento Científico e Tecnologico (CNPq), and by the University of Brasília (UnB).

Resumo

Com o rápido aumento na popularidade de dispositivos vestíveis, como smartwatches, aplicativos de monitoramento de saúde estão ganhando destaque devido à sua capacidade de monitorar uma variedade de métricas de saúde, como padrões de sono, frequência cardíaca e atividade física. Esses aplicativos costumam utilizar sensores de fotopletismografia para avaliar vários aspectos da saúde e bem-estar de um indivíduo. A fotopletismografia é um método óptico não invasivo e econômico que detecta variações no volume sanguíneo dentro da rede microvascular dos tecidos, fornecendo medições contínuas das mudanças fisiológicas ao longo do tempo. Examinar sinais de fotopletismografia permite a extração de informações valiosas sobre a saúde cardiovascular e várias métricas fisiológicas, incluindo variabilidade da frequência cardíaca, saturação periférica de oxigênio e padrões de sono. No entanto, apesar de seus benefícios, a fotopletismografia tem uma grande limitação: é particularmente suscetível a artefatos de movimento e interferências ambientais. Esses problemas podem prejudicar significativamente a eficácia dos aplicativos baseados em fotopletismografia, especialmente ao capturar sinais de fotopletismografia usando dispositivos vestíveis. Portanto, para obter medições precisas, é crucial ter sinais apropriados que sejam amostrados com alta confiabilidade.

Nesse contexto, avaliar a qualidade dos sinais é essencial para permitir a aplicação de monitoramento de saúde, já que uma alta qualidade do sinal é crucial para avaliar de forma confiável a condição médica do paciente. Para alcançar isso, algoritmos de aprendizado de máquina podem ser aplicados. Este trabalho apresenta um método inovador para avaliar a qualidade dos sinais de fotopletismografia, realizado através da fusão de projeções de sinais e técnicas de visão computacional. Para ser mais preciso, o sinal de fotopletismografia unidimensional é projetado em um conjunto de representações bidimensionais. Isso pode ser feito usando técnicas de imagem de séries temporais, como Gramian Angular Field, Markov Transition Field e Recurrence Plots, além de agregar seus resultados, o que chamamos de 'Projection Mix'. Após o pré-processamento do conjunto de dados Brno University of Technology Smartphone PPG Database (BUTPPG) em essas imagens, várias redes neurais profundas são treinadas e testadas, com hiperparâmetros selecionados através de busca heurística. Os resultados indicam que o Recurrence Plot e

o Projection Mix geralmente superaram o Gramian Angular Field e o Markov Transition Field na maioria dos modelos de visão computacional. Além disso, os métodos baseados em projeção alcançaram resultados comparáveis aos classificadores 1D de séries temporais. Por exemplo, a combinação de Wide ResNet com Projection Mix alcançou uma pontuação média de Cohen Kappa de 95,5

Palavras-chave: Fotopletismografia, Aprendizado de Máquina, Aprendizagem Profunda, Visão Computacional, Avaliação da Qualidade de Sinais, Sinais Biológicos, Projeção de Séries Temporais

Abstract

With the rapid rise in popularity of wearable devices like smartwatches, health monitoring applications are gaining traction due to their ability to monitor a range of health metrics, such as sleep patterns, heart rate, and physical activity. These applications commonly utilize photoplethysmograph sensors to assess various elements of an individual's health and wellness. Photoplethysmograph is a non-invasive and economical optical method that detects variations in blood volume within the microvascular network of tissues, providing continuous measurements of physiological changes over time. Examining photoplethysmograph signals allows for the extraction of valuable insights regarding cardiovascular health and various physiological metrics, including heart rate variability, peripheral oxygen saturation, and sleep patterns. However, despite its benefits, photoplethysmography has a major limitation: it is particularly susceptible to motion artifacts and environmental interferences. These issues can greatly impair the effectiveness of photoplethysmography-based applications, especially when capturing photoplethysmograph signals using wearable devices. Therefore, to achieve accurate measurements, it is crucial to have appropriate signals that are sampled with high reliability.

In this context, assessing the signal quality is essential for enabling health monitoring applications, as high signal quality is crucial for reliably assessing a patient's medical condition. To achieve this, machine learning algorithms can be applied. This work presents an innovative method for assessing the quality of photoplethysmograph signals, accomplished through a fusion of signal projections and computer vision techniques. To be more precise, the one-dimensional photoplethysmograph signal is projected to a set of bidimensional representations. This can be accomplished using time series imaging techniques, such as Gramian Angular Field, Markov Transition Field, and Recurrence Plots, and by aggregating their results, which is referred to as 'Projection Mix'. After pre-processing the BUTPPG dataset into these images, various deep neural networks are trained and tested, with hyperparameters selected through heuristic searching. The results indicate that the Recurrence Plot and Projection Mix generally outperformed Gramian Angular Field and Markov Transition Field across most compute vision models. Additionally, projection-based methods achieved results comparable to

1D time series classifiers. For instance, the combination of Wide ResNet with Projection Mix achieved a K-Fold mean Cohen Kappa score of 95.5% (rescaled from $[-1, 1]$ to $[0, 1]$) with a standard deviation of 0.101. An implementation of the method and experiments described in this thesis can be found at `https://gitlab.com/lisa-unb/projection-based-biological-signal-processing`.

Keywords: Photoplethysmography, Machine Learning, Deep Learning, Computer Vision, Signal Quality Assessment, Biological Signals, Time Series Imaging

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The human civilization constantly seeks improvements in life longevity and quality. One of the main research areas in that regard is the medicine, which provides means of preventing and remedying accidents and diseases. Diseases, specifically, can deteriorate a person's health silently. An example of that is the presence of atheromas, which, when untreated, can lead to lethal events such as heart attacks and strokes. For that reason, it is important to periodically search professional medical advice to detect diseases at early stages when it is still easy to treat them. However, individual professional service is expensive for most people and is unresponsive to sudden changes. Therefore, there is a demand for an automatic and constant health monitoring method.

A promising solution for that demand is the continuous healthcare monitor applications enabled by wearables. That approach is particularly possible due to the advent of Internet of Things (IoT), which is, in concept, a scalable network of interconnected devices that exchange information possibly acquired by sensors [3]. It is interesting for healthcare when those sensors extract environmental and physiological data that can hint the patient conditions. Examples of such data are body temperature, blood pressure and neural activity. We can obtain those indicators in the patient daily life through wearable devices that resemble quotidian objects, with the shape of belts, bracelets, rings, shoe soles, clothing, etc [4]. An popular example of such wearables is the smartwatches that, similarly to smartphones, can execute multiple applications, such as recording physiological data. Since devices like those support wireless connection, they can send that data to either directly to a medical staff or, as the now popular big data trend promotes, to an automatic intelligent system in the cloud. That intelligent system can, among countless patients, choose special cases that need attention. Therefore, the remote healthcare promises easier than ever access to key physiological signal data.

Despite the advantages of continuous health monitoring using smart wearables, the extraction of physiological signals is not free of interferences. PPG signals, for instance, is

under the influence of changes in illumination, low sensor quality, user skin physiognomy, adverse sensor positioning, etc. These influences can impair the signal to the point that its use becomes unfeasible. Moreover, PPG signals are highly susceptible to artifacts generated by motion or noise sources. For instance, wrist movements can disrupt the signal in a smartwatch PPG sensor, though the degree of distortion varies with the signal power and wavelength. These variation can cause high-amplitude distortions that not only can destroy the core information, but can also produce misleading events. These events are unacceptable in healthcare applications since misdiagnosis can expose the healthy patient to unnecessary risk, while lack of diagnosis can leave the unhealthy patient unattended. For that reason, it is of extreme importance to verify the quality of the signal before proceeding to further analysis on the signal. That task is known as Signal Quality Assessment (SQA) and this thesis proposes a method to achieve that goal for PPG signals.

1.1 Problem Description

Traditionally, the two most frequently used methods for evaluating the cardiac cycle and monitoring heart rate are Electrocardiogram (ECG) and Photoplethysmograph (PPG). The Electrocardiogram (ECG) has long been considered the gold standard for detecting heart rate and diagnosing cardiovascular conditions. It monitors the electrical impulses responsible for heart muscle contractions through electrodes attached to the body, usually positioned on the chest. Although ECG is the mainstay for cardiac assessments, it is not typically suitable for long-term monitoring or challenging environments due to its intricate data collection process. Conversely, PPG offers a more practical approach for observing cardiorespiratory metrics. It employs compact optical sensors and a light source to detect variations in skin color caused by blood flow following each heartbeat. The PPG measures the blood flow rate in tissues (e.g., wrist), influenced by the heart's pumping action, making it particularly effective for peripheral circulation monitoring, especially with wrist-worn or finger-mounted devices. Since both ECG and PPG gauge cardiovascular and circulatory parameters, they are interconnected, as depicted in Figure 1.1. The similarity in the signal periods of both methods suggests that either can be used to analyze metrics such as the Inter-Beat Interval (IBI). Additionally, Figure 1.1 emphasizes the reference points often utilized to assess health indicators related to blood pressure, oxygen saturation, and more. Thus, the PPG is potentially a good alternative to the ECG.

In more detail, PPG signals are optical signals that result from the interaction of light with human tissue. To extract the signal, two basic components are needed. The first is a light source, which emits light towards the tissue. An example of a light source is the

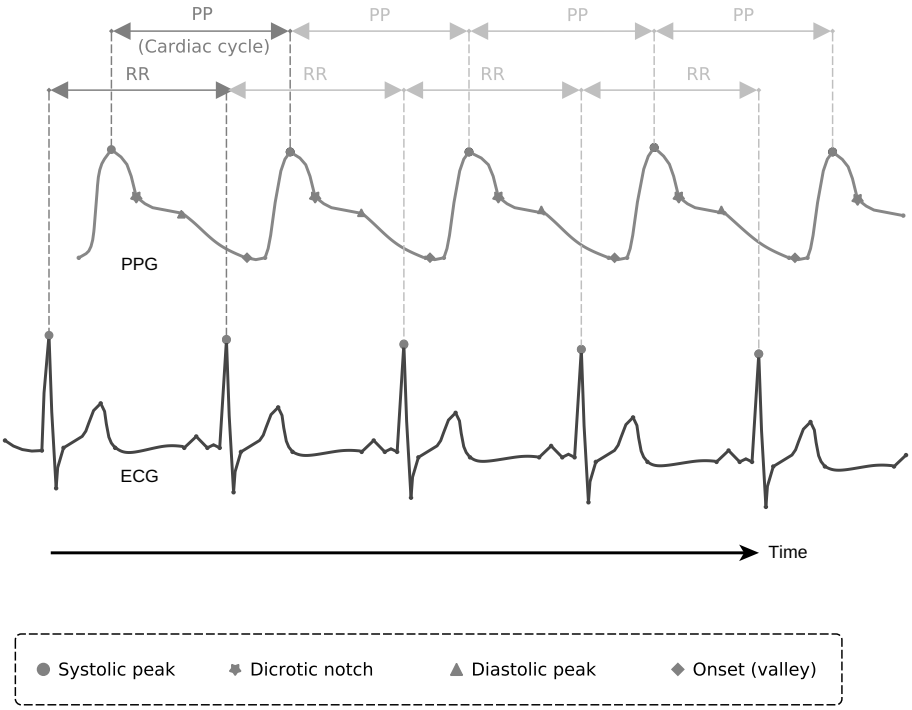


Figure 1.1: Inter-Beat Interval (IBI) estimation using RR Interval from Electrocardiogram (ECG) and the corresponding Peak-to-Peak Interval from Photoplethysmography (PPG).

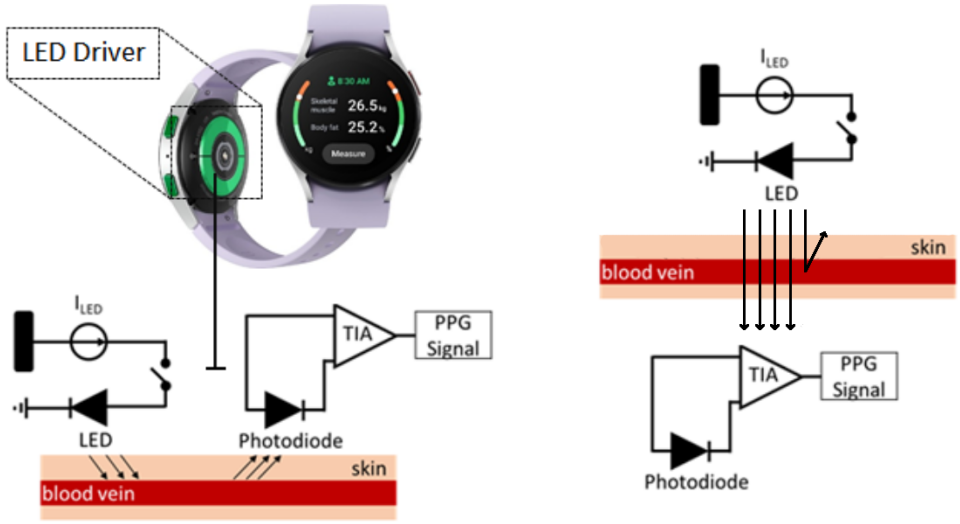


Figure 1.2: Schematic illustration of a PPG sensor. On the left, the signal is obtained through the reflectivity of light by human tissue, while on the right, it is obtained through the permeability of light through human tissue. This figure is a courtesy of Lucafó et al. [1]

Light-Emitting Diode (LED). The second component is a light receptor, which measures the light intensity. While a photodiode is commonly used for this purpose, cameras have also been employed. For signal extraction, these two types of devices are positioned to

exploit one of two light interaction principles. Figure 1.2 illustrates both principles. The first principle is light reflectance, where human tissue reflects incoming light rays. In this case, the devices should be positioned so that the tissue is not between them. An example of this application is smartwatches, where all devices are positioned below the main structure of the watch. The second principle is light permeability, where light can traverse the human tissue. Here, the devices should be placed such that the tissue is between them. An example of this setup is fingertip pulse oximeters. Once the setup is complete, the PPG signal can be extracted.

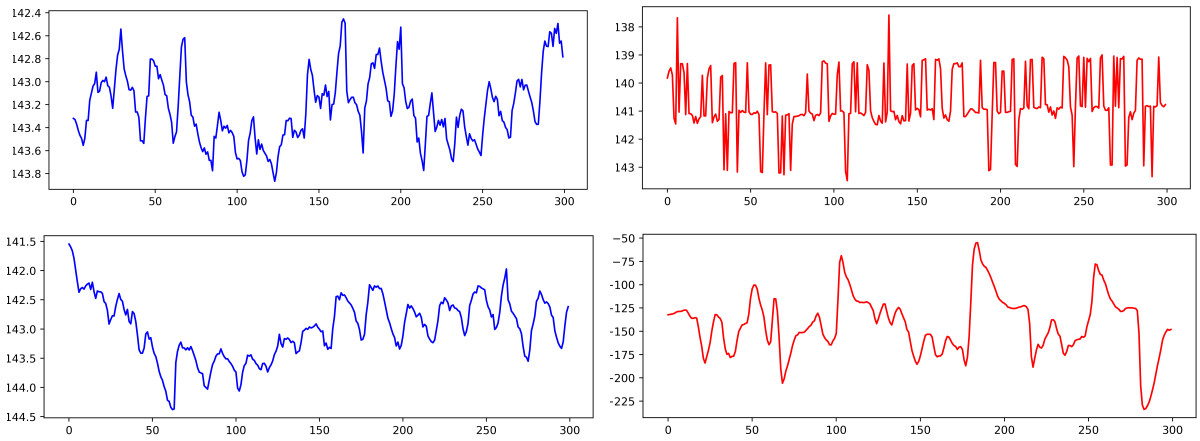


Figure 1.3: Example of 'good' (reliable) and 'bad' (unreliable) PPG signals. A 'good' signal (blue) exhibits symmetrical, well-defined, and more consistent patterns. In contrast, 'bad' signals (red) present irregular, asymmetrical patterns with reduced consistency and greater variability between periods. In these graphs, the vertical dimension is the average of all pixels intensity in a frame, while the horizontal dimension is the time instant in frames. These records are 10 seconds long and sampled with a frequency of 30 Hz. These signals were plotted using signals from BUTPPG [2] database.

Typically, PPG signals are highly prone to degradation due to various factors, especially motion artifacts. Excessive movement of the PPG sensor can cause significant distortion in the waveform, which affects the accuracy of subsequent signal analysis. These artifacts obscure or distort vital information within the signal. Figure 1.3 illustrates examples of both reliable and distorted PPG signals. Distorted signals can lead to incorrect decisions and misclassification, which is unacceptable in health and wellness applications. As a result, methods for assessing PPG signal quality are essential to prevent misinterpretation by differentiating between reliable and noisy data.

1.2 Contributions of This Work

The main contributions of this thesis are the development of a quality assessment framework for PPG signals. This framework is developed through several key contributions. Firstly, it proposes a new and effective approach for encoding time series into a set of 2D images. More specifically, the proposed method aggregates different projections into a composite hyperspectral image. This approach is based on the assumption that this aggregation provides better descriptiveness than using only a single projection. Additionally, this work evaluates the proposed approach in combination with a wide range of CV models, which previous works have not done. This evaluation provides insights into which types of models are ideal for the time series matrix embedding technique. Thirdly, the thesis explores a novel idea of transfer learning using a dataset outside the SQA domain, specifically the ImageNet dataset [5]. Finally, the thesis reports experiments conducted on a publicly available labeled dataset, named BUTPPG [2], which enhances the reproducibility and comparability of the experiments. The lack of reproducibility is a noticeable problem in the field currently, and the systematic empirical study conducted in this work can be useful for the community.

1.3 Organization of This Thesis

This Undergraduate thesis is organized in 5 chapters: this introduction; an overview of similar works existing in the literature; the proposed PPG signal quality assessment method; simulation, results and comparisons; and conclusions. Chapter 2 contains an overview of the SQA ecosystem and its quality assessment methods. Chapter 3 describes the contributions of this thesis, containing all the proposed quality assessment methods. Chapter 4 contains the experimental setup, simulation results and comparisons of the proposed SQA quality assessment methods and other state-of-the-art methods. Finally, Chapter 5 presents the conclusions of this work.

Chapter 2

Literature Review

This chapter is dedicated to exploring similar case studies similar and works in the area of SQA, especially for cardiological signals. SQA involves a wide range of problems in signal processing technologies, and it is not exclusive to the medical domain. Actually, its origins relate to the old communication systems, when researchers published their first works on the theory of information in the 1920s and 1930s. One example of such foundational works is the "Mathematical analysis of random noise", by Stephen O. Rice, in which he analyzes the statistical properties of communication device noises [6]. Another example is the work of Claude E. Shannon, "A mathematical theory of communication", which introduces fundamental concepts in communication systems [7]. In that millennium, studies already used the concept of SQA. We can see samples of this in the 1980s, such as the work of R. H. Stehle, which proposed an algorithm for assessing the quality of shortwave broadcast signals, trying to objectively measure the human perception of the signal message intelligibility [8]. Some of its conclusions are useful in the SQA in clinical contexts, such as the high degree of subjectivity in the human idea of quality. This makes labeling datasets properly fundamental to reflect this concept of quality in the proper evaluation of the developed SQA algorithms.

Past the 20th century, the SQA of physiological signals begin to become popular. Particularly, the early 2000s decade had several works in the subject. One of them is the work of Wang et al. [9], which proposed a ECG SQA method based on the difference between the areas of distinct QRS complexes. The work propose comparing the cumulative histograms of different ECG leads to assess its qualities. Later, Li et al. [10] suggested the combination of multiple quality indices and Heart Rate (HR) to assess ECG. Another work, by Deshmane et al. [11], posed the thresholding based on the Hjorth parameters [12] to assess the quality of PPG signals. Afterwards, Zhang et al. [13] elaborated a Arterial Blood Pressure (ABP) signal quality assessment metric based on the End Diastole Slope Sum and Slow Ejection Slope Sum features. Through those works, researchers proposed

quantifying the SQA in a metric. A common name that they used to refer to this metric was Signal Quality Indice (SQI).

2.1 Quality Assessment of Physiological Signals

Among the variety of physiological signals, the ECG is prevalent in the literature. This signal has multiple applications, such as disease classification, heartbeat type detection, biometric detection and emotion recognition [14]. There are a plenty of works in the SQA of ECG signals. In one of them, Naseri et al. [15] proposed two features for the estimation of a classification SQI of multi-channeled ECGs. One feature consists of verifying if two energy-like indices, measured in decibels, are within an admissible range. The other feature result from randomly choosing a target lead, feeding a Feed-Forward Neural Network (FFNN) with array of derivatives of all leads to reconstruct the targeted lead and finally comparing the original target lead to its artificial version with correlation analysis. In 2017, Orphanidou et al. [16] introduced a feature based on the extraction of the Heart Rate Variability of ECG signals. The method decomposes this new signal into wavelets with different frequency ranges and calculates the entropy of each of them, forming a feature vector. This vector feed a Suport Vector Machine (SVM), which classifies the signal as acceptable or not. Later, Shahriari et al. [17] developed an image-based feature that measures the Structural Similarity Measure between the input plot image, containing each signal channel Cartesian graph, and multiple template plot images of similar shape selected from the training dataset by using clustering analysis. One year later, Moeyersons et al. [18] proposed transforming the signal using the Auto Correlation Function and extracting simple features from it [18]. In the current year, Huerta et al. [19] generated phase space plots, such as Poincaré plots, and discretized them into a grid where each cell is the logarithm of the number of points contained in that cell. Thus, the SQA literature for ECG is already well developed.

Alternatively, the PPG has increased in popularity as an alternative to the ECG. In fact, its number of related articles published from 2013 to 2023 has increased in 176% [20]. This signal also elevated in presence in the SQA literature. A sample of this literature is the work of Li et al. [21], which poses the measurement of a SQI through the application of the Dynamic Time Warping (DTW) technique to measure the signal disparity to an established template. These authors, then, fed this SQI and some others to a Multi-Layer Perceptron (MLP) and to a self-made rule function, predicting a unique SQI. Experiments on private annotations on the MIMIC II dataset resulted on the MLP achieving the highest accuracy, 95.2%. Another sample is the work of Papini et al. [22], in which they proposed a method that, first, segments the input signal by finding all negative local minima points.

Then, the method creates a template signal by calculating the Dynamic Time Warping Barycenter Average of the signal segments. Finally, it calculates the SQI for each beat by comparing them to the template. This method obtained above 95% sensitivity and positive predictive values on two public datasets. Therefore, it is possible to achieve good predictive quality in the SQA of PPGs.

According to Such [23], SQA methods in biomedical signals generally fall into two categories: single-parameter and multiparameter approaches. Unlike single-parameter methods, multiparameter techniques utilize additional sensors that provide information related to the motion or the PPG itself. Examples of these additional sensors include accelerometers [24, 25] and optical source-detector pairs with peak responses beyond the red-infrared wavelength spectrum [26]. Some studies generate reference noise signals internally from the impaired PPG segments, reducing the need for extra hardware [27, 28]. Additional sensor channels can also transmit data about the same or a similar physiological indicator that reacts differently to artifacts. Nevertheless, the use of measured or synthetic reference signals to detect contaminated PPG segments often involves adaptive filtering, which, aside from its high computational and mathematical complexity, may require significant amounts of data and time to reach an optimal solution.

In contrast to the techniques mentioned earlier, SQA methods that do not rely on measured or synthetic reference signals (i.e., referenceless methods) may be better suited for wearable, real-time applications, since they eliminate the need for additional data collection and processing. In this context Machine Learning (ML) has made significant strides in this area by enabling the classification of PPG signals into 'reliable' or 'unreliable' based on the extraction of distinguishing information and the recognition of complex patterns, either automatically or with minimal human involvement [29].

One characteristic of the PPG SQA literature is the presence of both supervised and unsupervised machine learning approaches. Supervised learning involves learning features with the presence of labels. This allow the machine learning model to have access to more information in the learning process, but at the cost of requiring specialists to manually label the dataset. Most works in the SQA literature are supervised. Per example, Mohagheghian et al. [30] introduced a method to improve feature selection algorithms by ensembling feature subsets into a majority voting schema. A machine learning algorithm determines the voting threshold, such as AdaBoost, SVM, K-Nearest Neighbours (KNN) and discriminant analysis. Among those predictors, the AdaBoost presented the best performance in terms of Area Under Curve (AUC) Receiver Operating Characteristics (ROC) and accuracy. Furthemore, the method achieved accuracies of 91,55%, 92,29% and 95,86% on, respectively, the DeepBeat, UMMC Simband, and MIMICIII datasets. This result is competitive to other three compared methods, despite the labeled quality scores are

not publicly available. As another example, Tiwari et al. [31] proposed transforming the signal into a Modulation Spectrogram representation and, then, extracting features from it for, subsequently, feeding them to a machine learning model. Experiments compared the method to various SQIs by feeding them to a logistic regressor. Tests involved three wavelengths: red, green and infrared. The results showed that the method outperformed the others SQIs by 21.3% Balanced Accuracy (BACC) for green, 21.6% BACC for red, and 19.0% BACC for infrared wavelengths. A final example is the work of Miranda et al. [32], in which the SQA results from the application of a Interval Type-2 Fuzzy Logic system. Experiments on a private dataset lead to 77% and 93.72% Matthew's Correlation Coefficient (MCC) and Accuracy (ACC), respectively. However, one observation is that most of the experiments used originally unlabeled datasets, which make those results unreproducible.

On the other hand, unsupervised learning dismisses labels in the learning process. Even though this approach gives less information to the trained model, this makes the model independent to specialists guidance and its biases. As opposed to most works in the SQA literature, some works present unsupervised methods. For example, Singha et al. [33] propose extracting entropy and statistical features from the input signal and feeding them to a Self-Organizing Map, an unsupervised method. Experiments on a private dataset resulted in 92.01% accuracy in ternary classification. As another example, Mahmoudzadeh et al. [34] proposed extracting features from the time and frequency domains and feeding them to a Elliptical Envelope Algorithm, another unsupervised method. This method achieved 97% and 93% F1-Score on "Reliable" label for, respectively, intra and inter subject testing on a private dataset. Additionally, the combination of the supervised and unsupervised methods can produce semi-supervised methods, such as the method of Feli et al. [35], which feeds features to a Semi-Supervised One Class SVM, training it only on samples labeled as "Reliable". Comparing this semi-supervised approach to rule-based, unsupervised, supervised and deep learning methods lead to the proposed method surpassing all of them in terms of F1-Score for "Reliable" class with 99% value on a private dataset. However, likewise the supervised works, most datasets and its testing labeling are not public.

Another characteristic of the PPG SQA literature is the existence of the Cardiac Arrhythmia (CA) identification problem. The CA is the presence of an anomalous cardiac rate or rhythm without a physiological reason [36]. This medical condition is an obstacle in the design of SQI, since, in contrast to arrhythmic individuals, normal cardiac signals are periodic. Some features assume that the signal is periodic. This assumption can result in signals with arrhythmia being reject as unreliable signals, leaving those special patients misdiagnosed. Pereira et al. [37] conducted experiments on a private dataset that

contains cases of Atrial Fibrillation (AF), a type or arrhythmia. In this experiment, 40 features used in previous studies fed a SVM, which achieved an average accuracy superior to 94%. That accuracy was far higher than other existing methods, which the researchers also tested. Therefore, abundant feeding a machine learning algorithm with features and training it on datasets with arrhythmia cases already present an improvement in the detection of those special cases.

In sequence, two studies adopted a similar approach to the one mentioned in the past paragraph to attack the arrhythmia problem. In the first study, Pereira et al. [38] fed several features to three classifiers: SVM, KNN and Decision Tree. Similarly to the antecedent study, experiments on a private dataset demonstrated that the SVM was the best of all classifiers and it obtained above 95% surpassing the methods of other studies. The second study, by Pereira et al. [39], also included the SVM method, but added to the experiment deep learning models. The study contemplated both one-dimensional (1D) and bidimensional (2D) deep learning models, with the first receiving the raw signal and the second receiving its Cartesian plot image. Experiments on private data with presence of AF showed that the ResNet18 model was the best, with 98.5% accuracy. That last study highlighted that deep learning models have the potential to surpass conventional methods even with the presence of arrhythmic events. The next section further explores the use of deep learning in the literature.

2.2 Signal Quality Assessment Using Deep Learning

Methods based on Deep Learning (DL) has demonstrate the potential to achieve a higher accuracy when compared with feature-based models, even in the presence of CA. On one hand, differently of hand-crafted features, DL automatically extract features from the input signal, creating models that are adaptable to different dataset training contexts. Additionally, a high quality dataset can provide resources for the DL model to be robust to variations on the signal conditions. On the other hand, not only it creates a black-box that does not explain the reasons why the model attributed a certain SQI, but also requires large amounts of data to proper adjust the model parameters. Depite this, DL are worth exploring since it can provide the accuracy and robustness that the medical applications require.

In this context, several studies proposed the application of Convolutional Neural Network (CNN)s, one-dimensional DL models. For instance, Naeini et al. [40] designed a CNN to extract a binary SQI. Tests on a private dataset with data of three devices lead to 85% F1-score for the "Reliable" class. Alike, Zanelli et al. [41] employed a self-made CNN, but examined the effect of transfer learning as well. They conducted the experiment

on three private datasets. It starts training the model mostly on one dataset, in which it achieved an accuracy of 99.8%. Then, for each remaining database, they fine-tuned the the model with little training and tested on it. This procedure resulted on the 93% and 81% accuracies on the second and third datasets, respectively. Additionally, the model trained solely on the second database scored lower, 86%. Those results indicate that not only we can use CNN to achieve high accuracy but also we can transfer its learned features over different databases to improve its performance.

In a different light, some works go beyond the simple application of CNNs. The research of Lucafo et al. [1], per example, introduced a hybrid-model for quality assessment, which combines a CNN with a rule-based approach. This rule can bypass the utilization of the CNN by verifying if the signal min to max distance is less than an threshold. They determined the threshold by methods such as Last Value Thresholding and Nearest Value Thresholding. The researchers did it to avoid the unnecessary power demanded by the DL model. The method proved to be functional since it avoided the usage of the CNN for 3.27% of the input samples, while maintaining similar prediction scores if compared to the CNN without the rule component. Therefore, we can achieve particular advantages by combining the CNN with other methods.

Besides 1D CNNs, works explored the use of alternative DL models for the SQA of PPG. An example of this is recurrent networks, as we can see in the work of Gao et al. [42], in which they proposed the application of a long-short-term memory network for real time SQA, giving a SQI for each point in the signal. For the experiments, they labeled private and public datasets by applying Blind Source Separation to generate from each PPG signal one high-quality signal and one low-quality signal. When compared to baseline SQIs and existing models, it achieved competitive accuracy, while being lightweighted and enoughly fast to predict in real-time. Another example of alternative model is the combination of a Stack Denoising Auto-Encoder and a MLP, as we can see in the work of Singha et al. [43]. This model achieved 95% accuracy on a private dataset, better than baseline classifiers. A final example is the application of 2D CNNs through the projection of the signal into a image using Gramian Angular Fields (GAFs) by Naeini et al. [44]. Even though three 2D CNNs achieved above 90% accuracy, F1-Score, and AUC ROC scores on a private dataset, a proposed 1D CNN overperformed all of them. Thus, several approaches show results that compete with 1D CNNs. The usage of 2D CNNs, in particular, increased in interest in this last decade.

2.3 Methods Based on Time Series Imaging

Several works in the literature propose transforming the input signal into an image and, then, feeding it to a CV model. The figure 2.1 shows some of those transformations. The works that this paragraph mentions all used CNNs as a classifier of the SQI. One of those, Chen et al. [45], proposed the construction of a short-time Fourier Transform spectrogram as the signal transformation method. The researchers annotated the VitalDB database and, then, conducted experiments that lead to the proposed method achieving a better accuracy than four chosen baseline models. Its value was 98.3%. Another work, by Chatterjee et al. [46], transformed the signal into quantum pattern recognition images for PPG SQA. This resulted in 98.3% accuracy on the University of Queensland vital signs dataset with labels from the researchers, scoring above baseline models and an existing DL method. Roh et al. [47] embedded the signal into a Recurrence Plot matrix for PPG SQA. On a private dataset, the method achieved 97.5% accuracy. Based on these results, we notice that the application of time series image proved to be effective in the SQA literature.

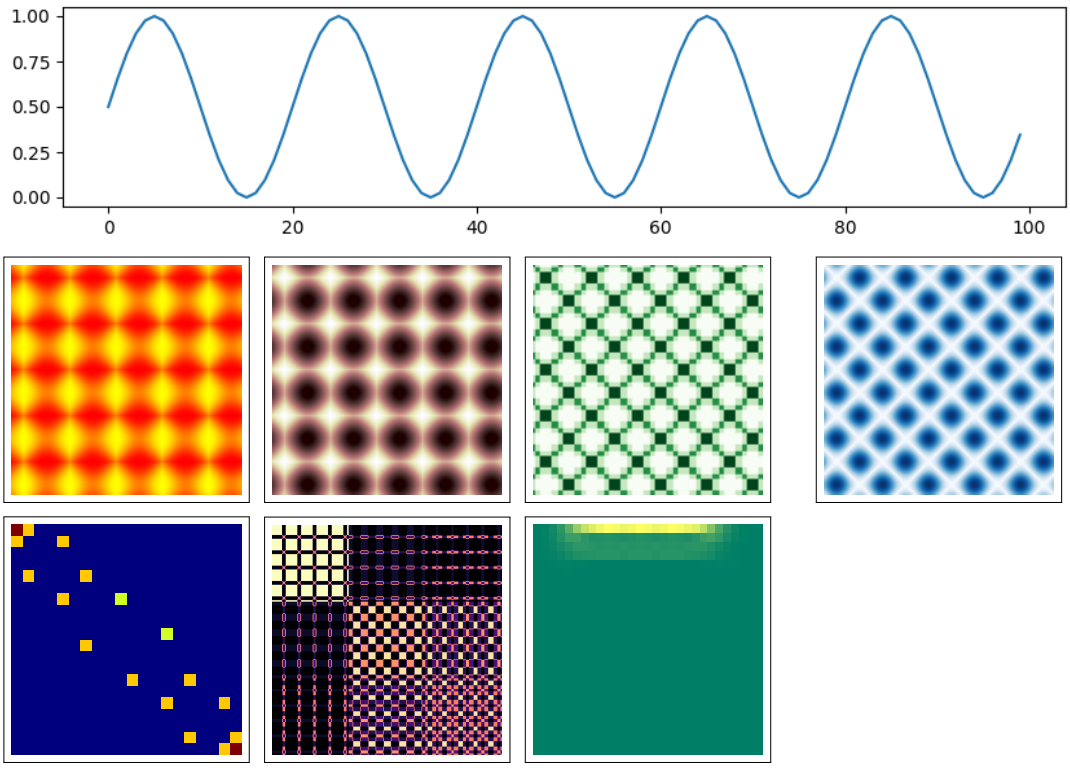


Figure 2.1: A signal and its various projection obtained by several methods. In the first line, from the left to the right, the methods are: Gramian Angular Diference Field, Gramian Angular Summation Field, Markov Transition Field and Recurrence Plot. The methods of the second line are, from the left to the right: Poincaré Plot Density Map, Multiscale Markov Transition Field and Short Time Fourier Transform Spectogram.

One particular method present in the SQA literature is the time series matrix embedding, which encodes time relationships of the original signal into a square matrix. For instance, Freitas et al. [48] fed a Vision Transformer with a Recurrence Plot or a Markov Transition Field, achieving, respectively, 89.9% and 90.3% accuracy on a private dataset. Freitas et al. [49] also fed images with a Vision Transformer, but used Gramian Angular Fields. The proposed approach reached 92.2% accuracy on a private dataset. Liu et al. [50] proposed the use of Multiscale Markov Transition Fields, version which concatenates the signal first and second derivatives. This projection fed a self-made CV model. Experiments with pre-training on the MIMIC-III and UCI databases, and fine-tuning and testing on the Queensland dataset resulted in 99,1% accuracy for binary classification. Thus, the combination of a generic CV model with a projection method, such as Recurrence Plot (RP), GAF or MTF, can achieve a decent accuracy, while it is possible to apply the multiscaling technique to improve some of those projections accuracy.

2.4 Final Considerations

Accurate identification of PPG sequences contaminated with artifacts is crucial for enabling reliable smart health applications, and ML techniques have brought outstanding progress in the field. Nevertheless, no previous study has provided an in-depth discussion of these methods. This chapter synthesized the current state-of-the-art approaches applying ML algorithms to assess PPG signals. Even though there are only a few datasets where signal data are labeled, supervised learning models are more frequent than their unsupervised counterparts, with SVM and CNN being the most widely. Although feature-engineered and deep learning methods demonstrate similar performance levels in some scenarios, deep learning may be more advantageous for addressing the limitations of manual feature engineering. In the current literature, there is also a need for more extensive exploration and validation across various body parts to confirm the reliability and adaptability of SQA methods for PPG signals. Additionally, a standardized series of experiments is required to test and validate these approaches. This thesis describes our efforts to fill these gaps by experimenting with various existing methods for time series, aiming to conduct a comprehensive study of the field.

Chapter 3

Methodology

This chapter first presents the overview of the method. Then, it describes the fundamental concepts about the matrix projection methods. Finally, it defines the new projection combination technique.

3.1 Overview of the Proposed Method

The Figure 3.1 depicts the proposed framework of this thesis. First, we transform the signal into four projections using the three before-mentioned algorithms: GAF (which generates two variants: GASF and GADF, MTF, and RP. Then, we aggregate those projections using composition, that is, we assign each of them to a different channel of a new input layer. Then, that layer feeds the computer vision model, which contained weights pre-trained on the ImageNet dataset. The Figure 3.11 pictures the feeding process to a three-channeled CV model, which consists in performing a point-wise convolution operation. Finally, that model classifies the signal into a binary SQI, which indicates if the signal is 'Good' or 'Bad'.

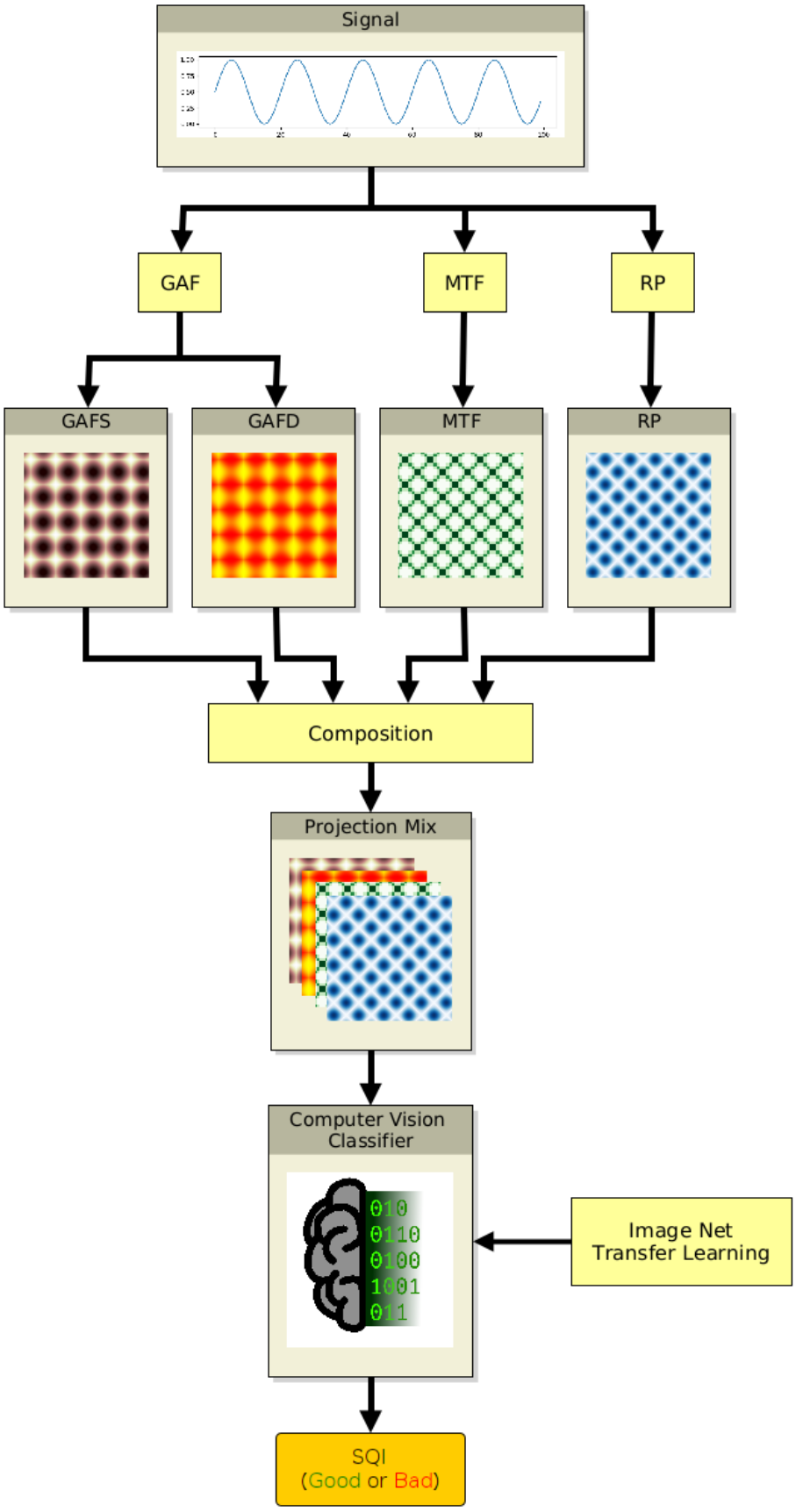


Figure 3.1: Diagram of the proposed PPG signal quality assessment framework.

3.2 Projection Methods

This section provides an analysis of the projections methods that this thesis used to convert the one-dimensional PPG signal into a 2D representation. For example purposes, we will use the signal in Figure 3.2. Additionally, this section also explores the influence of noise in the projections result.

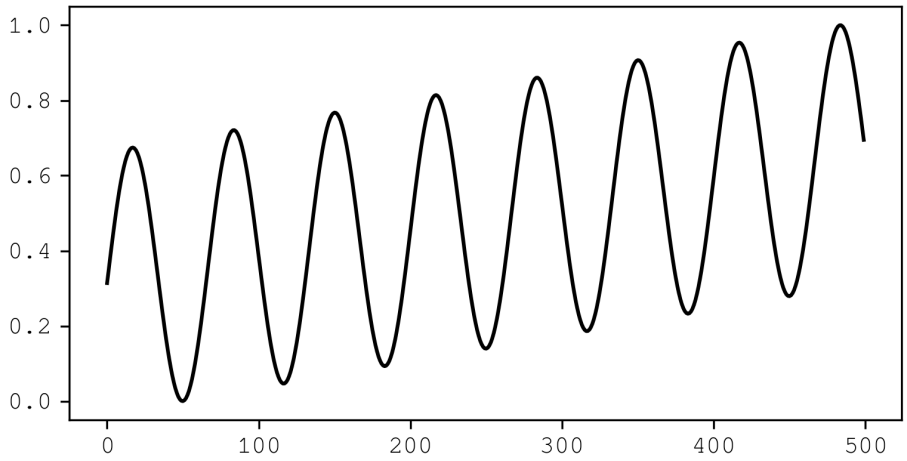


Figure 3.2: The example signal, with function $f(t) = sin(\frac{15 \cdot \pi \cdot t}{500}) + \frac{t}{500}$.

3.2.1 Recurrence Plot

The work of Eckmann et al. [51], of 1987, introduced the RP method as a tool for representing visually useful properties and behaviors of a time series. Since then, its application expanded to various domains, ranging areas such as earth sciences, finances, engineering, chemistry and physics [52]. It also is applicable in life sciences, with attemps on identifying the presence of Parkinson's disease [53], epileptic seizure [54], fetal hypoxia [55], and Alzheimer's disease [56]. Particularly, considering this thesis' scope, we employ RP in tasks involving cardiological signal processing. Thus, it is likely to be useful for SQA of cardiological signals.

The RP represents the occurrence of recurrences between the phase space values of time instant pairs. For this end, the first step is to embed the time series $X = \{x_1, x_2, ..., x_n\}$, with $x_i \in \mathbb{R}$ and $n \in \mathbb{N}$ samples, into a phase space, creating a new time series $S = \{\vec{s_1}, \vec{s_2}, ..., \vec{s_m}\}$, with $\vec{s_i} \in \mathbb{R}^d$ and $m \in \mathbb{N}$ elements. We can employ the time delays method to represent each element $\vec{s_i}$ of this new sequence S as follows:

$$\vec{s_i} = (x_i, x_{i+\tau}, x_{i+2 \cdot \tau} ..., x_{i+(d-1) \cdot \tau}) \tag{3.1}$$

where $d \in \mathbb{N}$ is the dimension and $\tau \in \mathbb{N}$ is the time delay of the phase space. Notice that the lenght m of the sequence S depends on both d and τ by the equation $m = n-(d-1)\cdot\tau$. Also notice that this embedding is optional, since the choice of the dimension $d = 1$ results in $S = X$, the original time series. The Figure 3.3 depicts the phase space of the example signal of the Figure 3.2.

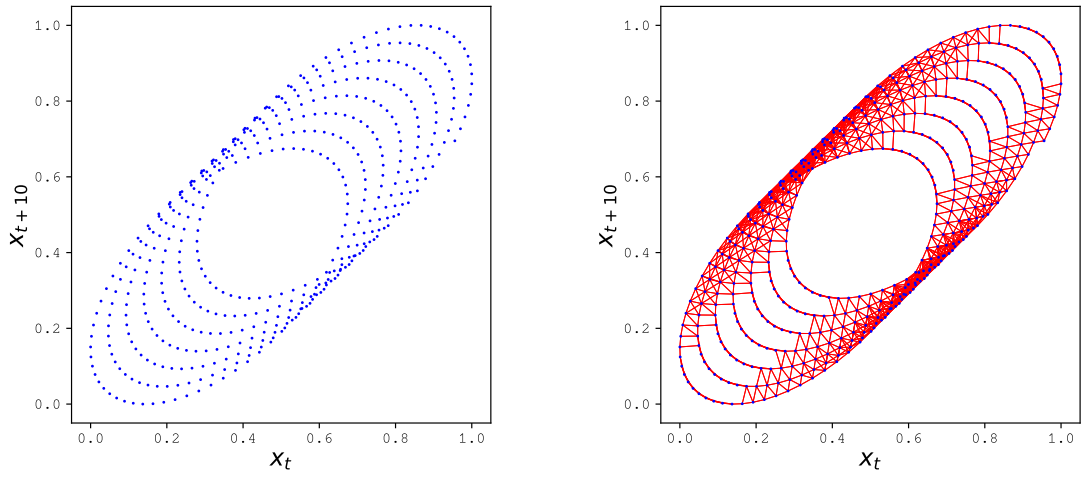


Figure 3.3: On the left, we have the signal of the figure 3.2 on the time delay phase space, without temporal information. Its parameters are dimension $d = 2$ and delay $\tau = 10$. On the right, we have almost the same figure, but with the recurrences represented by red lines $\vec{s_i} - \vec{s_j}$ that links the pair of near points that have a distance bellow $\varepsilon = 0.05$. That figure omits the recurrences to the point itself.

Then, the second step is to build a $m \times m$ matrix RP of recurrences where each cell $RP_{i,j} \in \{0, 1\}$ represents the presence or the ausence of a recurrence in a pair of points $\vec{s_i}, \vec{s_j}$ of the phase space S. We can represent this concept by measuring the distance $||\vec{s_i} - \vec{s_j}||$ between the points of the pair and verifying if it is smaller than a threshold $\varepsilon \in \mathbb{R}$, as the following equation:

$$RP_{i,j} = \mathcal{H}(\varepsilon - ||\vec{s_i} - \vec{s_j}||) \tag{3.2}$$

where $\mathcal{H} : \mathbb{R} \mapsto \{0, 1\}$ is the Heaviside function. Alternativelly, we can produce an unthresholded version $RP'_{m\times m}$ by attributing to each cell $RP'_{i,j} \in \mathbb{R}$ the points distance:

$$RP'_{i,j} = ||\vec{s_i} - \vec{s_j}|| \tag{3.3}$$

the Figure 3.4 exhibits both RP and RP' of the example signal.

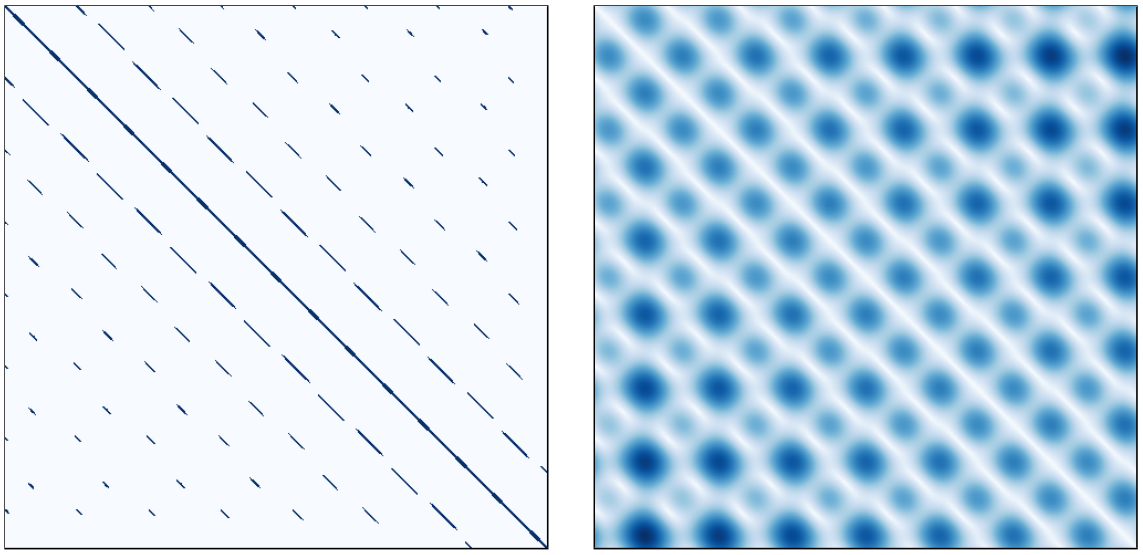


Figure 3.4: The resulting recurrence plots of the signal in figure 3.2, in coherence with the phase space of the figure 3.3. On the left, we have the thresholded version, while on the right we have the unthresholded version.

3.2.2 Gramian Angular Field

The work of Oates et al. [58] introduced the GAF method. This method, in summary, encodes the signal into angular relationships between pair of points. The first step to do this is to convert the signal $X = \{x_1, x_2, ..., x_n\}$, with $x_i \in \mathbb{R}$, $n \in \mathbb{N}$ samples, and time instants $\{t_1, t_2, ..., t_n\}$, into a polar coordinate series $P = \{p_1, p_2, ..., p_n\}$ with $p_i \in \mathbb{R}$. One manner to do that is to associate the time $i \in \mathbb{N}$ to the radius $r_i \in \mathbb{R}$ and the value $x_i \in \mathbb{R}$ to the angle by the inverse of the cosine as follows:

$$p_i(r_i, \phi_i) = f_{polar}(t_i, x_i) = \begin{cases} \phi_i = arccos(x_i), & -1 \leq x_i \leq 1 \\ r_i = \frac{t_i}{N}, & N \in \mathbb{R} \end{cases} \tag{3.4}$$

where N is a rescaling factor. Notice that it might be necessary to rescale the signal to fit each x_i in the range $[-1, 1]$. The Figure 3.5 shows the application of the function f_{polar} over the example signal. The polar coordinate system has one property of interest: the $f_{polar} : \mathbb{N} \times \{x \in \mathbb{R} | -1 \leq x \leq 1\} \mapsto \mathbb{R} \times \{\phi \in \mathbb{R} | 0 \leq \phi \leq \pi\}$ is bijective, since it has the inverse function $f^{-1}_{polar}(r_i, \phi_i) = (r_i \cdot N, cos(\phi_i)) = (t_i, x_i)$. This indicates that the application of the function f_{polar} does not result in loss of information.

The second step is to construct the temporal relationship matrix. We can achieve that by two methods that exploit trigonometric properties. One of them is calculating

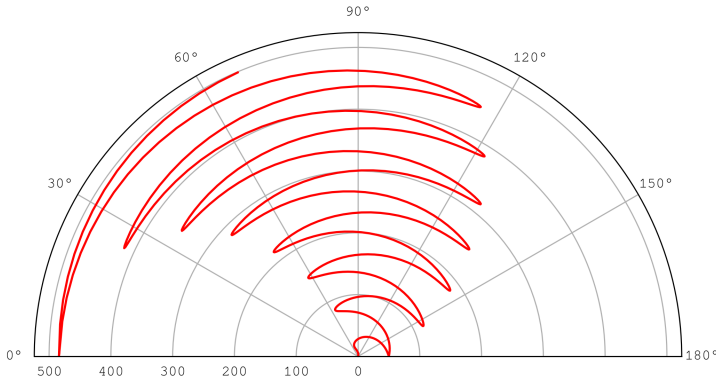


Figure 3.5: The example signal in its polar coordinate shape, with $N = 1$.

the cosine of the summation of the pairs of angles, constructing the matrix $GASF_{n\times n}$:

$$GASF_{i,j} = cos(\phi_i + \phi_j) \tag{3.5}$$
$$= cos(\phi_i) \cdot cos(\phi_j) - sin(\phi_i) \cdot sin(\phi_j) \tag{3.6}$$
$$= x_i \cdot x_j - \sqrt{1 - x_i^2} \cdot \sqrt{1 - x_j^2} \tag{3.7}$$

where GASF is the final matrix. Due to the inversibility of the *arccos* function, it is possible to express that calculation without trigonometric operations, as expressed by the equality 3.7. Thus, we can calculate GASF using matrix operations, as follows:

$$GASF = X^T \cdot X - \sqrt[\circ 2]{\mathbb{1} - X^{\circ 2}}^T \cdot \sqrt[\circ 2]{\mathbb{1} - X^{\circ 2}} \tag{3.8}$$

where $\square^{\circ 2}$ and $\sqrt[\circ 2]{\square}$ represents the element-wise square power and square root of the matrix $\square$, respectivelly, and $\mathbb{1}$ is a matrix in which all elements are 1. The other method is analogous, but uses the sine of the difference of the pairs of angles, constructing the following matrix $GADF_{n\times n}$:

$$GADF_{i,j} = sin(\phi_i - \phi_j) \tag{3.9}$$
$$= sin(\phi_i) \cdot cos\phi_j - cos(\phi_i) \cdot sin(\phi_j) \tag{3.10}$$
$$= \sqrt{1 - x_i^2} \cdot x_j - x_i \cdot \sqrt{1 - x_j^2} \tag{3.11}$$

Also similarly, by the equality 3.11, we can express GADF by matrix operations:

$$GADF = \sqrt[\circ 2]{\mathbb{1} - X^{\circ 2}}^T \cdot X - X^T \cdot \sqrt[\circ 2]{\mathbb{1} - X^{\circ 2}} \tag{3.12}$$

the Figure 3.6 shows the GASF and the GADF of the example signal.

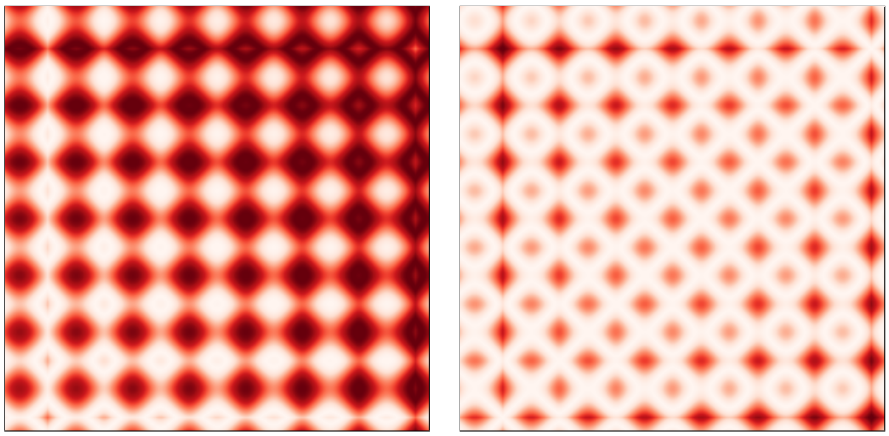


Figure 3.6: The example signal corresponding GADF (left) and GASF (right).

3.2.3 Markov Transition Field

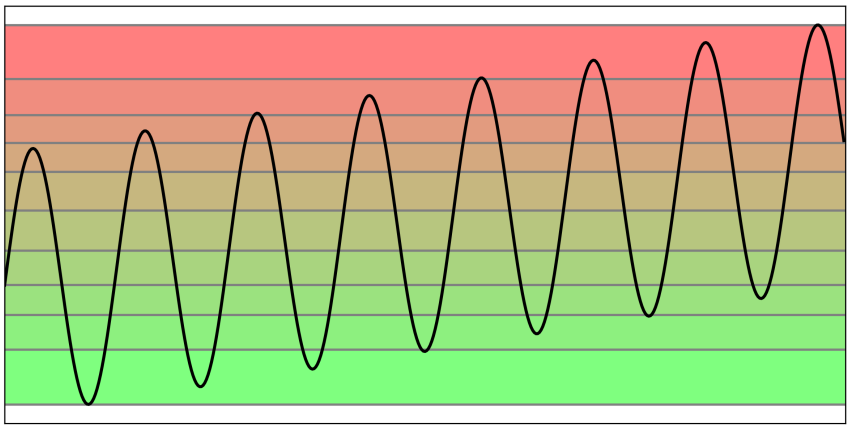


Figure 3.7: Original signal segmented into quantile bins.

Oates et al. [58] proposed the MTF as well, based on a signal to graph mapping of Campanharo et al. [59]. In fact, that mapping is the first step of this method. We map the signal $X = \{x_1, x_2, ..., x_n\}$, with $x_i \in \mathbb{R}$, to a graph $G = (N, W)$, with nodes set N and edges weights adjacency matrix W. Its nodes N corresponds to $m \in \mathbb{N}$ quantile bins $Q_i \subseteq \{x_i | i \in \{1, 2, ..., n\}\}$, that is, $|Q_1| = |Q_2| = ... = |Q_n|$ and, $\forall q_1 \in Q_1, \forall q_2 \in Q_2, ..., \forall q_n \in Q_n$, we have that $q_1 \leq q_2 \leq ... \leq q_n$. In other words, those quantiles bins separate the signal X into bands Q_i with equal ammount of samples x_i. The Figure 3.7 pictures this concept for the example signal. The other graph component, its edges, are directed and corresponds to the probability of a sample x_{k+1}, consecutive to a uniformly randomly chosen sample of a certain quantile $x_k \in Q_i$ (must have a consecutive), belonging

to a certain quantile Q_j. Those edges are akin to transitions of first-order Markov chains, since the probabilities summation of the transitions that sources from a state is always equal to 100%. We can express the adjacency matrix $W_{m\times m}$ as follows:

$$W_{i,j} = \frac{\sum\limits_{x_k \in Q_i, x_{k+1} \in X} f_{in}(x_{k+1}, Q_j)}{\sum\limits_{Q_l \in Q} \sum\limits_{x_k \in Q_i, x_{k+1} \in X} f_{in}(x_{k+1}, Q_l)}, \tag{3.13}$$

where

$$f_{in}(x, Q) = \begin{cases} 0, & x \notin Q \\ 1, & x \in Q \end{cases} \tag{3.14}$$

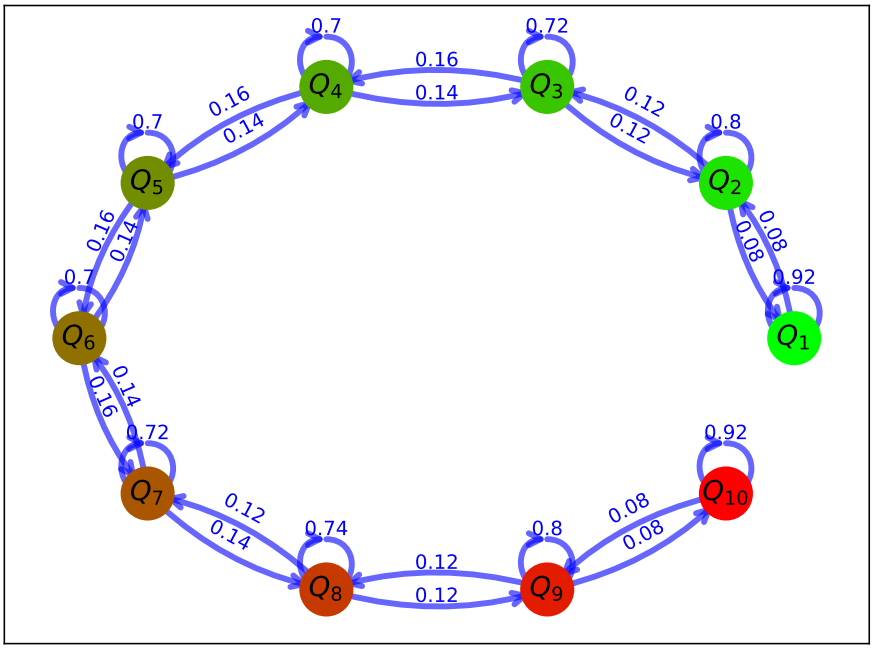


Figure 3.8: The graph of the Markov chain representing the transitions of the signal depicted in Figure 3.7.

The Figure 3.8 depicts the graph of the example signal. That graph G allows a probabilistic representation $X' = \{x'_1, x'_2, ..., x'_n\}$ of the input signal X by procedurally chosing a sample of the current quantile node and, then, transitioning to the next node according to the transitions probabilities. The algoritm 1 explicits that representation and the Figure 3.7 exemplifies it applied to the example signal. Therefore, the signal conversion to that graph representation is probabilistically reversible, meaning that it preserves statistical information of the signal, even though that means the sucessful recovery is not certain.

Algorithm 1 The probabilistic signal representation algorithm.

Require: Graph $G = (N = \{Q_1, Q_2, ..., Q_m\}, W)$
Ensure: Reconstructed Signal $X' = x'_1, x'_2, ..., x'_n$
  $Q_{current} \leftarrow Q \in_R N$
  for $k \in 1, 2, ..., n$ **do**
    $x'_k \leftarrow x \in_R Q_{current}$
    $Q_{current} \leftarrow Q_{next}$, with probability $W_{current,next}$
  end for

Since that graph does not retain temporal relationships, the second step of the MTF method is to build the matrix $MTF_{n \times n}$:

$$MTF_{i,j} = W_{u,v} | x_i \in Q_u, x_j \in Q_v \tag{3.15}$$

That is, each cell $MTF_{i,j}$ contains the transition probability between the quantiles Q_u, Q_v to which the samples x_i, x_j belong. The Figure 3.9 pictures the final result of the method applied to the example signal.

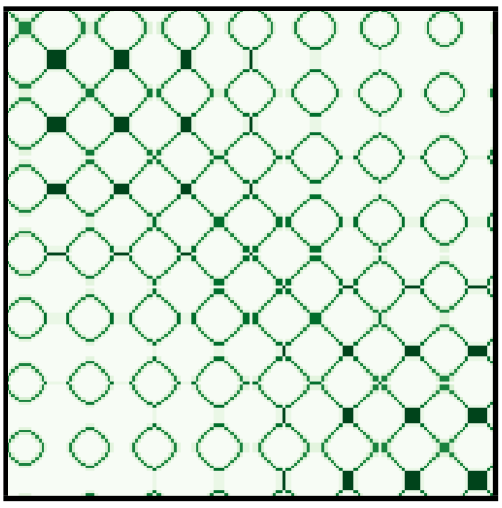


Figure 3.9: The MTF of the example signal, with the number of quantile bins $m = 8$.

3.2.4 Projections Comparison

The main idea of the projection methods is to reflect the signals properties and shape into visual patterns. That concept applies too to the inspection of the signal quality. The Figure 3.10 presents examples of the influence of different type of noises over the projections aspect. On observation is that the presence of low-frequency or high-frequency noises tend to produce, respectively, big or small scale structures. We can see in the effects of that observation in the figure, where the baseline wander figures manifest two to four big

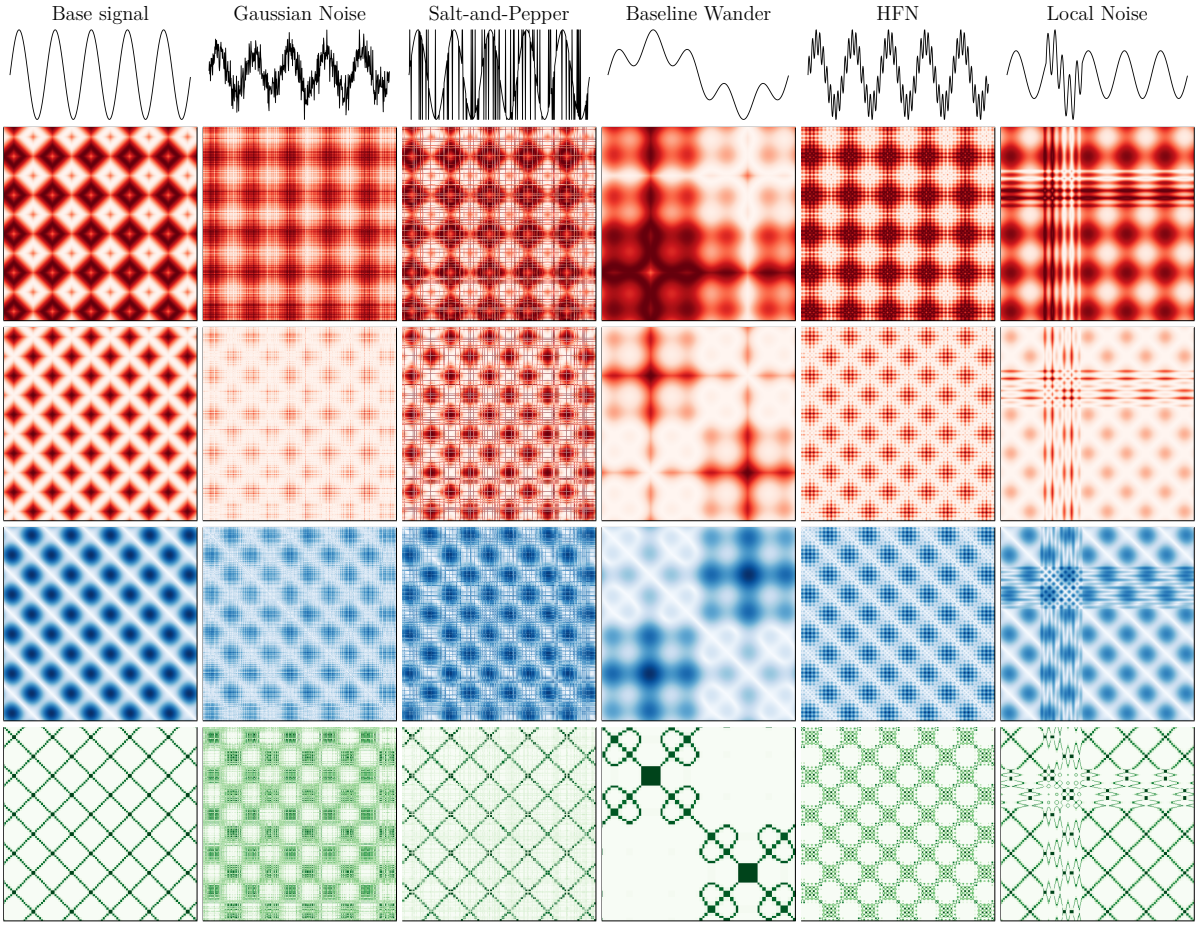


Figure 3.10: The signal, its impaired versions, and their corresponding 2D projections. From top to bottom: GADF, GASF, RP, and MTF.

structures and the high-frequency noise figures contain small dot-like structures. Another observation is that noises concentrated in a certain region tends to disturb the projection onto a cross-like structure, as we can see in the figure local noise column. We can see that property of "reflection" for usual noises, such as Gaussian, and salt and pepper noises. In the figure, the Gaussian noise still preserves the high-level structures and destroys low-level structures, while the salt and pepper noise produces vertical and horizontal lines with void or full colors in places corresponding to, respectively "salts" and "peppers" of the source signal. Therefore, those projections are likely to be useful in SQI tasks for CV approaches, since they present the noises visually while maintaining its characteristics.

3.3 Computer Vision Using Projection Mix

Since each projection has its unique characteristics, they are combined into an ensemble. For this end, all projections are fused to create an aggregated tensor. This tensor is

analogous to a hyperspectral image, i.e., unlike a typical color image, which consists of only three bands (red, green, and blue), the produced tensor provides additional spectral information for each pixel. The aggregation, which we call Projection Mix (PMix), consists in the assignment of each projection to a particular channel of a single input layer. Then, we "mix" that aggregation by performing a pointwise convolution operation as illustrated in Figure 3.11. Formally, the tensor $T_{in_{p\times n\times m}}$ used as input in the CV models is constructed by stacking the p projections $\{M_1, M_2, ..., M_p\}$, all with same dimensions $n \times m$, as defined as follows:

$$T_{in_{k,i,j}} = M_{k_{i,j}} \tag{3.16}$$

Then, the tensor T_{in} is 'mixed' into a new tensor $T_{mix_{q\times n\times m}}$ defined as:

$$T_{mix_{k,i,j}} = f_{actv}\left(\sum_{l=1}^{p} T_{in_{l,i,j}} \cdot w_{(k,i,j);l}\right) \tag{3.17}$$

where $w_{(k,i,j);l} \in \mathbb{R}$ is the weight applied to the link between the final tensor cell $T_{mix_{k,i,j}}$ and the input tensor cell $T_{in_{l,i,j}}$, while $f_{actv} : \mathbb{R} \mapsto \mathbb{R}$ is an activation function, such as ReLU, logistic sigmoid and Softmax. We can define the same tensor by directly assigning to the input projections:

$$T_{mix_{k,i,j}} = f_{actv}\left(\sum_{l=1}^{p} M_{l_{i,j}} \cdot w_{(k,i,j);l}\right) \tag{3.18}$$

We can observe that, for each cell with indexes i, j, the summatory $\sum_{l=1}^{p} M_{l_{i,j}} \cdot w_{(k,i,j);l}$ "mixes" the p projections by adding its values $M_{l_{i,j}}$, while attributing different degrees of contribution for each l-th projection depending on the weight $w_{(k,i,j);l}$. Then, we can use the f_{actv} function to mainly achieve binary distinguishability, leading to the final tensor value $T_{mix_{k,i,j}}$.

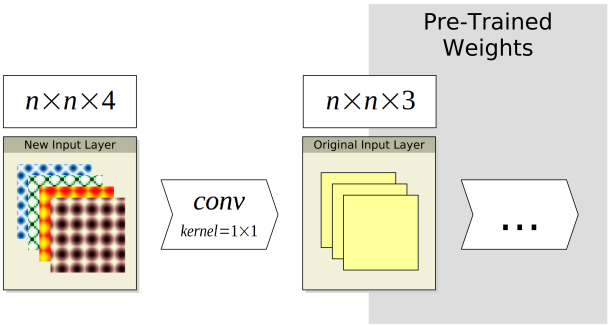


Figure 3.11: The three-channeled CV model feeding process. The figure begins in the left, in its input, and progresses to the right. It consisted in performing a 1×1 kernel size convolution operation. Notice that that may require resizing the projection composition.

Chapter 4

Experiments

In Chapter 3, we introduced the proposed technique for assessing PPG signals. In line with the goals outlined in Chapter 1, the aim of the current chapter is to investigate the effects of different methods on predicting whether a given input PPG signal is reliable or not. To achieve this aim, the chapter first presents the experimental setup, then discusses the experimental results, and finally addresses the limitations of the experiments.

4.1 Experimental Setup

This section discusses the experimental setup, analyzing the elements used in the experiments, such as datasets, programming libraries, and predictive models. Additionally, it clarifies the metrics to be evaluated and the approaches used to measure them.

4.1.1 The dataset

As with any machine learning task, we require a dataset to supply data for feeding the predictive models during parameter fitting, shaping them to the specific task's domain. In this work, assessing the quality of the signal is framed as a supervised classification problem, which can be described as the task of finding a function that best fits a predefined set of pairs of variables and labels, (X, y). In this context, the pair corresponds to the signal mapped to its quality label, either 'Good' or 'Bad'. To train the predictive methods and evaluate their performance in classifying the quality of heartbeat time series, the BUTPPG [2] dataset was employed.

The BUTPPG is a publicly available database produced by the Department of Biomedical Engineering at Brno University of Technology. It contains samples of PPG signals, their quality labels, and heart rate estimations. These signals were extracted using a low-cost method: recording with a smartphone camera. Specifically, the researchers recorded

the subject's index finger while it covered the camera lens and its LED light. For each video frame, they measured the average intensity of the red channel across all image pixels, resulting in a time series of averages. Finally, the signal was inverted.

They performed this method of obtaining PPG signals 48 times, with the samples distributed equally among 12 subjects—4 measurements per subject. Moreover, the recordings were taken in two situations: one where the subject was seated and remained static, a case in which the quality label 'Good' was likely; and another where the subject was walking, a case likely to result in a 'Bad' recording. This distinction is relevant because the walking condition occurred only once for each subject, resulting in approximately 25% of the recordings being labeled as 'Bad'. Therefore, this dataset is imbalanced, a factor that was addressed in our experiment.

For the definition of signal quality labels, specialists were designated to estimate the heart rate associated with the PPG signals using specialized software developed by the researchers. The organizers then compared the specialists' estimates with those provided by a gold standard method, which used an ECG recording as a reference instead of the PPG signal. The ECG was manually synchronized by the measurer. If the specialist's measurement had an error of 5 bpm or less, the estimate was considered correct. Finally, if 3 out of 5 specialists provided correct estimates, the PPG signal quality was labeled as 'Good.' Thus, the 'Good' labels in the dataset essentially indicate whether a signal is human-readable.

4.1.2 The dataset split

Machine learning tasks also require the dataset to be split into fragments. One of these is the training dataset, used for fitting the model's parameters. Another is the test dataset, used for evaluating the model's efficiency. An optional split is the validation dataset, used to select the best set of hyperparameters for the trained model. In our experiment, the training-test splits were defined using a cross-validation method called Leave One Subject Out (LOSO), which partitions the dataset into K train-test splits. The k-th train-test split assigns the k-th segment as the test dataset, leaving the remaining $K-1$ segments as the training dataset. In the case of BUTPPG, K equals 12, which corresponds to the number of subjects. Notably, the smallest unit of division is the subject, not the individual signals associated with each subject. This approach has the advantage of increasing the distinction between training and testing samples. Since the dataset is small, this splitting method maximizes the use of available resources by ensuring every sample is used as a test case at least once, without introducing bias into the results. Additionally, the training dataset was further divided to produce a validation dataset of size 3, the usage of which will be explained later.

4.1.3 The models

To evaluate the proposed projection-based framework and identify specific cases of superior performance, it was necessary to involve a large number of machine learning models. Firstly, this work compared the projection-based framework with other time series classification approaches using the Aeon-toolkit Python library [60], with the models listed in Table 4.1. Furthermore, the proposed method was combined with a wide variety of classification CV models, utilizing the PyTorch Python library [61], which supports a diverse range of neural network architectures. These architectures vary from simple convolutional networks to vision transformers. Table 4.2 lists all the CV models involved in the experiment. This approach subjected the framework to an extensive set of experimental scenarios.

However, defining the models alone is insufficient, as the selection of their hyperparameters is also required. Hyperparameters are high-level parameters that do not change during the training process. For the Aeon models, the default hyperparameters provided by the library were used. For the computational vision models, while most hyperparameters were set to their defaults, our experiments employed hyperparameter search for the learning rate, used by the optimization algorithm. The Optuna Python library [112] conducted this search by heuristically exploring the parameter space dynamically defined in the user code. Optuna prunes the search-space tree using various methods, and in our experiments, the median pruning method was applied. In this case, the guiding metric for the heuristic search was the accuracy score, defined as the ratio of correct predictions to the total number of samples. This score was measured on a validation dataset of size 2, created through a simple random split. This functionality allowed us to find a near-optimal combination of parameters without exhaustively testing all possible cases, using the model's validation dataset score as a heuristic.

4.1.4 Training strategy

Given the aforementioned models, dataset, and its divisions, it was necessary to establish the training method for adjusting the models' parameters. Since the Aeon implementation already contained a default training procedure, our experiment only established the fitting framework for the PyTorch computer vision models and the data feeding method. Our experiment involved feeding the models by loading the signals data, applying random oversampling before transforming them, as the dataset was unbalanced. After performing the projection transform, our experiment loaded pre-trained model weights provided by PyTorch, which were originally trained on the ImageNet [5] dataset. Following that, we fitted the PyTorch models using the Adam optimization algorithm [113] to minimize the

Table 4.1: Non-Computer Vision models list, containing its references.

Classification	Model	Reference
Convolution-Based	Arsenal	[62]
	Rocket Classifier	[63]
Deep Learning	Zhao's CNN Classifier	[64]
	Wang's FCN Classifier	[65]
	Wang's MLP Classifier	[65]
	Inception Time Classifier	[66][67]
	Individual Inception Classifier	[66][67]
	LITE Time Classifier	[68]
Dictionary-Based	BOSS Ensemble	[69]
	Contractable BOSS	[70]
	Individual BOSS	[69]
	Individual TDE	[71]
	MUSE	[72]
	TemporalDictionaryEnsemble	[71]
	WEASEL	[73]
	WEASEL V2	[74]
	REDCOMETS	[75][76]
Distance-Based	Elastic Ensemble	[77]
	K-Neighbors Time Series Classifier	—
	Shape DTW	[78]
Feature-Based	Catch-22 Classifier	[79]
	Summary Classifier	—
	TS Fresh Classifier	[80]
Hybrid	HIVE-COTE V1	[81][62]
	HIVE-COTE V1	[62]
Inverval-Based	Canonical Interval Forest Classifier	[82]
	DrCIFClassifier	[62]
	Random Interval Spectral Ensemble Classifier	[83]
	Supervised Time Series Forest	[84]
	Time Series Forest Classifier	[85]
	Random Interval Classifier	—
Shapelet-Based	Shapelet Transform Classifier	[86][87]
	RDST Classifier	[88][89]
Ordinal Classification	Individual Ordinal TDE	[90]
	Ordinal TDE	[90]
Other	Continuous Interval Tree	[91]
	Rotation Forest Classifier	[92]

Table 4.2: Computer Vision models list, containing its citations.

Classification	Model	Reference
Transformer	Vision Transformer	[93]
	MaxViT	[94]
	Swin Transformer	[95]
	Swin Transformer V2	[96]
Residual Net	ResNet	[97]
	ResNeXt	[98]
	WideResNeXt	[99]
Extreme Net	DenseNet	[100]
	VGG	[101]
	SqueezeNet	[102]
Mobile-Oriented	MNASNet	[103]
	MobileNet V2	[104]
	MobileNet V3	[105]
Efficiency-Oriented	EfficientNet	[106]
	EfficientNet V2	[107]
	ShuffleNet V2	[108]
Diverse	AlexNet	[109]
	ConvNeXt	[110]
	RegNet	[111]

cross-entropy loss function. The implementation of the training strategy performed this optimization cycle with a number of epochs determined by a median-based early stopping technique. The formula below gives the score of the n-th epoch:

$$EarlyStopScore(n) = med([|l_{n-i} - med([l_{n-i}]_{i=0}^{9})|]_{i=0}^{9}) \tag{4.1}$$

Where l_k is the loss value (i.e., cross-entropy loss) of the k-th epoch, med is the median and $[f(i)]_{i=0}^{k}$ is the sequence generated by $f(i)$ when varying i from 0 to k. In other words, the formula calculates the median of the absolute deviations of the medians of the last 10 loss values using the central value. If $EarlyStopScore(n) \leq 0.1$, the training stops in the n-th epoch. With that established, it remains to determine the metrics to be measured" for smoother readability.

4.1.5 Performance measurements

We chose the metrics to evaluate the efficacy of the solution. For these experiments, we can categorize the metrics into two groups: prediction metrics, which measure the quality of the model's signal quality assessments, and benchmarking metrics, which measure resource usage and the model speed. As the prediction metrics, our experiments used the

Cohen kappa score, the F1-score, and the precision. The Cohen kappa score, in binary classification tasks, measures the agreement between the obtained accuracy acc_o and the expected accuracy acc_e. The following equations define those accuracies and the Cohen kappa score, in terms of confusion matrix cell values:

$$acc_o(R) = \frac{TP + TN}{N}, \tag{4.2}$$

$$acc_e(R) = \left(\frac{TP + FP}{N} \cdot \frac{TP + FN}{N}\right) + \left(\frac{TN + FP}{N} \cdot \frac{TN + FN}{N}\right), \tag{4.3}$$

and

$$\textit{Cohen-kappa}(R) = \frac{acc_o(R) - acc_e(R)}{1 - acc_e(R)} \tag{4.4}$$

Where $N = TP+TN+FP+FN$ is the total number of samples and R is the set of pairs $\{(f(X), y') | \forall (X, y') \in \textit{Dataset}\}$, where f is the predictor. For the purpose of aligning this metric with others, we can rescale that metric from $[-1, 1]$ to $[0, 1]$:

$$\textit{Cohen-kappa-Rescaled}(R) = \frac{\textit{Cohen-kappa}(R) + 1}{2} \tag{4.5}$$

In sequence, the Precision is a metric that measures the ratio of hits in the set of positive predictions. In our context, a higher Precision imply that the predictor approved a low amount of "Bad" signals, which is desirable in applications where we do not want to show to the user measurements based on unreliable signals. From the Precision and from the Recall, the ratio of hits in the set of all existing positives, we can obtain the F1-Score. Precisely, the F1-Score is the harmonic mean between those two metrics. In other words, a high F1-Score indicates a good balance between Precision and Recall scores. In our application, it measures the same as the Precision plus the Recall, which would measure the amount of "Good" signals that would feed the application. This is an desirable quality when we want to provide constant feedback to the user. The following equations define those metrics:

$$\textit{Precision} = \frac{TP}{TP + FP}, \tag{4.6}$$

$$\textit{Recall} = \frac{TP}{TP + FN}, \tag{4.7}$$

and

$$\textit{F1-Score} = \textit{Harmonic-Mean}(\textit{Precision}, \textit{Recall}) = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}. \tag{4.8}$$

Therefore, the Cohen kappa score provides an overall sense of accuracy, the F1-Score suggests the model's usability level, and precision indicates the predictor's reliability.

Regarding the benchmarking metrics, our experiment measured the memory usage of the model in bytes and the inference time (including the 1D-to-2D projection time for projection-based models) in seconds. Memory usage is crucial because practical applications for heart rate estimation often impose hardware constraints that limit allowable memory usage. Additionally, inference time is important for achieving near-instantaneous evaluations, which enhances the application's responsiveness.

4.1.6 Overall schema

The Figure 4.1 illustrates the experiment framework for the CV models. One notable difference from the framework used for non-CV models is that the 1D-to-2D conversion acts as a boundary between the dataset and the other components. This means that the experiment for non-CV models can be represented using a similar schema by omitting the conversion block. The experiment began with hyperparameter selection, involving the splitting of the BUTPPG dataset through a simple division method to subsequently select the optimal hyperparameters for the CV models. With the best hyperparameters chosen, all models, including non-CV models, will be evaluated using the LOSO strategy. For each fold, our experiments subjected the model to a training procedure that iterates through epochs of training and validation until early stopping is triggered. The model is then tested to produce the metrics for that fold.

4.1.7 The implementation details

The dataset sourcing procedure was carried out using the PyTorch multithreading data feeding solution, Data Loader[1]. This was configured to load batches of size 32. Prior to loading the batches, the training dataset was balanced using the Imbalanced Learn library [114] and its random oversampling method[2]. The batches were then transformed from 1D signals into 2D images using the projection algorithms of the PyTS library [115]. Although the signals are now 2D, their width and height might not be compatible with the original network's input dimensions, especially considering pre-trained weights. To address this issue, the PyTorch resize transform[3] was applied to adjust the width and height. Additionally, a new convolutional layer corresponding to the PMix method was incorporated.

[1]Documentation available at `https://pytorch.org/docs/stable/data.html#torch.utils.data.DataLoader`

[2]Documentation available at `https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.RandomOverSampler.html#imblearn.over_sampling.RandomOverSampler`

[3]Documentation available at `https://pytorch.org/vision/stable/generated/torchvision.transforms.Resize.html`

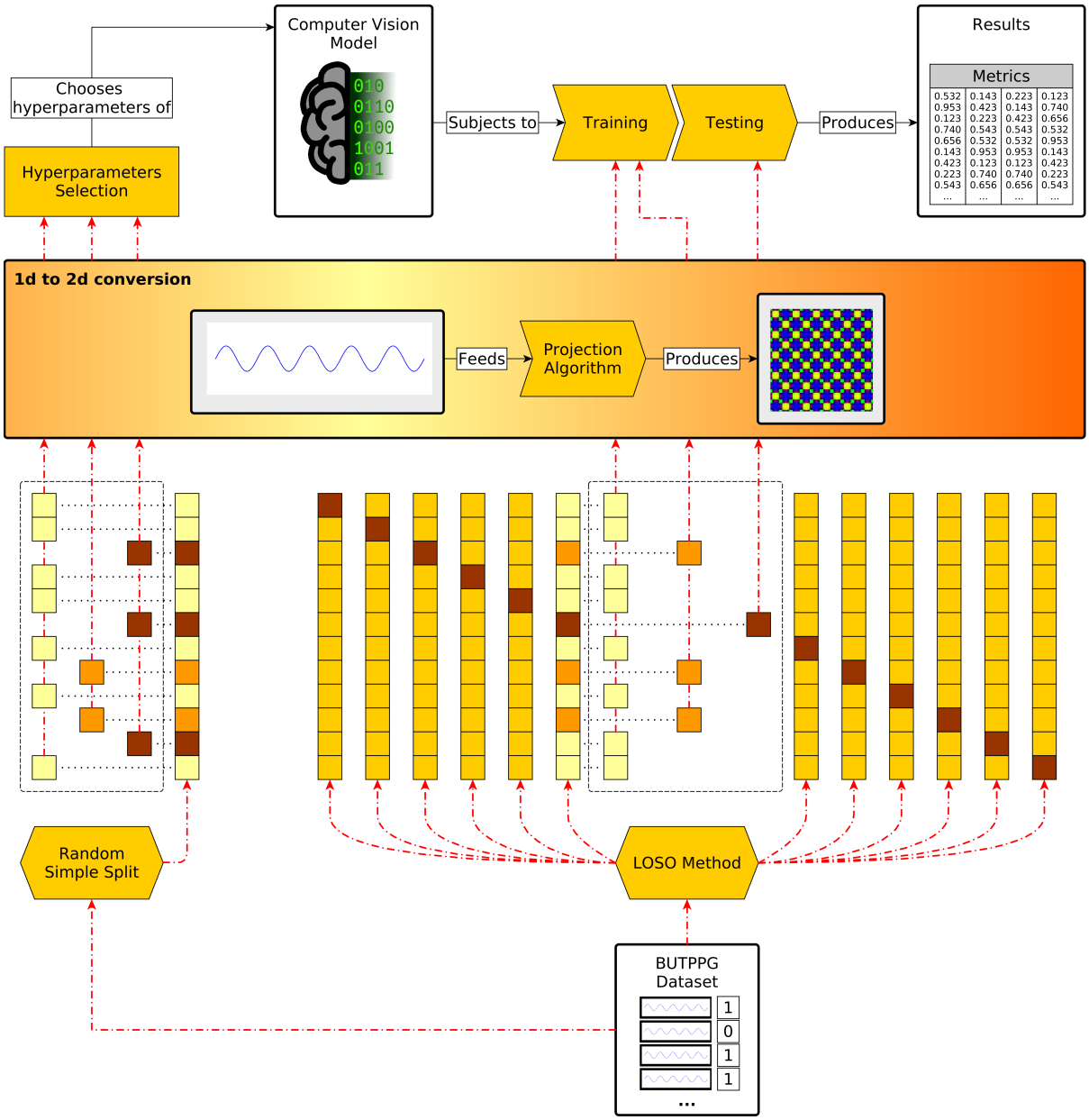


Figure 4.1: The framework of the experiment. Red dotted arrows indicates data flow that sources from the BUTPPG Dataset, while the full black lines, labeled with an verb, represent a relationship "A do B", where the arrow starts on A and end on B. Notice that the figure presents the training-testing cycle for only one of the twelve folds. The above schema is replicated for each combination of Computer Vision Model and Projection Algorithm.

The CV models were trained using a single NVIDIA RTX 3090 TI. For training, our implementation used the Pytorch implementation of the Adam optimization algorithm[4], which uses the gradients evaluated by the Pytorch autograd engine [116]. The loss class (which, in our case, is the torch.nn.CrossEntropyLoss[5]) backpropagates the gradients based on the model forward pass errors. For the models testing, our implementation used the Sklearn [117] metrics[6]. For model memory measurement, our implementation counted the summation of the size of each parameter and buffer tensors in the CV models, while for the non-CV models, our implementation used the `asizeof` function[7] of the Pympler library [118]. Finally, we describe the inference time measurement, for which our implementation extracted 500 measurement samples. For the non-CV models, our implementation used the Python time method[8], of the time library, to measure instants of time. For the CV models, our implementation marked the time instants by using CUDA events interface provided by Pytorch[9], while, before measuring, performing 500 iterations to warm-up the GPU.

4.2 Experimental results

The results were analyzed by comparing the score metrics of the models and assessing their trade-offs with respect to memory consumption and inference time. Given the large number of models considered, the analysis was organized into sections. First, each section focused on one of the CV model families listed in Table 4.2: Transformers, Residual Nets, Mobile-Oriented, Extreme Nets, Efficiency-Oriented, and Diverse. Within each category, the analysis identified the best combinations of model variants and projection methods. Subsequently, the top non-CV models from the Aeon toolkit library were selected. Finally, the overall best choices were determined, and differences between the projection methods were discussed.

4.2.1 Analysis by Computer Vision Model Family

This analysis covers each CV model family listed in Table 4.2.

[4]Documentation available at `https://pytorch.org/docs/stable/generated/torch.optim.Adam.html#torch.optim.Adam`

[5]Documentation available at `https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html#torch.nn.CrossEntropyLoss`

[6]Documentation at `https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics`

[7]Documentation at `https://pympler.readthedocs.io/en/latest/library/asizeof.html`

[8]Documentation at `https://docs.python.org/3/library/time.html#time.time`

[9]Documentation at `https://pytorch.org/docs/stable/generated/torch.cuda.Event.html`

Transformers

The experiments tested four transformers: Vision Transformer (ViT), Multi-axis Vision Transformer (MaxViT), Shifted Windows Transformer (SwinT), and its second version, Shifted Windows Transformer Version 2 (SwinTV2). The ViT serves as the base model for the others, transforming visual input into a sequence where each element is a linear embedding of subimage patches obtained by partitioning the original image into a grid-like pattern. Subsequent models build on this base by incorporating additional layers and altering attention mechanisms. For example, the MaxViT utilizes architectural blocks that alternate between two self-attention modes: grid attention, which operates with high granularity, and block attention, which operates with low granularity. The SwinT modifies attention at both the layer level—by merging patches from the previous layer—and at the block level—by shifting self-attention windows to different positions. The SwinTV2 introduces several specific improvements over the earlier version.

One metric table was generated for each type of transformer. The Table 4.3 presents the ViT scores, with variants categorized as Base (B), Large (L), or Huge (H) in parameter size and patch sizes of 14, 16, or 32. The table shows that the PMix and RP projection methods achieved the best scores across all metrics. In most cases, PMix was equal to or better than RP, except for the H 14 variant, where RP was superior. Among the combinations of variants and projections, RP and PMix with B 16 and L 16, as well as PMix with B 32 and L 32, yielded the best scores. Table 4.4 shows the MaxViT scores. For this model, the PMix method achieved the highest scores for the Cohen Kappa and Precision metrics, while the RP surpassed it for the F1-Score, despite PMix having the smallest dispersion for that metric. Table 4.5 exhibits the SwinT scores, with variants categorized as Base (B), Small (S), or Tiny (T) based on parameter count. The RP method achieved the best scores for the B and S variants, while the PMix method resulted in better scores for the T variant. Specifically, the PMix method with the T variant attained the highest Cohen Kappa and Precision scores, but ranked second for the F1-Score, which was surpassed by the RP method with the S variant.

Table 4.6 displays the SwinTV2 scores, with variants categorized as Base (B), Small (S), or Tiny (T). The PMix method achieved better scores for the B and S variants, while the RP method performed better for the T variant, despite RP having the largest dispersion for the F1-Score in this case. Specifically, the PMix method with the S variant resulted in the highest Cohen Kappa and F1 scores, and the second-best Precision, where the RP method with the T variant was superior. When considering all tables 4.3, 4.4, 4.5, and 4.6, the RP and PMix methods with ViT B 16 and ViT L 16, as well as PMix with ViT B 32 and ViT L 32, and the SwinTV2 S with PMix, generally achieved better scores. The benchmarking metrics for these combinations are summarized next.

Table 4.3: Averages and standard deviations of the folds evaluation for the VisionTransformer variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
Vision Transformer: B 16	GAF	0.562 ± 0.155	0.833 ± 0.090	0.771 ± 0.167
	MTF	0.518 ± 0.040	0.771 ± 0.163	0.773 ± 0.175
	RP	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
Vision Transformer: B 32	GAF	0.583 ± 0.163	0.844 ± **0.074**	0.785 ± 0.148
	MTF	0.515 ± 0.207	0.790 ± 0.167	0.729 ± 0.155
	RP	0.883 ± 0.184	0.913 ± 0.154	**0.944** ± 0.130
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
Vision Transformer: H 14	GAF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± **0.129**
	MTF	0.477 ± 0.075	0.799 ± 0.154	0.708 ± 0.144
	RP	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
	PMix	0.833 ± 0.222	0.931 ± 0.087	0.903 ± 0.146
Vision Transformer: L 16	GAF	0.667 ± 0.246	0.873 ± 0.117	0.812 ± 0.188
	MTF	0.594 ± 0.254	0.851 ± 0.118	0.736 ± 0.284
	RP	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
Vision Transformer: L 32	GAF	0.674 ± 0.257	0.868 ± 0.119	0.819 ± 0.173
	MTF	0.612 ± 0.196	0.828 ± 0.141	0.811 ± 0.167
	RP	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130

Table 4.4: Averages and standard deviations of the folds evaluation for the Maxvit variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
MaxViT	GAF	0.653 ± 0.261	0.857 ± 0.132	0.806 ± 0.192
	MTF	0.544 ± 0.190	0.706 ± 0.186	0.788 ± 0.222
	RP	0.854 ± 0.225	**0.932** ± 0.111	0.910 ± 0.172
	PMix	**0.875** ± **0.169**	0.921 ± **0.098**	**0.944** ± **0.130**

Table 4.7 shows that the SwinT V2 S variant uses considerably less memory than the ViT variants. Therefore, the SwinT V2 S with PMix can achieve high scores while utilizing less memory. However, Figure 4.2 indicates that the SwinT V2 S variant has a slower inference speed compared to the ViT variants. Among the ViT variants, the B 32 variant was the fastest, suggesting that the combination of PMix with ViT B 32 can produce high scores with lower inference time.

Table 4.5: Averages and standard deviations of the folds evaluation for the SwinTransformer variants rounded to 3 decimal places, in the format *mean* ± *std*. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
Swin Transformer: B	GAF	0.625 ± 0.226	0.861 ± 0.110	0.792 ± 0.179
	MTF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± **0.129**
	RP	0.883 ± 0.184	0.913 ± 0.154	**0.944** ± 0.130
	PMix	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
Swin Transformer: S	GAF	0.696 ± 0.236	0.838 ± 0.160	0.854 ± 0.198
	MTF	0.568 ± 0.226	0.820 ± 0.181	0.750 ± 0.194
	RP	0.875 ± 0.199	**0.943** ± **0.086**	0.924 ± 0.140
	PMix	0.792 ± 0.234	0.919 ± 0.087	0.882 ± 0.148
Swin Transformer: T	GAF	0.571 ± 0.216	0.765 ± 0.187	0.771 ± 0.198
	MTF	0.514 ± 0.166	0.806 ± 0.110	0.736 ± 0.154
	RP	0.727 ± 0.261	0.897 ± 0.127	0.833 ± 0.195
	PMix	**0.896** ± 0.167	0.938 ± 0.093	**0.944** ± 0.130

Table 4.6: Averages and standard deviations of the folds evaluation for the SwinTransformer V2 variants rounded to 3 decimal places, in the format *mean* ± *std*. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
Swin Transformer V2: B	GAF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
	MTF	0.568 ± 0.162	0.860 ± 0.085	0.764 ± 0.132
	RP	0.833 ± 0.222	0.931 ± 0.087	0.931 ± 0.127
	PMix	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
Swin Transformer V2: S	GAF	0.611 ± 0.239	0.829 ± 0.134	0.785 ± 0.183
	MTF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
	RP	0.833 ± 0.195	0.910 ± 0.097	0.924 ± 0.140
	PMix	**0.917** ± 0.163	**0.955** ± **0.083**	0.944 ± 0.130
Swin Transformer V2: T	GAF	0.674 ± 0.257	0.868 ± 0.119	0.819 ± 0.173
	MTF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
	RP	0.842 ± 0.210	0.901 ± 0.152	**0.951** ± **0.115**
	PMix	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129

Table 4.7: Memory size in Mega Bytes of each Transformers family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
Swin Transformer: T	110.083712	Swin Transformer: B	346.981336
Swin Transformer V2: T	110.336672	Swin Transformer V2: B	347.632024
MaxViT	122.144800	Vision Transformer: B 32	349.827128
Swin Transformer: S	195.355424	Vision Transformer: H 14	1213.214776
Swin Transformer V2: S	195.880352	Vision Transformer: L 16	1213.214776
Vision Transformer: B 16	343.200824	Vision Transformer: L 32	1222.049848

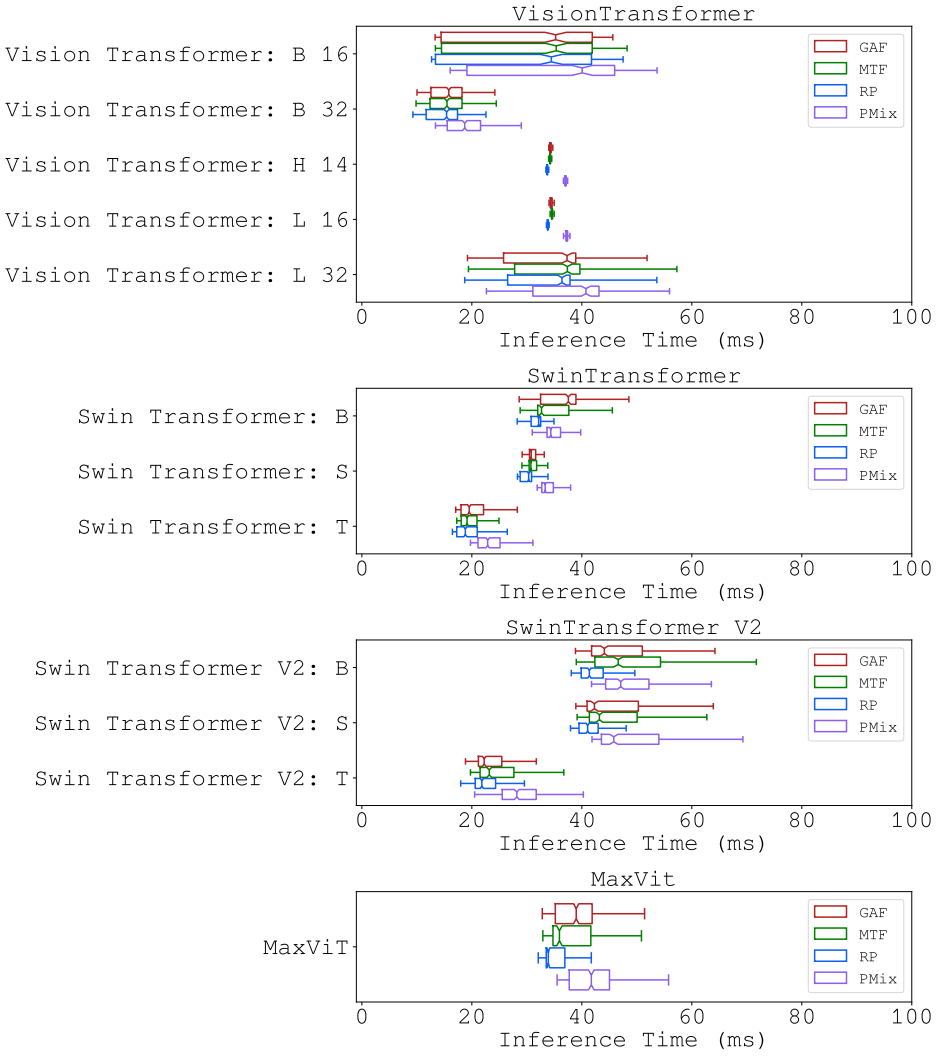


Figure 4.2: Inference time in milliseconds of each Transformers family model variant.

Residual Nets

The ResNet model and its variations are analyzed. ResNet introduced residual connections, which are links between non-adjacent layers that bypass intermediate layers. The two variations considered are Wide ResNet and ResNeXt. Wide ResNet expands the original network by increasing the number of channels per block, offering an alternative to increasing layer depth. In contrast, ResNeXt employs a multipath approach, aggregating paths through an additive operation. Instead of increasing width and depth, ResNeXt introduces an additional dimension for enhancement.

The experiments produced three score tables. Table 4.8 presents the ResNet scores for variants with 18, 34, 50, 101, and 152 layers. Among these, the PMix and RP methods outperformed the other projection methods. Specifically, the PMix method achieved the best scores when combined with the 50 and 101-layer variants. Table 4.9 displays the ResNeXt scores for variants with 50 or 101 layers, cardinality of 32 or 64, and bottleneck width of 4 or 8. Among these, the PMix and RP methods achieved the best scores. Notably, the RP method with the ResNeXt 101 $32 \times 8d$ variant achieved the highest scores. Table 4.10 lists the Wide ResNet scores for variants with 50 or 101 layers and a widening factor of 2. The PMix and RP methods consistently performed better across all metrics. Notably, the PMix method with the Wide ResNet 101-2 variant achieved the best scores for Cohen kappa and F1-Score, and the second-best score for Precision. Observing Tables 4.8, 4.9, and 4.10 together reveals that the Wide ResNet 101-2 with PMix was the top-performing combination in terms of scoring. However, this combination had the highest memory usage and was the fourth slowest in inference time. An alternative with nearly the second-best scores but significantly lower memory usage and inference time is the ResNet 50 with PMix.

Table 4.8: Averages and standard deviations of the folds evaluation for the ResNet variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
ResNet: 101	GAF	0.558 ± 0.223	0.848 ± 0.106	0.708 ± 0.279
	MTF	0.500 ± **0.000**	0.825 ± **0.074**	0.729 ± 0.129
	RP	0.667 ± 0.244	0.851 ± 0.147	0.867 ± 0.185
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
ResNet: 152	GAF	0.609 ± 0.266	0.874 ± 0.123	0.729 ± 0.291
	MTF	0.557 ± 0.168	0.787 ± 0.135	0.785 ± 0.183
	RP	0.792 ± 0.234	0.925 ± 0.089	0.894 ± 0.149
	PMix	0.875 ± 0.199	0.913 ± 0.154	0.931 ± 0.166
ResNet: 18	GAF	0.661 ± 0.256	0.807 ± 0.172	0.861 ± 0.182
	MTF	0.547 ± 0.188	0.799 ± 0.148	0.771 ± 0.155
	RP	0.854 ± 0.198	0.926 ± 0.093	0.924 ± 0.140
	PMix	0.771 ± 0.249	0.908 ± 0.109	0.868 ± 0.176
ResNet: 34	GAF	0.599 ± 0.210	0.799 ± 0.148	0.799 ± 0.165
	MTF	0.500 ± **0.000**	0.833 ± 0.098	0.725 ± 0.142
	RP	0.896 ± 0.167	0.938 ± 0.093	**0.944** ± 0.130
	PMix	0.854 ± 0.225	0.932 ± 0.111	0.931 ± **0.127**
ResNet: 50	GAF	0.510 ± 0.254	0.772 ± 0.178	0.681 ± 0.293
	MTF	0.470 ± 0.067	0.806 ± 0.110	0.715 ± 0.130
	RP	0.818 ± 0.226	0.931 ± 0.087	0.882 ± 0.148
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130

Table 4.9: Averages and standard deviations of the folds evaluation for the ResNeXt variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
ResNeXt: 101; 32x8d	GAF	0.729 ± 0.271	0.853 ± 0.179	0.847 ± 0.204
	MTF	0.568 ± 0.162	0.844 ± 0.102	0.771 ± 0.167
	RP	**0.875** ± 0.199	**0.943** ± 0.086	**0.924** ± 0.140
	PMix	0.758 ± 0.230	0.877 ± 0.145	0.882 ± 0.148
ResNeXt: 101; 64x4d	GAF	0.479 ± 0.188	0.752 ± 0.159	0.708 ± 0.169
	MTF	0.511 ± 0.198	0.580 ± 0.217	0.806 ± 0.257
	RP	0.758 ± 0.204	0.856 ± 0.149	0.917 ± 0.144
	PMix	0.833 ± 0.222	0.915 ± 0.115	0.903 ± 0.146
ResNeXt: 50; 32x4d	GAF	0.542 ± **0.144**	0.794 ± 0.161	0.750 ± 0.158
	MTF	0.568 ± 0.162	0.860 ± **0.085**	0.764 ± **0.132**
	RP	0.750 ± 0.282	0.870 ± 0.183	0.847 ± 0.204
	PMix	0.792 ± 0.234	0.919 ± 0.087	0.882 ± 0.148

Table 4.10: Averages and standard deviations of the folds evaluation for the Wide ResNet variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
Wide ResNet: 101-2	GAF	0.625 ± 0.226	0.826 ± 0.158	0.803 ± 0.172
	MTF	0.508 ± 0.110	0.702 ± 0.189	0.750 ± 0.194
	RP	0.842 ± 0.210	0.905 ± 0.159	**0.970 ± 0.101**
	PMix	**0.955 ± 0.101**	**0.967 ± 0.078**	0.944 ± 0.130
Wide ResNet: 50-2	GAF	0.486 ± 0.117	0.737 ± 0.154	0.713 ± 0.196
	MTF	0.550 ± 0.145	0.786 ± 0.142	0.773 ± 0.175
	RP	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
	PMix	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140

Table 4.11: Memory size in Mega Bytes of each Residual Nets family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
ResNet: 18	44.710680	ResNet: 152	232.595392
ResNet: 34	85.143704	Wide ResNet: 50-2	267.354672
ResNeXt: 50; 32x4d	91.937328	ResNeXt: 101; 64x4d	325.644024
ResNet: 50	94.049840	ResNeXt: 101; 32x8d	346.988280
ResNet: 101	170.019576	Wide ResNet: 101-2	499.369720

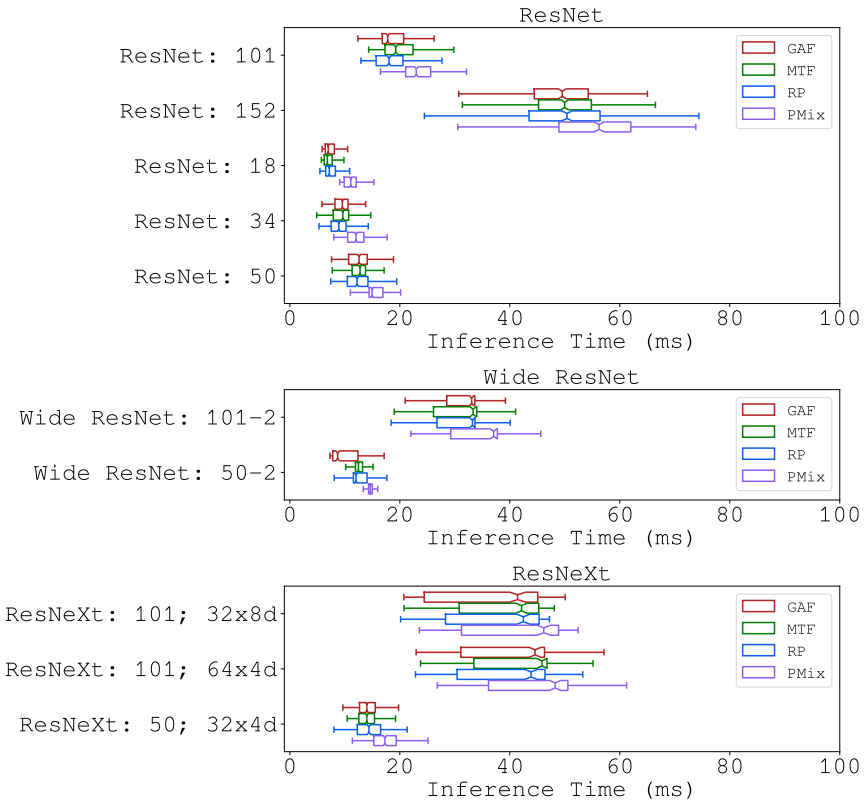


Figure 4.3: Inference time in milliseconds of each Residual Nets family model variant.

Mobile-Oriented

Mobile-oriented networks are designed specifically to address mobile hardware constraints. Three models were tested: MNASNet, MobileNet V2, and MobileNet V3. MobileNet V2, introduced first, incorporates architectural changes to reduce memory usage while maintaining accuracy, including inverted residual blocks. This alteration swaps high- and low-channel layers, connecting layers with fewer channels and thus reducing the number of parameters in the block. MNASNet selects blocks to fit a predefined architectural skeleton, optimizing model performance on real-world mobile hardware. MobileNet V3 combines these approaches and introduces additional changes, such as incorporating the NetAdapt [119] algorithm into the architectural search.

Three metric tables were generated for mobile-oriented family of CV models. Table 4.12 records the scores obtained by the MNASNet variants, which can have depth multipliers of 0.5, 0.75, 1.0, or 1.3, affecting the number of channels. Notably, the combination of MNASNet 1.0 with PMix achieved the best scores across all metrics. Table 4.13 presents the scores for MobileNet V2. The RP projection achieved the best Cohen kappa and Precision scores, while PMix obtained the highest F1-Score. However, RP demonstrated greater consistency, with lower variability in results compared to the standard deviations of PMix. Table 4.14 lists the MobileNet V3 variants, which include Small and Large configurations in terms of resource usage. Notably, PMix with the Large variant achieved the best Cohen kappa and Precision scores, while RP with the Small variant excelled in the F1-Score metric. From the tables 4.12, 4.13 and 4.14, the combination of MNASNet 1.0 with PMix emerges as the overall best case. This combination demonstrates a median inference time below 20 ms, as shown in Figure 4.4, and memory consumption under 15 MB, according to Table 4.15.

Table 4.12: Averages and standard deviations of the folds evaluation for the MNASNet variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
MNASNet: 0.5	GAF	0.513 ± 0.211	0.812 ± 0.167	0.552 ± 0.367
	MTF	0.480 ± 0.133	0.798 ± 0.136	0.681 ± 0.263
	RP	0.619 ± 0.260	0.787 ± 0.222	0.843 ± 0.197
	PMix	0.691 ± 0.220	0.866 ± 0.147	0.848 ± 0.148
MNASNet: 0.75	GAF	0.500 ± **0.000**	0.830 ± 0.072	0.750 ± 0.144
	MTF	0.524 ± 0.097	0.689 ± 0.142	0.771 ± 0.212
	RP	0.854 ± 0.225	0.932 ± 0.111	0.910 ± 0.172
	PMix	0.674 ± 0.298	0.796 ± 0.225	0.811 ± 0.230
MNASNet: 1.0	GAF	0.588 ± 0.213	0.698 ± 0.235	0.861 ± 0.220
	MTF	0.527 ± 0.185	0.839 ± 0.236	0.750 ± 0.433
	RP	0.521 ± 0.072	0.830 ± **0.064**	0.750 ± 0.125
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
MNASNet: 1.3	GAF	0.583 ± 0.163	0.842 ± 0.078	0.765 ± 0.139
	MTF	0.530 ± 0.164	0.833 ± 0.114	0.743 ± 0.153
	RP	0.523 ± 0.075	0.848 ± 0.073	0.743 ± **0.109**
	PMix	0.604 ± 0.249	0.884 ± 0.107	0.750 ± 0.312

Table 4.13: Averages and standard deviations of the folds evaluation for the MobileNet V2 variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
MobileNet V2	GAF	0.565 ± 0.214	0.776 ± 0.164	0.788 ± 0.294
	MTF	0.485 ± 0.200	0.773 ± 0.201	0.708 ± 0.193
	RP	**0.875** ± **0.199**	0.927 ± 0.116	**0.924** ± **0.140**
	PMix	0.854 ± 0.249	**0.951** ± **0.086**	0.889 ± 0.296

Table 4.14: Averages and standard deviations of the folds evaluation for the MobileNet V3 variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
MobileNet V3: Large	GAF	0.507 ± 0.140	0.758 ± 0.146	0.765 ± 0.178
	MTF	0.561 ± 0.227	0.777 ± 0.185	0.806 ± 0.195
	RP	0.683 ± 0.272	0.838 ± 0.191	0.818 ± 0.318
	PMix	**0.792** ± 0.234	0.908 ± 0.109	**0.910** ± **0.135**
MobileNet V3: Small	GAF	0.473 ± **0.090**	0.807 ± 0.153	0.639 ± 0.283
	MTF	0.474 ± 0.091	0.731 ± 0.165	0.697 ± 0.150
	RP	0.727 ± 0.236	**0.912** ± **0.087**	0.848 ± 0.148
	PMix	0.667 ± 0.246	0.822 ± 0.174	0.833 ± 0.207

Table 4.15: Memory size in Mega Bytes of each Mobile-Oriented family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
MNASNet: 0.5	3.761592	MNASNet: 1.0	12.420792
MNASNet: 0.75	7.568376	MobileNet V3: Large	16.819528
MobileNet V2	8.907032	MNASNet: 1.3	20.016568

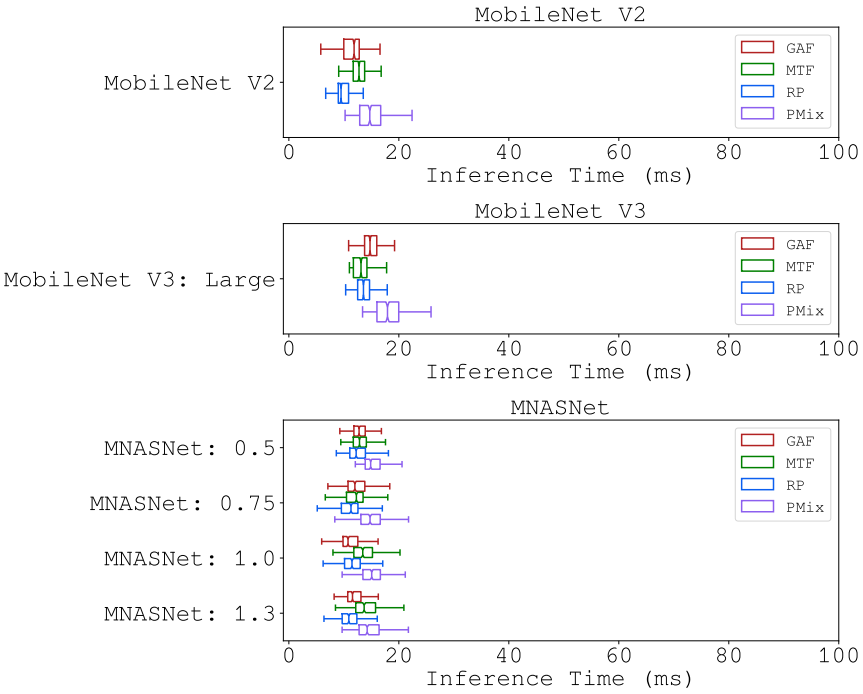


Figure 4.4: Inference time in milliseconds of each Mobile-Oriented family model variant.

Extreme Nets

Neural models that focus on specific concepts, such as layer depth, model compression, or residual connections, include VGG, DenseNet, and SqueezeNet. VGG employs 3×3 filters to allow for deeper network architectures by adding more layers, thus increasing model depth. DenseNet uses skipping connections among all pairs of architectural blocks in the network, which brings each layer closer to both the input and output, enhancing model performance. SqueezeNet focuses on minimizing memory usage through model compression techniques and by introducing a new architectural module that reduces the number of channels in a layer before applying large convolution filters, such as 3×3 filters. This approach significantly reduces the number of parameters.

Three metric tables were constructed, each corresponding to a model within the CV family. Table 4.16 presents the DenseNet results, where the variants include depths of 121, 161, 169, or 201 layers. For the 169 and 201 layer variants, the PMix method achieved superior performance, whereas the RP method was the best for the 161 layer variant. For the 121 layer variant, the PMix method obtained the highest Cohen kappa score, while the RP method excelled in the F1-Score and achieved perfect Precision. Overall, the DenseNet 161 with RP, DenseNet 201 with PMix, and DenseNet 121 with RP achieved the best scores in terms of Cohen kappa, F1-Score, and Precision metrics, respectively. Notably, the DenseNet 161 with RP demonstrated a good balance across metrics, attaining the best Cohen kappa score and the second-best F1 and Precision scores. Table 4.17 exhibits the SqueezeNet results for versions 1.0 and 1.1. The optimized version 1.1 achieved the highest scores when paired with the PMix method, attaining the best Cohen kappa and F1 scores. When combined with the RP method, the optimized version 1.1 achieved the best Precision score. Both combinations demonstrated generally strong performance across all metrics. Table 4.18 details the VGG scores across variants with 11, 13, 16, or 19 layers, with or without Batch Normalization (BN). The RP and PMix methods achieved the best scores for all variants, though some cases exhibited higher dispersion. Notably, the combination of VGG 16 with RP excelled in Cohen kappa and Precision metrics, while VGG 16 BN with PMix achieved the highest F1-Score. However, the VGG 16 with RP combination showed considerable dispersion in the F1-Score metric, making VGG 16 BN with PMix a more reliable choice. the combinations SqueezeNet 1.1 with PMix and VGG 16 BN with PMix stand out. Specifically, SqueezeNet 1.1 with PMix achieved the best Cohen kappa, while VGG 16 BN with PMix attained the highest F1-score among all Extreme Nets CV family models. The Figure 4.5 illustrates that SqueezeNet 1.1 with PMix outperforms most other variants in terms of inference speed. Additionally, Table 4.19 shows that this combination also ranks as the smallest in memory consumption.

Table 4.16: Averages and standard deviations of the folds evaluation for the DenseNet variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
DenseNet: 121	GAF	0.621 ± 0.248	0.795 ± 0.190	0.799 ± 0.196
	MTF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
	RP	0.771 ± 0.225	0.917 ± 0.099	**1.000** ± **0.000**
	PMix	0.862 ± 0.212	0.902 ± 0.167	0.931 ± 0.166
DenseNet: 161	GAF	0.557 ± 0.168	0.724 ± 0.191	0.788 ± 0.191
	MTF	0.676 ± 0.239	0.818 ± 0.167	0.847 ± 0.204
	RP	**0.896** ± 0.167	0.938 ± 0.093	0.944 ± 0.130
	PMix	0.750 ± 0.238	0.907 ± 0.085	0.861 ± 0.148
DenseNet: 169	GAF	0.480 ± 0.103	0.700 ± 0.211	0.767 ± 0.188
	MTF	0.653 ± 0.261	0.857 ± 0.132	0.806 ± 0.192
	RP	0.636 ± 0.275	0.848 ± 0.133	0.799 ± 0.153
	PMix	0.833 ± 0.222	0.938 ± 0.088	0.917 ± 0.144
DenseNet: 201	GAF	0.600 ± 0.256	0.861 ± 0.116	0.729 ± 0.291
	MTF	0.558 ± 0.223	0.854 ± **0.058**	0.701 ± 0.351
	RP	0.683 ± 0.183	0.773 ± 0.179	0.889 ± 0.175
	PMix	0.875 ± 0.199	**0.943** ± 0.086	0.924 ± 0.140

Table 4.17: Averages and standard deviations of the folds evaluation for the SqueezeNet variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
SqueezeNet: 1.0	GAF	0.591 ± 0.231	0.804 ± 0.192	0.771 ± 0.198
	MTF	0.586 ± 0.194	0.742 ± 0.197	0.812 ± 0.217
	RP	0.787 ± 0.206	0.816 ± 0.194	0.958 ± 0.144
	PMix	0.729 ± 0.271	0.838 ± 0.210	0.856 ± 0.211
SqueezeNet: 1.1	GAF	0.500 ± **0.000**	0.819 ± **0.080**	0.700 ± 0.105
	MTF	0.450 ± 0.112	0.794 ± 0.161	0.646 ± 0.249
	RP	0.904 ± 0.181	0.914 ± 0.168	**0.972** ± **0.096**
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	0.944 ± 0.130

Table 4.18: Averages and standard deviations of the folds evaluation for the VGG variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
VGG: 11	GAF	0.659 ± 0.257	0.840 ± 0.182	0.811 ± 0.201
	MTF	0.500 ± 0.174	0.790 ± 0.116	0.729 ± 0.155
	RP	0.829 ± 0.192	0.855 ± 0.188	0.944 ± 0.130
	PMix	0.833 ± 0.222	0.931 ± 0.087	0.903 ± 0.146
VGG: 11 BN	GAF	0.562 ± 0.155	0.848 ± 0.073	0.764 ± 0.132
	MTF	0.545 ± 0.151	0.849 ± 0.101	0.750 ± 0.151
	RP	0.875 ± 0.169	0.921 ± 0.098	0.944 ± 0.130
	PMix	0.854 ± 0.198	0.926 ± 0.093	0.924 ± 0.140
VGG: 13	GAF	0.667 ± 0.244	0.863 ± 0.121	0.819 ± 0.173
	MTF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
	RP	0.896 ± 0.198	0.944 ± 0.110	0.931 ± 0.166
	PMix	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
VGG: 13 BN	GAF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
	MTF	0.530 ± 0.164	0.833 ± 0.114	0.743 ± 0.153
	RP	0.833 ± 0.246	0.921 ± 0.131	0.896 ± 0.198
	PMix	0.873 ± 0.189	0.913 ± 0.154	0.924 ± 0.140
VGG: 16	GAF	0.583 ± 0.163	0.860 ± **0.052**	0.780 ± 0.113
	MTF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
	RP	**0.904** ± 0.181	0.930 ± 0.151	**0.972** ± **0.096**
	PMix	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
VGG: 16 BN	GAF	0.541 ± 0.196	0.861 ± 0.090	0.701 ± 0.257
	MTF	0.569 ± 0.177	0.828 ± 0.090	0.778 ± 0.152
	RP	0.625 ± 0.199	0.856 ± 0.087	0.799 ± 0.125
	PMix	0.896 ± 0.198	**0.960** ± 0.075	0.951 ± 0.115
VGG: 19	GAF	0.568 ± 0.117	0.855 ± 0.052	0.778 ± 0.109
	MTF	0.479 ± 0.078	0.776 ± 0.139	0.736 ± 0.154
	RP	0.854 ± 0.225	0.932 ± 0.111	0.910 ± 0.172
	PMix	0.688 ± 0.217	0.879 ± 0.077	0.840 ± 0.144
VGG: 19 BN	GAF	0.549 ± 0.199	0.790 ± 0.152	0.757 ± 0.172
	MTF	0.553 ± 0.262	0.764 ± 0.211	0.743 ± 0.215
	RP	0.875 ± 0.199	0.927 ± 0.116	0.931 ± 0.166
	PMix	0.708 ± 0.257	0.885 ± 0.123	0.833 ± 0.195

Table 4.19: Memory size in Mega Bytes of each Extreme Nets family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
SqueezeNet: 1.1	2.894136	VGG: 11 BN	515.120376
SqueezeNet: 1.0	2.945848	VGG: 13	515.836216
DenseNet: 121	27.826576	VGG: 13 BN	515.860008
DenseNet: 169	49.955344	VGG: 16	537.075000
DenseNet: 201	72.391952	VGG: 16 BN	537.109104
DenseNet: 161	105.909584	VGG: 19	558.313784
VGG: 11	515.098168	VGG: 19 BN	558.358200

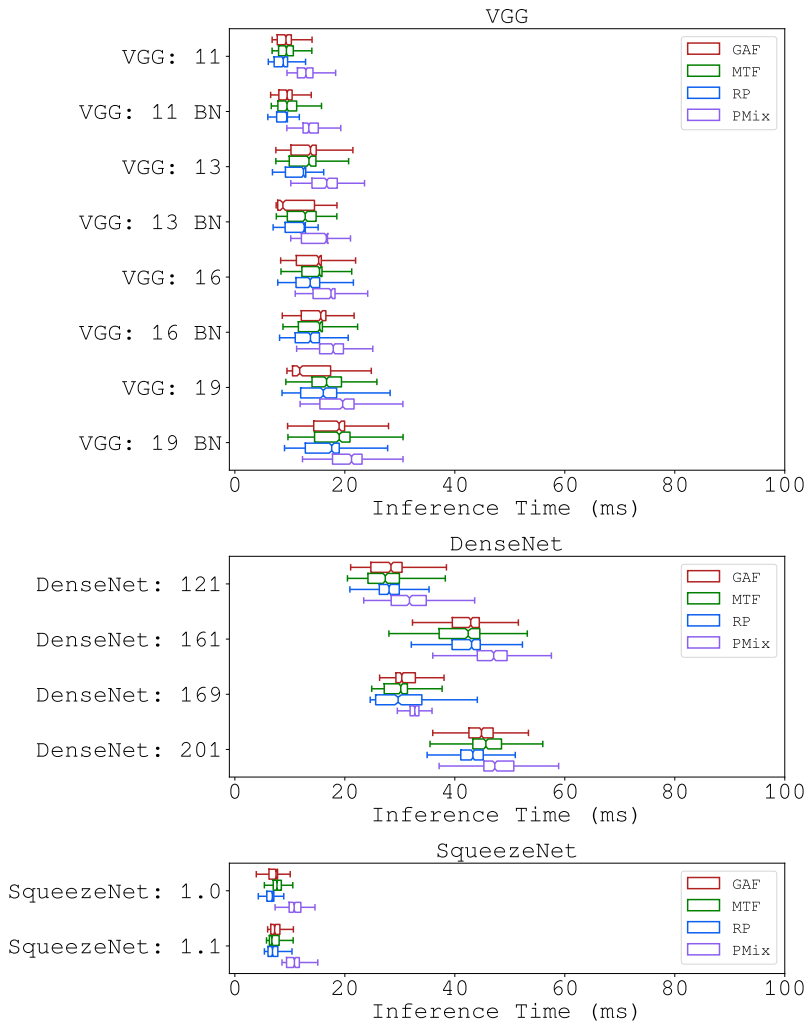


Figure 4.5: Inference time in milliseconds of each Extreme Nets family model variant.

Efficiency-Oriented

The models designed for efficient resource utilization aim to maximize performance with fewer parameters. This work examined ShuffleNet V2, EfficientNet, and EfficientNet V2. ShuffleNet V2 is an advancement of ShuffleNet, introducing the channel shuffle operator to facilitate information exchange among channels. It improves upon its predecessor by incorporating a channel split operation within each block, which avoids the use of costly grouped convolutions. EfficientNet focuses on model scaling through a compound resizing method that proportionally increases multiple dimensions, such as depth, number of channels, and resolution. This approach creates a highly efficient base model that can be scaled up to larger variants while preserving the original model's advantages. EfficientNet V2 builds on the original EfficientNet by proposing non-proportional scaling and utilizing network architecture search. It also introduces progressive learning, which involves gradually increasing dataset regularization.

Three score tables were constructed for the Efficiency-Oriented CV family. Table 4.20 lists the results for EfficientNet variants ranging from B0, the smallest, to B4, the largest, in terms of parameter scaling. The PMix projection achieved the highest scores for variants B0, B1, and B2. For the B3 variant, the MTF method excelled in the F1-score, while the RP method performed better for the other metrics. Overall, the combination of EfficientNet B1 with PMix emerged as the top performer. Table 4.21 presents the scores for EfficientNet V2, with the PMix projection outperforming all other methods. Table 4.22 displays the metrics for ShuffleNet V2 variants, which include multipliers of $\times 0.5$, $\times 1.0$, $\times 1.5$, and $\times 2.0$ on the number of channels in each architectural block. Across all variants, the PMix projection method outperformed all others. Specifically, the best scores for ShuffleNet V2 were achieved with the $\times 0.5$ and $\times 1.0$ variants combined with the PMix method. When analyzing the results from Tables 4.20, 4.21, and 4.22, it is evident that the usage of EfficientNet V2, ShuffleNet V2 $\times 0.5$, and ShuffleNet V2 $\times 1.0$ with PMix achieved high scores across all metrics, including the best Cohen kappa and F1 scores, as well as the second-best Precision. Among these, the ShuffleNet V2 $\times 0.5$ with PMix was the most efficient in terms of memory usage, as shown in Table 4.23, while being one of the fastest in inference, as visible in the Figure 4.6.

Table 4.20: Averages and standard deviations of the folds evaluation for the EfficientNet variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
EfficientNet: B0	GAF	0.569 ± 0.177	0.828 ± 0.090	0.778 ± 0.152
	MTF	0.507 ± 0.206	0.806 ± 0.155	0.694 ± 0.257
	RP	0.736 ± 0.199	0.856 ± 0.142	0.882 ± 0.148
	PMix	0.841 ± 0.231	0.916 ± 0.134	0.896 ± 0.198
EfficientNet: B1	GAF	0.517 ± 0.247	0.755 ± 0.178	0.688 ± 0.278
	MTF	0.503 ± 0.131	0.735 ± 0.186	0.757 ± 0.172
	RP	0.694 ± 0.294	0.841 ± 0.196	0.856 ± 0.211
	PMix	**0.875** ± 0.199	**0.943** ± 0.086	**0.924** ± **0.140**
EfficientNet: B2	GAF	0.436 ± 0.161	0.729 ± 0.168	0.546 ± 0.344
	MTF	0.473 ± **0.117**	0.671 ± 0.228	0.750 ± 0.217
	RP	0.682 ± 0.276	0.858 ± 0.179	0.806 ± 0.192
	PMix	0.854 ± 0.198	0.926 ± 0.093	**0.924** ± **0.140**
EfficientNet: B3	GAF	0.583 ± 0.163	0.841 ± **0.082**	0.792 ± 0.163
	MTF	0.579 ± 0.229	0.863 ± 0.116	0.719 ± 0.339
	RP	0.696 ± 0.210	0.817 ± 0.151	0.868 ± 0.176
	PMix	0.604 ± 0.225	0.788 ± 0.181	0.792 ± 0.209

Table 4.21: Averages and standard deviations of the folds evaluation for the EfficientNetV2 variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
EfficientNet V2	GAF	0.600 ± 0.200	0.826 ± 0.167	0.800 ± 0.197
	MTF	0.545 ± **0.151**	0.849 ± 0.101	0.750 ± 0.151
	RP	0.708 ± 0.234	0.895 ± **0.080**	0.840 ± 0.144
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± **0.130**

Table 4.22: Averages and standard deviations of the folds evaluation for the ShuffleNet V2 variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
ShuffleNet V2: x0.5	GAF	0.523 ± 0.208	0.784 ± 0.199	0.736 ± 0.210
	MTF	0.473 ± 0.090	0.851 ± 0.090	0.667 ± 0.280
	RP	0.821 ± 0.231	0.876 ± 0.190	0.910 ± 0.172
	PMix	**0.917** ± 0.163	**0.955** ± **0.083**	0.944 ± 0.130
ShuffleNet V2: x1.0	GAF	0.504 ± 0.194	0.744 ± 0.180	0.688 ± 0.285
	MTF	0.591 ± 0.202	0.862 ± 0.122	0.775 ± 0.184
	RP	0.727 ± 0.236	0.932 ± 0.095	0.885 ± 0.160
	PMix	**0.917** ± 0.163	**0.955** ± **0.083**	0.944 ± 0.130
ShuffleNet V2: x1.5	GAF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
	MTF	0.489 ± 0.156	0.772 ± 0.122	0.659 ± 0.248
	RP	0.592 ± 0.220	0.792 ± 0.207	0.767 ± 0.301
	PMix	0.800 ± 0.198	0.868 ± 0.148	**0.951** ± **0.115**
ShuffleNet V2: x2.0	GAF	0.625 ± 0.226	0.861 ± 0.110	0.792 ± 0.179
	MTF	0.527 ± 0.199	0.700 ± 0.230	0.778 ± 0.234
	RP	0.480 ± 0.235	0.789 ± 0.189	0.629 ± 0.358
	PMix	0.729 ± 0.249	0.896 ± 0.106	0.847 ± 0.173

Table 4.23: Memory size in Mega Bytes of each Efficiency-Oriented family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
ShuffleNet V2: x0.5	1.376760	EfficientNet: B1	26.064688
ShuffleNet V2: x1.0	5.024008	EfficientNet: B2	30.816952
ShuffleNet V2: x1.5	9.924088	EfficientNet: B3	42.799144
EfficientNet: B0	16.041664	EfficientNet V2	80.722888
ShuffleNet V2: x2.0	21.397768		

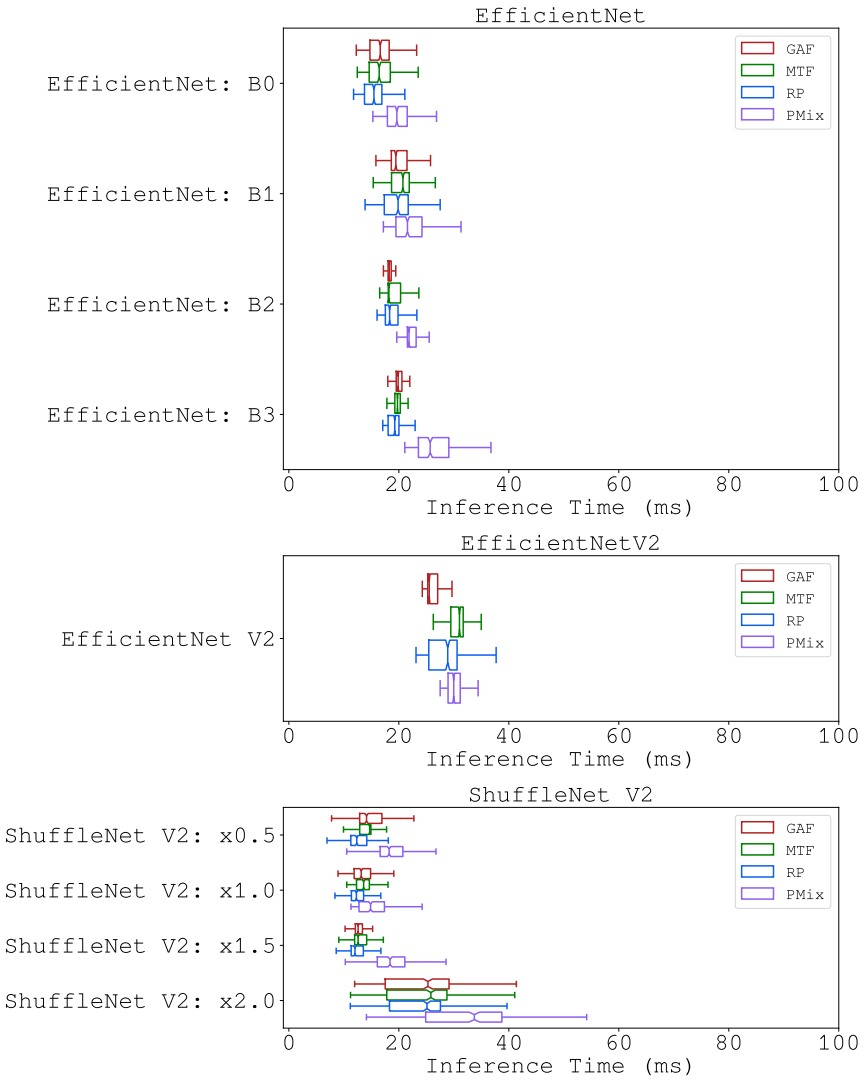


Figure 4.6: Inference time in milliseconds of each Efficiency-Oriented family model variant.

Further Architectures

The analysis includes the following models: AlexNet, ConvNeXt, and RegNet. Introduced in 2012, AlexNet was one of the earliest deep learning models designed to be trained across multiple GPUs, which accelerated the training process and utilized dropout to mitigate overfitting. In contrast, ConvNeXt, a modern model from 2022, integrates various convolutional techniques from recent years, such as patchified convolutions, inverted bottlenecks, and grouped convolutions, with the goal of advancing traditional convolutional networks. On the other hand, RegNet departs from designing individual networks by focusing on creating network families defined by linear parameter spaces, facilitating architectural search within these defined populations. The experiments encompassed networks with significant variations among them.

Three score tables were generated for the remaining computer vision models. The Table 4.25 contains the scores of AlexNet. Notably, the PMix projection earned the best scores for all metrics in that table. Table 4.26 presents the results for the ConvNeXt model, with variants including Tiny, Small, Base, and Large in terms of parameter size. Among these variants, the RP and PMix methods achieved the best scores overall, though some instances, such as RP with the Tiny variant, exhibited higher dispersion, particularly for the Cohen kappa score. Specifically, the PMix method with the ConvNeXt Tiny variant achieved the highest F1-score, while the PMix method with the ConvNeXt Small variant obtained the best Cohen kappa and Precision scores, and also secured the second best F1-score. Table 4.24 exhibits the results for RegNet variants, categorized into RegNetX and RegNetY design spaces [111], with varying float operations per second rates such as 400 Mega Flops (MF) or 16 Giga Flops (GF). Among these variants, several achieved scores above the third quartile across all metrics. For the X space, notable cases include RP with the 400 MF and 800 MF variants, and PMix with the 800 MF, 3.2 GF, 8 GF, and 16 GF variants. For the Y space, noteworthy cases are RP with the 400 MF, 1.6 GF, 16 GF, and 32 GF variants, and PMix with the 800 MF, 3.2 GF, and 16 GF variants. Among these high-scoring variants, the PMix method with the RegNet X 3.2 GF, RegNet X 800 MF, RegNet Y 400 MF, and RegNet Y 800 MF variants achieved the best Cohen kappa and F1 scores, and the third best Precision score. Of these top-performing combinations, the RegNet Y 400 MF with PMix had the lowest memory usage, as shown in Table 4.27, while the RegNet X 800 MF with PMix had the fastest inferences of the model variants, as seen in Figure 4.7. When evaluating the top-performing models from each type within the Diverse CV family, the RegNet Y 400 MF with RP, and the RegNet X 800 MF and AlexNet with PMix achieved the highest scores. Notably, the RegNet Y 400 MF with RP exhibited the lowest memory usage, as shown in Table 4.27, while AlexNet had the fastest inference times across all projections, as illustrated in Figure 4.7.

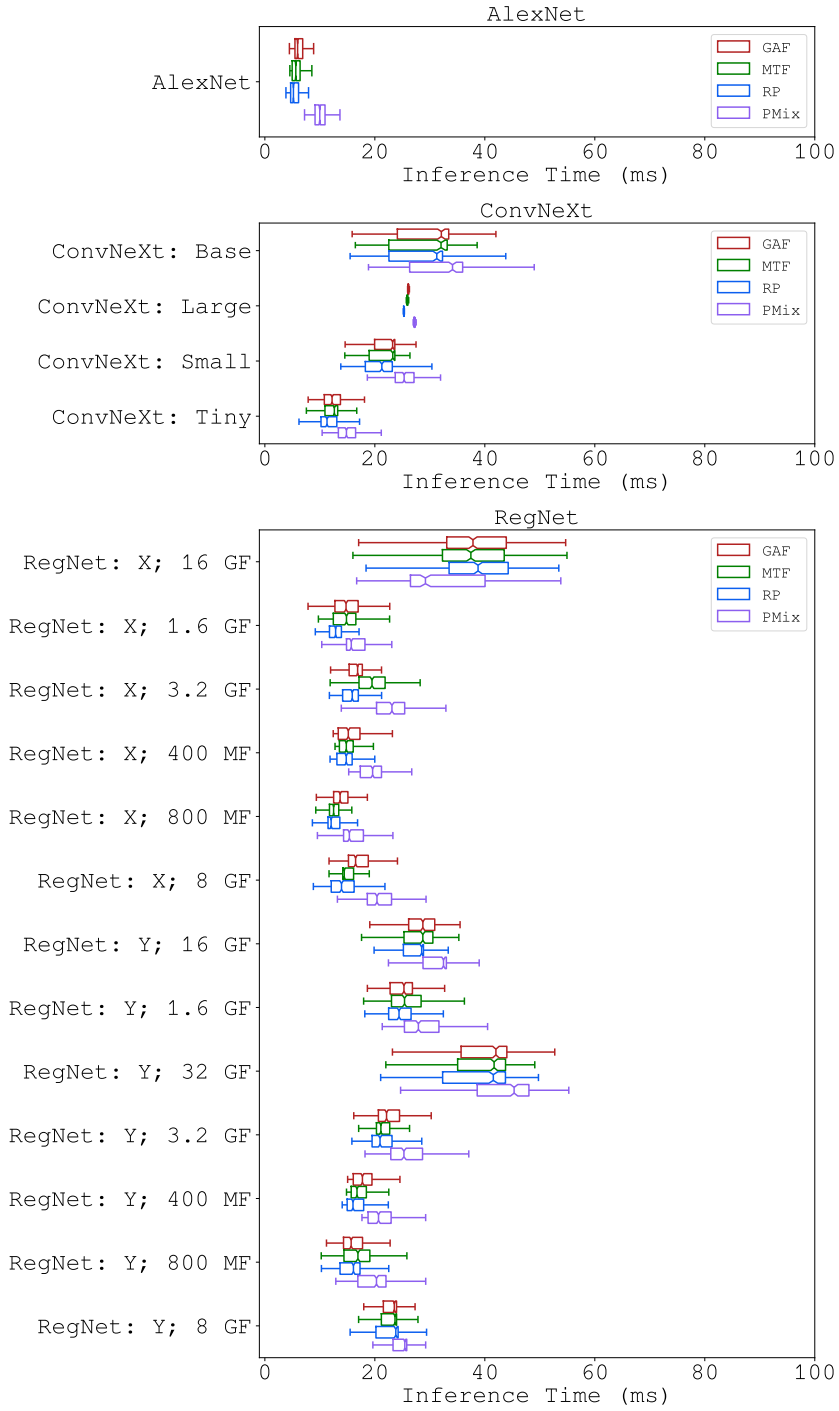


Figure 4.7: Inference time in milliseconds of each Diverse family model variant.

Table 4.24: Averages and standard deviations of the folds evaluation for the RegNet variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
RegNet: X; 16 GF	GAF	0.668 ± 0.226	0.837 ± 0.167	0.841 ± 0.202
	MTF	0.542 ± 0.226	0.771 ± 0.154	0.736 ± 0.303
	RP	0.729 ± 0.198	0.838 ± 0.142	0.902 ± 0.178
	PMix	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
RegNet: X; 1.6 GF	GAF	0.515 ± 0.174	0.817 ± 0.123	0.736 ± 0.154
	MTF	0.553 ± 0.176	0.829 ± 0.114	0.764 ± 0.170
	RP	0.768 ± 0.233	0.865 ± 0.195	0.955 ± 0.101
	PMix	0.854 ± 0.225	0.918 ± 0.151	0.910 ± 0.172
RegNet: X; 32 GF	GAF	0.508 ± 0.095	0.830 ± 0.094	0.735 ± 0.117
	MTF	0.527 ± 0.185	0.812 ± 0.157	0.705 ± 0.292
	RP	0.862 ± 0.184	0.896 ± 0.154	0.944 ± 0.130
	PMix	0.896 ± 0.198	0.930 ± 0.151	0.931 ± 0.166
RegNet: X; 3.2 GF	GAF	0.461 ± 0.095	0.810 ± 0.088	0.657 ± 0.278
	MTF	0.550 ± 0.098	0.772 ± 0.122	0.800 ± 0.197
	RP	0.896 ± 0.198	0.914 ± 0.168	0.931 ± 0.166
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	0.944 ± 0.130
RegNet: X; 400 MF	GAF	0.523 ± 0.075	0.831 ± 0.104	0.778 ± 0.150
	MTF	0.479 ± 0.113	0.773 ± 0.095	0.729 ± 0.155
	RP	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
	PMix	0.550 ± 0.098	0.791 ± 0.119	0.792 ± 0.163
RegNet: X; 800 MF	GAF	0.594 ± 0.278	0.857 ± 0.145	0.720 ± 0.306
	MTF	0.402 ± 0.117	0.656 ± 0.238	0.667 ± 0.173
	RP	0.875 ± 0.199	0.943 ± 0.086	0.951 ± 0.115
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	0.944 ± 0.130
RegNet: X; 8 GF	GAF	0.545 ± 0.151	0.849 ± 0.101	0.750 ± 0.151
	MTF	0.530 ± 0.164	0.812 ± 0.157	0.742 ± 0.169
	RP	0.854 ± 0.198	0.932 ± 0.095	**0.970** ± **0.101**
	PMix	0.875 ± 0.199	0.951 ± 0.086	0.939 ± 0.135
RegNet: Y; 16 GF	GAF	0.612 ± 0.196	0.811 ± 0.149	0.818 ± 0.197
	MTF	0.477 ± 0.118	0.778 ± 0.117	0.727 ± 0.163
	RP	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
	PMix	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
RegNet: Y; 1.6 GF	GAF	0.636 ± 0.259	0.846 ± 0.173	0.799 ± 0.217
	MTF	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
	RP	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
	PMix	0.795 ± 0.218	0.914 ± 0.092	0.882 ± 0.148
RegNet: Y; 32 GF	GAF	0.583 ± 0.222	0.822 ± 0.158	0.764 ± 0.170
	MTF	0.568 ± 0.226	0.823 ± 0.173	0.750 ± 0.185
	RP	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
	PMix	0.667 ± 0.222	0.883 ± 0.074	0.819 ± 0.137
RegNet: Y; 3.2 GF	GAF	0.712 ± 0.280	0.881 ± 0.143	0.826 ± 0.199
	MTF	0.547 ± 0.246	0.753 ± 0.206	0.758 ± 0.212
	RP	0.826 ± 0.234	0.910 ± 0.119	0.896 ± 0.155
	PMix	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
RegNet: Y; 400 MF	GAF	0.590 ± 0.260	0.823 ± 0.173	0.771 ± 0.211
	MTF	0.475 ± 0.112	0.690 ± 0.193	0.729 ± 0.198
	RP	**0.917** ± 0.163	**0.955** ± 0.083	0.944 ± 0.130
	PMix	0.773 ± 0.236	0.919 ± 0.087	0.861 ± 0.148
RegNet: Y; 800 MF	GAF	0.521 ± 0.072	0.832 ± **0.061**	0.742 ± 0.121
	MTF	0.486 ± 0.191	0.660 ± 0.240	0.713 ± 0.196
	RP	0.792 ± 0.257	0.902 ± 0.121	0.868 ± 0.296
	PMix	**0.917** ± 0.163	**0.955** ± 0.083	0.944 ± 0.130
RegNet: Y; 8 GF	GAF	0.508 ± 0.029	0.790 ± 0.123	0.750 ± 0.158
	MTF	0.482 ± 0.166	0.738 ± 0.158	0.648 ± 0.303
	RP	0.875 ± 0.199	0.927 ± 0.116	0.924 ± 0.140
	PMix	0.771 ± 0.249	0.908 ± 0.109	0.868 ± 0.176

Table 4.25: Averages and standard deviations of the folds evaluation for the AlexNet variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
AlexNet	GAF	0.545 ± **0.151**	0.849 ± 0.101	0.750 ± 0.151
	MTF	0.598 ± 0.247	0.827 ± 0.174	0.773 ± 0.163
	RP	0.704 ± 0.204	0.819 ± 0.168	0.910 ± 0.135
	PMix	**0.917** ± 0.163	**0.955** ± **0.083**	**0.944** ± **0.130**

Table 4.26: Averages and standard deviations of the folds evaluation for the ConvNeXt variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
ConvNeXt: Base	GAF	0.536 ± 0.157	0.792 ± 0.137	0.764 ± 0.170
	MTF	0.473 ± 0.144	0.789 ± 0.159	0.667 ± 0.268
	RP	0.883 ± 0.184	0.913 ± 0.154	**0.944** ± **0.130**
	PMix	0.854 ± 0.198	0.926 ± 0.093	0.924 ± 0.140
ConvNeXt: Large	GAF	0.611 ± 0.239	0.845 ± 0.124	0.785 ± 0.183
	MTF	0.545 ± 0.151	0.849 ± 0.101	0.750 ± 0.151
	RP	0.862 ± 0.184	0.896 ± 0.154	**0.944** ± **0.130**
	PMix	0.862 ± 0.184	0.896 ± 0.154	**0.944** ± **0.130**
ConvNeXt: Small	GAF	0.708 ± 0.257	0.885 ± 0.123	0.833 ± 0.195
	MTF	0.611 ± 0.239	0.845 ± 0.124	0.785 ± 0.183
	RP	0.854 ± 0.198	0.926 ± 0.093	0.924 ± 0.140
	PMix	**0.896** ± 0.167	0.938 ± 0.093	**0.944** ± **0.130**
ConvNeXt: Tiny	GAF	0.688 ± 0.241	0.884 ± 0.101	0.826 ± 0.168
	MTF	0.523 ± **0.075**	0.833 ± 0.090	0.750 ± 0.151
	RP	0.778 ± 0.257	0.903 ± 0.113	0.875 ± 0.157
	PMix	0.875 ± 0.199	**0.951** ± **0.086**	0.939 ± 0.135

Table 4.27: Memory size in Mega Bytes of each Diverse family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
RegNet: Y; 400 MF	15.618528	RegNet: Y; 8 GF	149.476520
RegNet: X; 400 MF	20.385968	RegNet: X; 8 GF	150.624888
RegNet: Y; 800 MF	22.598608	ConvNeXt: Small	197.824952
RegNet: X; 800 MF	26.354432	RegNet: X; 16 GF	208.937392
RegNet: X; 1.6 GF	33.118416	AlexNet	228.048184
RegNet: Y; 1.6 GF	41.264048	RegNet: Y; 16 GF	322.287328
RegNet: X; 3.2 GF	57.161352	ConvNeXt: Base	350.274104
RegNet: Y; 3.2 GF	71.708240	RegNet: Y; 32 GF	565.367496
ConvNeXt: Tiny	111.286712	ConvNeXt: Large	784.933688

Table 4.28: Averages and standard deviations of the folds evaluation for the Non-CV variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Cohen Kappa	F1 Score	Precision
Arsenal	0.639 ± 0.252	0.804 ± 0.140	0.819 ± 0.204
BOSS Ensemble	0.688 ± 0.241	0.884 ± 0.101	0.826 ± 0.168
Zhao's CNN Classifier	0.875 ± 0.199	0.951 ± 0.086	0.939 ± 0.135
Canonical Interval Forest Classifier	0.862 ± 0.184	0.896 ± 0.154	0.944 ± 0.130
Catch 22 Classifier	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
Continuous Interval Tree	0.632 ± 0.240	0.856 ± 0.112	0.799 ± 0.165
Contractable BOSS	0.792 ± 0.234	0.919 ± 0.087	0.882 ± 0.148
DrCIF Classifier	0.904 ± 0.181	0.930 ± 0.151	**0.972 ± 0.096**
Elastic Ensemble	0.812 ± 0.188	0.893 ± 0.097	0.924 ± 0.140
Wang's FCN Classifier	0.842 ± 0.210	0.901 ± 0.152	0.924 ± 0.140
Inception Time Classifier	0.799 ± 0.242	0.898 ± 0.116	0.896 ± 0.155
Individual BOSS	0.875 ± 0.169	0.921 ± 0.098	0.944 ± 0.130
Individual Inception Classifier	0.694 ± 0.228	0.858 ± 0.111	0.875 ± 0.163
Individual Ordinal TDE	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
Individual TDE	0.862 ± 0.184	0.896 ± 0.154	0.944 ± 0.130
K-Neighbors Time Series Classifier	0.875 ± 0.199	0.927 ± 0.116	0.931 ± 0.166
LITE Time Classifier	0.771 ± 0.249	0.908 ± 0.109	0.868 ± 0.176
Wang's MLP Classifier	0.485 ± **0.050**	0.821 ± 0.102	0.722 ± 0.130
MUSE	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
Ordinal TDE	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
RDST Classifier	0.633 ± 0.196	0.842 ± 0.125	0.819 ± 0.137
REDCOMETS	0.508 ± 0.095	0.817 ± 0.101	0.743 ± 0.153
Random Interval Classifier	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
Random Interval Spectral Ensemble Classifier	**0.938** ± 0.155	**0.971** ± 0.068	**0.972 ± 0.096**
Rocket Classifier	0.729 ± 0.225	0.859 ± 0.122	0.868 ± 0.176
Rotation Forest Classifier	0.854 ± 0.225	0.932 ± 0.111	0.910 ± 0.172
Shape DTW	0.729 ± 0.225	0.875 ± 0.106	0.868 ± 0.176
Shapelet Transform Classifier	0.625 ± 0.199	0.873 ± **0.067**	0.803 ± 0.131
Summary Classifier	0.883 ± 0.184	0.913 ± 0.154	0.944 ± 0.130
Supervised Time Series Forest	0.862 ± 0.184	0.896 ± 0.154	**0.972 ± 0.096**
TS Fresh Classifier	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
Temporal Dictionary Ensemble	**0.938** ± 0.155	**0.971** ± 0.068	**0.972 ± 0.096**
Time Series Forest Classifier	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
WEASEL	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
WEASEL V2	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130

4.2.2 Non-computer vision models comparison

Table 4.28 presents the scores for non-computationally-visual baseline models. The models Individual Ordinal TDE, Random Interval Classifier, Random Interval Spectral Ensemble Classifier (RISEC), TS Fresh Classifier, Temporal Dictionary Ensemble (TDE), and WEASEL V2 achieved high scores across all metrics. Among these, RISEC and TDE surpassed the other models in every score metric. Specifically, RISEC demonstrated faster inference, as shown in Figure 4.8,while TDE had lower memory consumption, according to Table 4.29.

Table 4.29: Memory size in Mega Bytes of each Non-CV family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
Continuous Interval Tree	0.011384	Arsenal	3.320648
Individual Ordinal TDE	0.019264	Wang's FCN Classifier	3.353256
Individual TDE	0.019264	REDCOMETS	3.890264
Individual BOSS	0.079880	K-Neighbors Time Series Classifier	4.943104
Summary Classifier	0.244776	Elastic Ensemble	5.229568
Catch 22 Classifier	0.245600	Random Interval Classifier	5.303208
WEASEL	0.424416	Time Series Forest Classifier	6.495832
WEASEL V2	0.486456	Shape DTW	7.591056
Temporal Dictionary Ensemble	0.586264	BOSS Ensemble	8.204624
Ordinal TDE	0.587536	Canonical Interval Forest Classifier	9.487720
Rocket Classifier	0.651192	Supervised Time Series Forest	10.044264
TS Fresh Classifier	0.665896	Individual Inception Classifier	10.378496
MUSE	0.776376	DrCIF Classifier	13.327184
RDST Classifier	0.971184	Shapelet Transform Classifier	17.377296
Random Interval Spectral Ensemble Classifier	1.171312	LITE Time Classifier	25.222448
Zhao's CNN Classifier	1.602328	Rotation Forest Classifier	33.036872
Contractable BOSS	1.902736	Inception Time Classifier	51.942384
Wang's MLP Classifier	1.911008		

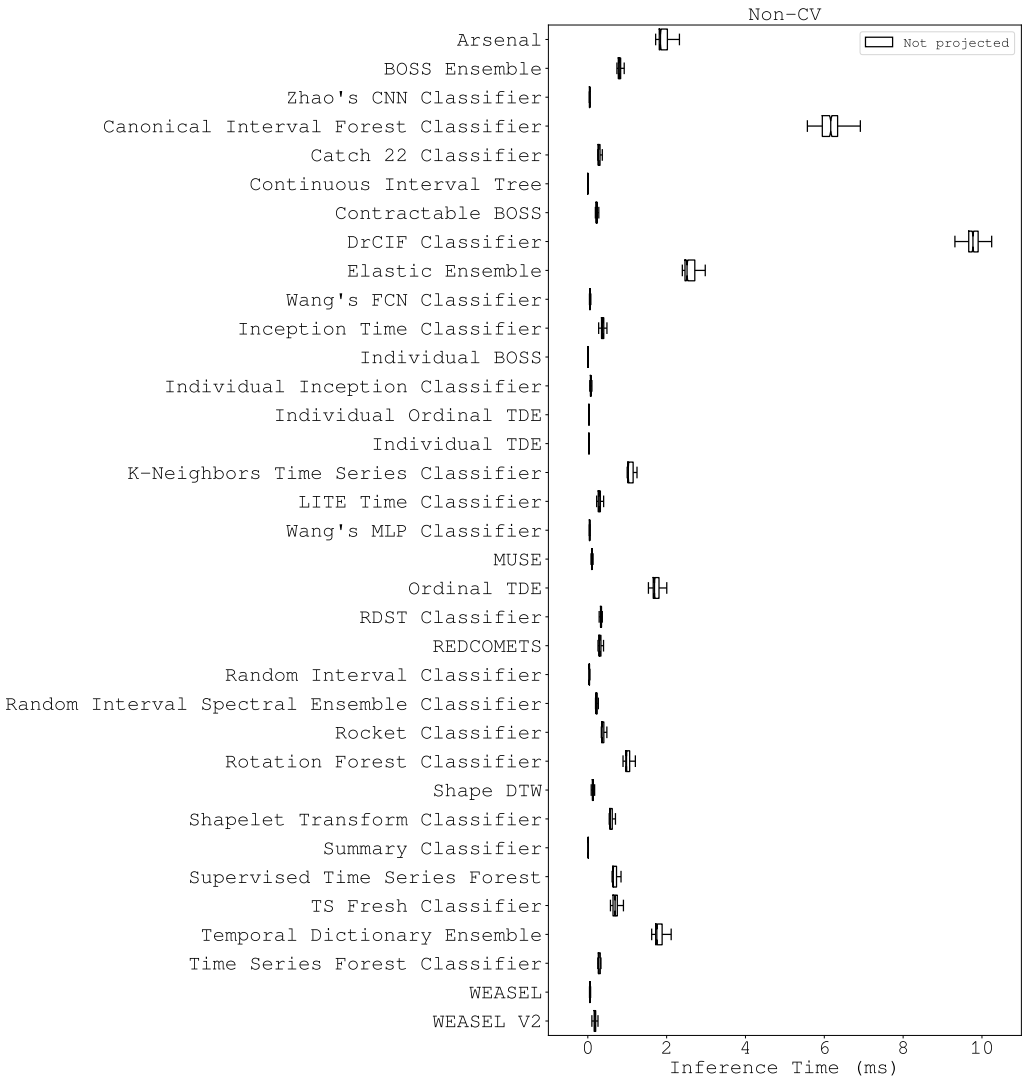


Figure 4.8: Inference time in milliseconds of each Non-CV family model variant.

Table 4.30: Averages and standard deviations of the folds evaluation for the best models variants rounded to 3 decimal places, in the format $mean \pm std$. The colors red and green represent, respectively, values lesser or equal than the first quartile and values greater or equal than the third quartile, when considering one mean or standard deviation column. The color yellow represents the other values that are none of the other two colors.

Model	Projection	Cohen Kappa	F1 Score	Precision
AlexNet	PMix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
MNASNet: 1.0	PMix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
RandomIntervalSpectralEnsembleClassifier	Not projected	0.938 ± 0.155	**0.971 ± 0.068**	**0.972 ± 0.096**
RegNet: Y; 400 MF	RP	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
ResNet: 50	PMix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
ShuffleNet V2: x0.5	PMix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
SqueezeNet: 1.1	PMix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
Swin Transformer V2: S	PMix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
TemporalDictionaryEnsemble	Not projected	0.938 ± 0.155	**0.971 ± 0.068**	**0.972 ± 0.096**
VGG: 16 BN	PMix	0.896 ± 0.198	0.960 ± 0.075	0.951 ± 0.115
Vision Transformer: B 32	PMix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
Wide ResNet: 101-2	PMix	**0.955 ± 0.101**	0.967 ± 0.078	0.944 ± 0.130

Table 4.31: Memory size in Mega Bytes of each best models family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
TemporalDictionaryEnsemble	0.586264	ResNet: 50	94.049840
RandomIntervalSpectralEnsembleClassifier	1.171312	Swin Transformer V2: S	195.880352
ShuffleNet V2: x0.5	1.376760	AlexNet	228.048184
SqueezeNet: 1.1	2.894136	Vision Transformer: B 32	349.827128
MNASNet: 1.0	12.420792	Wide ResNet: 101-2	499.369720
RegNet: Y; 400 MF	15.617376	VGG: 16 BN	537.109104

4.2.3 Comparison of the Top-Performing Models

The best combinations between models and projections from previous analyses are aggregated in Table 4.30. The top-performing models include the TDE, the RISEC, and the Wide ResNet 100-2 with PMix. While the Wide ResNet achieved the highest Cohen kappa score, it is notable that this model was the second largest in terms of memory usage, as indicated in Table 4.31. Additionally, it did not achieve the highest F1-Score or Precision and was the second slowest in terms of inference speed, as shown in Figure 4.9. In contrast, the TDE and RISEC models excelled in usability and security metrics, including F1-Score and Precision. They also demonstrated superior performance in terms of inference speed and memory efficiency. Consequently, while the CV approach, represented by the Wide ResNet 100-2 with PMix, achieved higher accuracy, the non-CV approach, embodied by the TDE and RISEC, offers better resource efficiency and speed, making it a more practical choice for applications requiring lower resource consumption and faster performance.

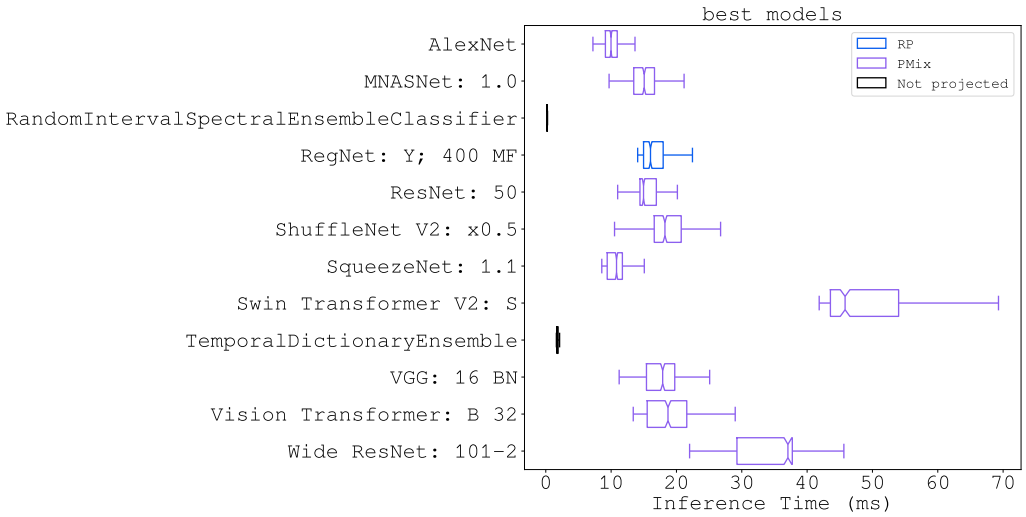


Figure 4.9: Inference time in milliseconds of each best models family model variant.

4.3 Limitations

Despite the promising results, some limitations can be considered. The experiments utilized only the BUTPPG dataset for testing. This has a series of implications in our context. Firstly, even though the proposed method allowed the use of CV models with a good performance in the BUTPPG dataset, the same could not necessarily be concluded for different datasets. This is because different methods of measurement, sensor qualities, signal lengths, and individual medical conditions could lead to alterations on the obtained performance. One evidence of that is the absence of confirmed CA cases in the BUTPPG dataset, which could be present in external data. Furthermore, the small size of the dataset resulted in a reduced testing dataset, which makes the obtained results less general, that is, unreliable when we consider the possible variability of data that is external to the dataset. A small dataset size also implies that the deep learning models had less data samples to effectively learn. This contrasts with the usual treatment for deep learning, where usually large amounts of data feed the training of the model, allowing the proper adjustment of the large set of parameters. Therefore, our experiments would be more complete if our experiments tested on different and larger datasets.

Another constraint is that the experiments did not explore all available options in terms of models. For instance, our experiments left out some of the models of the Pytorch and the Aeon libraries. Examples of them are the GoogLeNet [120], from the Pytorch, and the Hydra Classifier [121], from the Aeon. Additionally, models external to these libraries, such as Xception [122] available in the Keras library [123], were not tested. Furthermore, not all variants of the tested models were evaluated, such as EfficientNet B7. Additionally, the hyperparameters of the projection methods, such as the number of

dimensions in RP, were not optimized. Exploring a broader range of options, including different libraries and hyperparameter search techniques like Optuna, could yield more comprehensive results.

Finally, additional limitations were identified at the implementation level. Firstly, the implementation utilized random oversampling to balance the dataset. However, alternative methods specifically designed for time series data, such as those described in [124], could have been employed. These methods not only balance the dataset but also can augment it. Secondly, resizing transforms were used to adapt projection images to model inputs, potentially leading to significant loss of information for matrix images that encode pixel relationships. As a consequence, the CV models probably did not performed as good as they could. Thirdly, there was no research for the early stop method that our implementation used. Consequently, there is a chance that this method prematurely stopped the training for models that required more epochs. Lastly, benchmarking metrics were measured using the Python standard API, which may be limited by the interpreter. Additionally, our measurements did not control the environment where measured the inference time. This could imply that external users have scheduled tasks that competed with mine measurements. Therefore, improvements at the implementation level could include applying time series augmentation techniques, resizing images without distortion by using integer multipliers and padding, researching and optimizing early stopping methods, and conducting measurements in a more controlled environment using low-level interfaces.

Chapter 5

Conclusion

This work presented a study on SQA for PPG signals. The investigated approach involved projecting 1D signals onto 2D images using the RP, GAF, and MTF methods. In addition to these methods, we proposed a mixed approach combining them. The results indicate that the RP and PMix projection methods outperformed the GAF and MTF methods, with RP and PMix yielding similar outcomes. Although the set of machine learning models was extensive, the BUTPPG dataset was small and unbalanced, limiting the conclusiveness of the results. Consequently, the experiment should be replicated on a larger dataset, either by using data augmentation techniques to balance and expand the BUTPPG dataset or by utilizing a different dataset with more samples.

The experiments in Chapter 4 provide insights into the importance of the novel projection method proposed in Chapter 3. One key finding is that the PMix method appeared more frequently among the best-performing results compared to the other projection methods. Specifically, in the projection-based ensembles listed in Table 4.30, all but one used the PMix method. This suggests that the proposed method can enhance the performance of a set of isolated projection methods. However, it should be noted that this method increases memory requirements due to the cumulative size of the individual projections. The same experiments also demonstrate the overall effectiveness of projection methods. Notably, in the best-performing models listed in Table 4.30, the PMix method combined with Wide ResNet outperformed the baseline time series classification models in terms of the Cohen Kappa score. This indicates that a projection-based approach can be highly accurate for both binary SQI classes. While this highlights the viability of the projection-based approach alongside other time series classifiers, the conventional time series classifiers achieved higher F1 and Precision scores, suggesting they performed better for the positive SQI class. An additional drawback is that combining projection-based methods with CV models is computationally more expensive and requires more memory than using 1D classifiers. Nevertheless, the projection-based approach proved to

be effective for SQA, fulfilling our original objective described in Section 1.1.

The results reveal the proposed method as a promising tool for real-life applications, as presented in Chapter 1. One could envision this thesis technique as a tool for Artificial Intelligence (AI) engineers, offering a trade-off between memory and computational cost in exchange for improved accuracy. This is achieved by combining various projection methods, which respectively increase image size and incur the 1D-to-2D conversion cost of each projection. For that reason, even though the CV models incorporated into the proposed method have sufficiently low latency to support a responsive application, they may not be advisable for wearable devices, such as smartwatches, due to memory constraints. It is necessary to process the signal on a remote device with greater memory capacity, such as a server in a remote healthcare environment. Therefore, this method is suitable for remote healthcare applications.

When considering the experimental results and the decisions that produced them, it is possible to understand the place of this work in the literature reviewed in Chapter 2. Regarding the proposed projection method, there is no known work suggesting this approach, which makes our work innovative. However, since our experiments did not directly compare the method with existing SQA approaches, the position of the PMix method in the literature remains uncertain. In addition to its originality, our experiments tested an unusually wide variety of 2D and 1D models, which is not common in the literature. This positions our work as a valuable reference for identifying models that synergize well with the SQA task, despite the limitation of testing on only a small dataset. Furthermore, in contrast to most works in the literature, our experiments can be reproduced. Our implementation uses open-source libraries for both the ML models and projection methods and works with a publicly available dataset with established labeling. Moreover, our software implementation is also publicly available[1]. Although it is not yet fully refined or thoroughly documented for external use, it is accessible for review. Therefore, this work is innovative, useful and reproducible, as the Section 1.2 exposed.

There are several improvement points, some already presented in the Section 4.3, for which future works could seek their corresponding solutions. For instance, we used only a single dataset, which is limited in size, data quality, variety, and recording length. Future work could involve experiments with larger and more diverse datasets, and cross-dataset validation would yield more reliable results. Conversely, the experiments did not explore many 1D and 2D models with open-source implementations, nor did they vary the parameters of the projections and ML models. Exploring these aspects could reveal new relationships among the models and their parameters. Increasing the model options, another improvement point is to test the proposed method against specialized

[1] `https://gitlab.com/lisa-unb/projection-based-biological-signal-processing`

methods from the SQA literature. This would assess the real relevance of the proposed method. Another idea related to the SQA literature would be to combine the proposed method with other existing SQA techniques, such as the signal multiscaling technique of Liu et al. [50]. That would possibly further increase performance of the PMix. Finally, the implementation could be further refined not only by selecting more effective pre-processing techniques and ML training strategies but also by improving the experimental setup and environment. Hence, there is significant potential for improvement in future work regarding both the experimental setup and the proposed method.

References

[1] Lucafó, Giovani Decico, Pedro Freitas, Rafael Lima, Gustavo da Luz, Ruan Bispo, Paula Rodrigues, Frank Cabello, and Otavio Penatti: *Signal quality assessment of photoplethysmogram signals using hybrid rule-and learning-based models.* Journal of Health Informatics, 15(Especial), 2023. xi, 3, 11

[2] Nemcova, Andrea, Enikö Vargova, Radovan Smisek, Lucie Marsanova, Lukas Smital, and Martin Vitek: *Brno university of technology smartphone ppg database (but ppg): Annotated dataset for ppg quality assessment and heart rate estimation.* BioMed Research International, 2021. `https://doi.org/10.1155/2021/3453007`. xi, 4, 5, 25

[3] Aouedi, Ons, Thai Hoc Vu, Alessio Sacco, Dinh C Nguyen, Kandaraj Piamrat, Guido Marchetto, and Quoc Viet Pham: *A survey on intelligent internet of things: applications, security, privacy, and future directions.* IEEE Communications Surveys & Tutorials, 2024. 1

[4] Weenen, Eva van: *Smart wearables in healthcare.* In *Dimensions of Intelligent Analytics for Smart Digital Health Solutions*, pages 23–61. Chapman and Hall/CRC, 2024. 1

[5] Stanford Vision Lab, Stanford University, Princeton University: *Imagenet.* `https://www.image-net.org/`, 2020. Accessed: May 18, 2024. 5, 27

[6] Rice, S. O.: *Mathematical analysis of random noise.* The Bell System Technical Journal, 23(3):282–332, 1944. 6

[7] Shannon, C. E.: *A mathematical theory of communication.* The Bell System Technical Journal, 27(3):379–423, 1948. 6

[8] Stehle, R.H.: *A double-sideband shortwave-broadcast signal-quality estimation algorithm.* IEEE Transactions on Broadcasting, 34(2):263–282, 1988. 6

[9] Wang, J.Y.: *A new method for evaluating ecg signal quality for multi-lead arrhythmia analysis.* In *Computers in Cardiology*, pages 85–88, 2002. 6

[10] Li, Q, R G Mark, and G D Clifford: *Robust heart rate estimation from multiple asynchronous noisy sources using signal quality indices and a kalman filter.* Physiological Measurement, 29(1):15, dec 2007. `https://dx.doi.org/10.1088/0967-3334/29/1/002`. 6

[11] Deshmane, Anagha Vishwas: *False arrhythmia alarm suppression using ecg, abp, and photoplethysmogram*. Master's thesis, Massachusetts Institute of Technology, 2009. 6

[12] Hjorth, Bo: *Eeg analysis based on time domain properties*. Electroencephalography and Clinical Neurophysiology, 29(3):306–310, 1970, ISSN 0013-4694. https://www.sciencedirect.com/science/article/pii/0013469470901434. 6

[13] Zhang, Pandeng, Jia Liu, Xinyu Wu, Xiaochang Liu, and Qingchun Gao: *A novel feature extraction method for signal quality assessment of arterial blood pressure for monitoring cerebral autoregulation*. In *2010 4th International Conference on Bioinformatics and Biomedical Engineering*, pages 1–4, 2010. 6

[14] Kaplan Berkaya, Selcan, Alper Kursat Uysal, Efnan Sora Gunal, Semih Ergin, Serkan Gunal, and M. Bilginer Gulmezoglu: *A survey on ecg analysis*. Biomedical Signal Processing and Control, 43:216–235, 2018, ISSN 1746-8094. https://www.sciencedirect.com/science/article/pii/S1746809418300636. 7

[15] Naseri, H. and M.R. Homaeinezhad: *Electrocardiogram signal quality assessment using an artificially reconstructed target lead*. Computer Methods in Biomechanics and Biomedical Engineering, 18(10):1126–1141, 2015. https://doi.org/10.1080/10255842.2013.875163, PMID: 24460414. 7

[16] Orphanidou, Christina and Ivana Drobnjak: *Quality assessment of ambulatory ecg using wavelet entropy of the hrv signal*. IEEE Journal of Biomedical and Health Informatics, 21(5):1216–1223, 2017. 7

[17] Shahriari, Yalda, Richard Fidler, Michele M. Pelter, Yong Bai, Andrea Villaroman, and Xiao Hu: *Electrocardiogram signal quality assessment based on structural image similarity metric*. IEEE Transactions on Biomedical Engineering, 65(4):745–753, 2018. 7

[18] Moeyersons, Jonathan, Elena Smets, John Morales, Amalia Villa, Walter De Raedt, Dries Testelmans, Bertien Buyse, Chris Van Hoof, Rik Willems, Sabine Van Huffel, and Carolina Varon: *Artefact detection and quality assessment of ambulatory ecg signals*. Computer Methods and Programs in Biomedicine, 182:105050, 2019, ISSN 0169-2607. https://www.sciencedirect.com/science/article/pii/S0169260719312817. 7

[19] Huerta Álvaro, Arturo Martinez-Rodrigo, Vicente Bertomeu-González, Óscar Ayo-Martin, José J. Rieta, and Raúl Alcaraz: *Single-lead electrocardiogram quality assessment in the context of paroxysmal atrial fibrillation through phase space plots*. Biomedical Signal Processing and Control, 91:105920, 2024, ISSN 1746-8094. https://www.sciencedirect.com/science/article/pii/S1746809423013538. 7

[20] Scardulla, Francesco, Gloria Cosoli, Susanna Spinsante, Angelica Poli, Grazia Iadarola, Riccardo Pernice, Alessandro Busacca, Salvatore Pasta, Lorenzo Scalise,

and Leonardo D'Acquisto: *Photoplethysmograhic sensors, potential and limitations: Is it time for regulation? a comprehensive review.* Measurement, 218:113150, 2023, ISSN 0263-2241. https://www.sciencedirect.com/science/article/pii/S0263224123007145. 7

[21] Li, Q and G D Clifford: *Dynamic time warping and machine learning for signal quality assessment of pulsatile signals.* Physiological Measurement, 33(9):1491, aug 2012. https://dx.doi.org/10.1088/0967-3334/33/9/1491. 7

[22] Papini, Gabriele B., Pedro Fonseca, Xavier L. Aubert, Sebastiaan Overeem, Jan W.M. Bergmans, and Rik Vullings: *Photoplethysmography beat detection and pulse morphology quality assessment for signal reliability estimation.* In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 117–120, 2017. 7

[23] Such, Olaf: *Motion tolerance in wearable sensors-the challenge of motion artifact.* In *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1542–1545. IEEE, 2007. 8

[24] Nabavi, Seyedfakhreddin and Sharmistha Bhadra: *A robust fusion method for motion artifacts reduction in photoplethysmography signal.* IEEE Transactions on Instrumentation and Measurement, 69(12):9599–9608, 2020. 8

[25] Tăuţan, Alexandra Maria, Alex Young, Eva Wentink, and Fokko Wieringa: *Characterization and reduction of motion artifacts in photoplethysmographic signals from a wrist-worn device.* In *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 6146–6149. IEEE, 2015. 8

[26] Zhang, Yifan, Shuang Song, Rik Vullings, Dwaipayan Biswas, Neide Simões-Capela, Nick Van Helleputte, Chris Van Hoof, and Willemijn Groenendaal: *Motion artifact reduction for wrist-worn photoplethysmograph sensors based on different wavelengths.* Sensors, 19(3):673, 2019. 8

[27] Ram, M Raghu, K Venu Madhav, E Hari Krishna, Nagarjuna Reddy Komalla, and K Ashoka Reddy: *A novel approach for motion artifact reduction in ppg signals based on as-lms adaptive filter.* IEEE Transactions on Instrumentation and Measurement, 61(5):1445–1457, 2011. 8

[28] Raghuram, M, Kosaraju Sivani, and K Ashoka Reddy: *Use of complex emd generated noise reference for adaptive reduction of motion artifacts from ppg signals.* In *2016 international conference on electrical, electronics, and optimization techniques (ICEEOT)*, pages 1816–1820. IEEE, 2016. 8

[29] Janiesch, Christian, Patrick Zschech, and Kai Heinrich: *Machine learning and deep learning.* Electronic Markets, 31(3):685–695, 2021. 8

[30] Mohagheghian, Fahimeh, Dong Han, Andrew Peitzsch, Nishat Nishita, Eric Ding, Emily L. Dickson, Danielle DiMezza, Edith M. Otabil, Kamran Noorishirazi, Jessica Scott, Darleen Lessard, Ziyue Wang, Cody Whitcomb, Khanh Van Tran, Timothy

P. Fitzgibbons, David D. McManus, and Ki H. Chon: *Optimized signal quality assessment for photoplethysmogram signals using feature selection.* IEEE Transactions on Biomedical Engineering, 69(9):2982–2993, 2022. 8

[31] Tiwari, Abhishek, Gordon Gray, Parker Bondi, Amin Mahnam, and Tiago H. Falk: *Automated multi-wavelength quality assessment of photoplethysmography signals using modulation spectrum shape features.* Sensors, 23(12), 2023, ISSN 1424-8220. `https://www.mdpi.com/1424-8220/23/12/5606`. 9

[32] Miranda, Jose A., Celia López-Ongil, and Javier Andreu-Perez: *Personalised and adjustable interval type-2 fuzzy-based ppg quality assessment for the edge.* In *2023 IEEE International Conference on Fuzzy Systems (FUZZ)*, pages 1–5, 2023. 9

[33] Roy, Monalisa Singha, Rajarshi Gupta, and Kaushik Das Sharma: *Photoplethysmogram signal quality evaluation by unsupervised learning approach.* In *2020 IEEE Applied Signal Processing Conference (ASPCON)*, pages 6–10, 2020. 9

[34] Mahmoudzadeh, Aysan, Iman Azimi, Amir M. Rahmani, and Pasi Liljeberg: *Lightweight photoplethysmography quality assessment for real-time iot-based health monitoring using unsupervised anomaly detection.* Procedia Computer Science, 184:140–147, 2021, ISSN 1877-0509. `https://www.sciencedirect.com/science/article/pii/S1877050921006499`, The 12th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 4th International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops. 9

[35] Feli, Mohammad, Iman Azimi, Arman Anzanpour, Amir M. Rahmani, and Pasi Liljeberg: *An energy-efficient semi-supervised approach for on-device photoplethysmogram signal quality assessment.* Smart Health, 28:100390, 2023, ISSN 2352-6483. `https://www.sciencedirect.com/science/article/pii/S2352648323000181`. 9

[36] Antzelevitch, Charles and Alexander Burashnikov: *Overview of basic mechanisms of cardiac arrhythmia.* Cardiac electrophysiology clinics, 3(1):23–45, 2011. 9

[37] Pereira, Tania, Kais Gadhoumi, Mitchell Ma, Rene Colorado, Kevin J Keenan, Karl Meisel, and Xiao Hu: *Robust assessment of photoplethysmogram signal quality in the presence of atrial fibrillation.* In *2018 Computing in Cardiology Conference (CinC)*, volume 45, pages 1–4, 2018. 9

[38] Pereira, Tania, Kais Gadhoumi, Mitchell Ma, Xiuyun Liu, Ran Xiao, Rene A. Colorado, Kevin J. Keenan, Karl Meisel, and Xiao Hu: *A supervised approach to robust photoplethysmography quality assessment.* IEEE Journal of Biomedical and Health Informatics, 24(3):649–657, 2020. 10

[39] Pereira, Tania, Cheng Ding, Kais Gadhoumi, Nate Tran, Rene A Colorado, Karl Meisel, and Xiao Hu: *Deep learning approaches for plethysmography signal quality assessment in the presence of atrial fibrillation.* Physiological Measurement, 40(12):125002, dec 2019. `https://dx.doi.org/10.1088/1361-6579/ab5b84`. 10

[40] Naeini, Emad Kasaeyan, Iman Azimi, Amir M. Rahmani, Pasi Liljeberg, and Nikil Dutt: *A real-time ppg quality assessment approach for healthcare internet-of-things*. Procedia Computer Science, 151:551–558, 2019, ISSN 1877-0509. https://www.sciencedirect.com/science/article/pii/S1877050919305368, The 10th International Conference on Ambient Systems, Networks and Technologies (ANT 2019) / The 2nd International Conference on Emerging Data and Industry 4.0 (EDI40 2019) / Affiliated Workshops. 10

[41] Zanelli, Serena, Mounim A. El Yacoubi, Magid Hallab, and Mehdi Ammi: *Transfer learning of cnn-based signal quality assessment from clinical to non-clinical ppg signals*. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 902–905, 2021. 10

[42] Gao, Haoyuan, Chao Zhang, Shengbing Pei, and Xiaopei Wu: *Lstm-based real-time signal quality assessment for blood volume pulse analysis*. Biomed. Opt. Express, 14(3):1119–1136, Mar 2023. https://opg.optica.org/boe/abstract.cfm?URI=boe-14-3-1119. 11

[43] Singha Roy, Monalisa, Biplab Roy, Rajarshi Gupta, and Kaushik Das Sharma: *On-device reliability assessment and prediction of missing photoplethysmographic data using deep neural networks*. IEEE Transactions on Biomedical Circuits and Systems, 14(6):1323–1332, 2020. 11

[44] Naeini, Emad Kasaeyan, Fatemeh Sarhaddi, Iman Azimi, Pasi Liljeberg, Nikil Dutt, and Amir M. Rahmani: *A deep learning–based ppg quality assessment approach for heart rate and heart rate variability*. ACM Trans. Comput. Healthcare, 4(4), nov 2023. https://doi.org/10.1145/3616019. 11

[45] Chen, Jianzhong, Ke Sun, Yi Sun, and Xinxin Li: *Signal quality assessment of ppg signals using stft time-frequency spectra and deep learning approaches*. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 1153–1156, 2021. 12

[46] Chatterjee, Tamaghno, Aayushman Ghosh, and Sayan Sarkar: *Signal quality assessment of photoplethysmogram signals using quantum pattern recognition technique and lightweight cnn module*. In *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 3382–3386, 2022. 12

[47] Roh, Donggeun and Hangsik Shin: *Recurrence plot and machine learning for signal quality assessment of photoplethysmogram in mobile environment*. Sensors, 21(6), 2021, ISSN 1424-8220. https://www.mdpi.com/1424-8220/21/6/2188. 12

[48] Freitas, Pedro Garcia, Rafael G. De Lima, Giovani D. Lucafo, and Otávio A.B. Penatti: *Photoplethysmogram signal quality assessment via 1d-to-2d projections and vision transformers*. In *2023 15th International Conference on Quality of Multimedia Experience (QoMEX)*, pages 165–170, 2023. 13

[49] Freitas, Pedro Garcia, Rafael G. De Lima, Giovani D. Lucafo, and Otávio A. B. Penatti: *Assessing the quality of photoplethysmograms via gramian angular fields and vision transformer*. In *2023 31st European Signal Processing Conference (EUSIPCO)*, pages 1035–1039, 2023. 13

[50] Liu, Jian, Shuaicong Hu, Ya'nan Wang, Qihan Hu, Daomiao Wang, and Cuiwei Yang: *A lightweight hybrid model using multiscale markov transition field for real-time quality assessment of photoplethysmography signals*. IEEE Journal of Biomedical and Health Informatics, 28(2):1078–1088, 2024. 13, 64

[51] Eckmann, Jean Pierre, S Oliffson Kamphorst, David Ruelle, *et al.*: *Recurrence plots of dynamical systems*. World Scientific Series on Nonlinear Science Series A, 16:441–446, 1995. 16

[52] Marwan, Norbert: *A historical review of recurrence plots*. The European Physical Journal Special Topics, 164(1):3–12, 2008. 16

[53] Afonso, Luis C.S., Gustavo H. Rosa, Clayton R. Pereira, Silke A.T. Weber, Christian Hook, Victor Hugo C. Albuquerque, and João P. Papa: *A recurrence plot-based approach for parkinson's disease identification*. Future Generation Computer Systems, 94:282–292, 2019, ISSN 0167-739X. https://www.sciencedirect.com/science/article/pii/S0167739X18322507. 16

[54] Shankar, Anand, Hnin Kay Khaing, Samarendra Dandapat, and Shovan Barma: *Analysis of epileptic seizures based on eeg using recurrence plot images and deep learning*. Biomedical Signal Processing and Control, 69:102854, 2021, ISSN 1746-8094. https://www.sciencedirect.com/science/article/pii/S1746809421004511. 16

[55] Zhao, Zhidong, Yang Zhang, Zafer Comert, and Yanjun Deng: *Computer-aided diagnosis system of fetal hypoxia incorporating recurrence plot with convolutional neural network*. Frontiers in Physiology, 10, 2019, ISSN 1664-042X. https://www.frontiersin.org/journals/physiology/articles/10.3389/fphys.2019.00255. 16

[56] Li, Xuemei, Tao Zhou, and Shi Qiu: *Alzheimer's disease analysis algorithm based on no-threshold recurrence plot convolution network*. Frontiers in Aging Neuroscience, 14, 2022, ISSN 1663-4365. https://www.frontiersin.org/journals/aging-neuroscience/articles/10.3389/fnagi.2022.888577. 16

[57] Mathunjwa, Bhekumuzi M., Yin Tsong Lin, Chien Hung Lin, Maysam F. Abbod, and Jiann Shing Shieh: *Ecg arrhythmia classification by using a recurrence plot and convolutional neural network*. Biomedical Signal Processing and Control, 64:102262, 2021, ISSN 1746-8094. https://www.sciencedirect.com/science/article/pii/S174680942030389X.

[58] Wang, Zhiguang and Tim Oates: *Encoding time series as images for visual inspection and classification using tiled convolutional neural networks*. In *Workshops at the twenty-ninth AAAI conference on artificial intelligence*, 2015. 18, 20

[59] Campanharo, Andriana S. L. O., M. Irmak Sirer, R. Dean Malmgren, Fernando M. Ramos, and Luís A. Nunes. Amaral: *Duality between time series and networks.* PLOS ONE, 6(8):1–13, August 2011. https://doi.org/10.1371/journal.pone.0023378. 20

[60] Team, Aeon: *Aeon.* https://www.aeon-toolkit.org/en/stable/. Accessed: May 18, 2024. 27

[61] Foundation, PyTorch: *Pytorch documentation.* https://pytorch.org/docs/stable/index.html, 2023. Accessed: May 18, 2024. 27

[62] Middlehurst, Matthew, James Large, Michael Flynn, Jason Lines, Aaron Bostrom, and Anthony Bagnall: *Hive-cote 2.0: a new meta ensemble for time series classification.* Mach. Learn., 110(11–12):3211–3243, dec 2021, ISSN 0885-6125. https://doi.org/10.1007/s10994-021-06057-9. 28

[63] Dempster, Angus, François Petitjean, and Geoffrey I. Webb: *ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels.* Data Min. Knowl. Discov., 34(5):1454–1495, 2020. https://doi.org/10.1007/s10618-020-00701-z. 28

[64] Zhao, Bendong, Huanzhang Lu, Shangfeng Chen, Junliang Liu, and Dongya Wu: *Convolutional neural networks for time series classification.* Journal of Systems Engineering and Electronics, 28(1):162–169, 2017. 28

[65] Wang, Zhiguang, Weizhong Yan, and Tim Oates: *Time series classification from scratch with deep neural networks: A strong baseline.* In *2017 International Joint Conference on Neural Networks, IJCNN 2017, Anchorage, AK, USA, May 14-19, 2017*, pages 1578–1585. IEEE, 2017. https://doi.org/10.1109/IJCNN.2017.7966039. 28

[66] Ismail Fawaz, Hassan, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F. Schmidt, Jonathan Weber, Geoffrey I. Webb, Lhassane Idoumghar, Pierre Alain Muller, and François Petitjean: *Inceptiontime: Finding alexnet for time series classification.* Data Min. Knowl. Discov., 34(6):1936–1962, nov 2020, ISSN 1384-5810. https://doi.org/10.1007/s10618-020-00710-y. 28

[67] Ismail-Fawaz, Ali, Maxime Devanne, Jonathan Weber, and Germain Forestier: *Deep learning for time series classification using new hand-crafted convolution filters.* In *2022 IEEE International Conference on Big Data (Big Data)*, pages 972–981, 2022. 28

[68] Ismail-Fawaz, Ali, Maxime Devanne, Stefano Berretti, Jonathan Weber, and Germain Forestier: *Lite: Light inception with boosting techniques for time series classification.* In *2023 IEEE 10th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10, 2023. 28

[69] Schäfer, Patrick: *The BOSS is concerned with time series classification in the presence of noise.* Data Min. Knowl. Discov., 29(6):1505–1530, 2015. https://doi.org/10.1007/s10618-014-0377-7. 28

[70] Middlehurst, Matthew, William Vickers, and Anthony Bagnall: *Scalable dictionary classifiers for time series classification.* In Yin, Hujun, David Camacho, Peter Tino, Antonio J. Tallón-Ballesteros, Ronaldo Menezes, and Richard Allmendinger (editors): *Intelligent Data Engineering and Automated Learning – IDEAL 2019*, pages 11–19, Cham, 2019. Springer International Publishing, ISBN 978-3-030-33607-3. 28

[71] Middlehurst, Matthew, James Large, Gavin Cawley, and Anthony Bagnall: *The temporal dictionary ensemble (tde) classifier for time series classification.* In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part I*, page 660–676, Berlin, Heidelberg, 2020. Springer-Verlag, ISBN 978-3-030-67657-5. https://doi.org/10.1007/978-3-030-67658-2_38. 28

[72] Schäfer, Patrick and Ulf Leser: *Multivariate time series classification with WEASEL+MUSE.* CoRR, abs/1711.11343, 2017. http://arxiv.org/abs/1711.11343. 28

[73] Schäfer, Patrick and Ulf Leser: *Fast and accurate time series classification with weasel.* In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 637–646, New York, NY, USA, 2017. Association for Computing Machinery, ISBN 9781450349185. https://doi.org/10.1145/3132847.3132980. 28

[74] Schäfer, Patrick and Ulf Leser: *WEASEL 2.0: a random dilated dictionary transform for fast, accurate and memory constrained time series classification.* Mach. Learn., 112(12):4763–4788, 2023. https://doi.org/10.1007/s10994-023-06395-w. 28

[75] Bennett, Luca A. and Zahraa S. Abdallah: *RED comets: An ensemble classifier for symbolically represented multivariate time series.* In Ifrim, Georgiana, Romain Tavenard, Anthony J. Bagnall, Patrick Schäfer, Simon Malinowski, Thomas Guyet, and Vincent Lemaire (editors): *Advanced Analytics and Learning on Temporal Data - 8th ECML PKDD Workshop, AALTD 2023, Turin, Italy, September 18-22, 2023, Revised Selected Papers*, volume 14343 of *Lecture Notes in Computer Science*, pages 76–91. Springer, 2023. https://doi.org/10.1007/978-3-031-49896-1_6. 28

[76] Abdallah, Zahraa S. and Mohamed Medhat Gaber: *Co-eye: a multi-resolution ensemble classifier for symbolically approximated time series.* Mach. Learn., 109(11):2029–2061, 2020. https://doi.org/10.1007/s10994-020-05887-3. 28

[77] Lines, Jason and Anthony J. Bagnall: *Time series classification with ensembles of elastic distance measures.* Data Min. Knowl. Discov., 29(3):565–592, 2015. https://doi.org/10.1007/s10618-014-0361-2. 28

[78] Zhao, Jiaping and Laurent Itti: *shapedtw: Shape dynamic time warping.* Pattern Recognition, 74:171–184, 2018, ISSN 0031-3203. https://www.sciencedirect.com/science/article/pii/S0031320317303710. 28

[79] Lubba, Carl Henning, Sarab S. Sethi, Philip Knaute, Simon R. Schultz, Ben D. Fulcher, and Nick S. Jones: *catch22: Canonical time-series characteristics - selected through highly comparative time-series analysis.* Data Min. Knowl. Discov., 33(6):1821–1852, 2019. https://doi.org/10.1007/s10618-019-00647-x. 28

[80] Christ, Maximilian, Nils Braun, Julius Neuffer, and Andreas W. Kempa-Liehr: *Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package).* Neurocomputing, 307:72–77, 2018, ISSN 0925-2312. https://www.sciencedirect.com/science/article/pii/S0925231218304843. 28

[81] Bagnall, Anthony, Michael Flynn, James Large, Jason Lines, and Matthew Middlehurst: *On the usage and performance of the hierarchical vote collective of transformation-based ensembles version 1.0 (hive-cote v1.0).* In *Advanced Analytics and Learning on Temporal Data: 5th ECML PKDD Workshop, AALTD 2020, Ghent, Belgium, September 18, 2020, Revised Selected Papers*, page 3–18, Berlin, Heidelberg, 2020. Springer-Verlag, ISBN 978-3-030-65741-3. https://doi.org/10.1007/978-3-030-65742-0_1. 28

[82] Middlehurst, Matthew, James Large, and Anthony Bagnall: *The canonical interval forest (cif) classifier for time series classification.* In *2020 IEEE International Conference on Big Data (Big Data)*, pages 188–195, 2020. 28

[83] Lines, Jason, Sarah Taylor, and Anthony Bagnall: *Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles.* ACM Trans. Knowl. Discov. Data, 12(5), jul 2018, ISSN 1556-4681. https://doi.org/10.1145/3182382. 28

[84] Cabello, Nestor, Elham Naghizade, Jianzhong Qi, and Lars Kulik: *Fast and accurate time series classification through supervised interval search.* In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 948–953, 2020. 28

[85] Deng, Houtao, George Runger, Eugene Tuv, and Martyanov Vladimir: *A time series forest for classification and feature extraction.* Information Sciences, 239:142–153, 2013, ISSN 0020-0255. https://www.sciencedirect.com/science/article/pii/S0020025513001473. 28

[86] Hills, Jon, Jason Lines, Edgaras Baranauskas, James Mapp, and Anthony J. Bagnall: *Classification of time series by shapelet transformation.* Data Min. Knowl. Discov., 28(4):851–881, 2014. https://doi.org/10.1007/s10618-013-0322-1. 28

[87] Bostrom, Aaron and Anthony J. Bagnall: *Binary shapelet transform for multiclass time series classification.* Trans. Large Scale Data Knowl. Centered Syst., 32:24–46, 2017. https://doi.org/10.1007/978-3-662-55608-5_2. 28

[88] Guillaume, Antoine, Christel Vrain, and Wael Elloumi: *Random dilated shapelet transform: A new approach for time series shapelets.* In El-Yacoubi, Mounim A., Eric Granger, Pong Chi Yuen, Umapada Pal, and Nicole Vincent (editors): *Pattern Recognition and Artificial Intelligence - Third International Conference, ICPRAI 2022, Paris, France, June 1-3, 2022, Proceedings, Part I*, volume 13363 of *Lecture*

Notes in Computer Science, pages 653–664. Springer, 2022. https://doi.org/10.1007/978-3-031-09037-0_53. 28

[89] Guillaume, Antoine, Christel Vrain, and Wael Elloumi: *Time series classification with Shapelets: Application to predictive maintenance on event logs. (Classification de séries temporelles avec les Shapelets : application à la maintenance prédictive via journaux d'événements).* PhD thesis, University of Orléans, France, 2023. https://tel.archives-ouvertes.fr/tel-04368849. 28

[90] Ayllón-Gavilán, Rafael, David Guijo-Rubio, Pedro Antonio Gutiérrez, and César Hervás-Martínez: *A dictionary-based approach to time series ordinal classification.* In *Advances in Computational Intelligence: 17th International Work-Conference on Artificial Neural Networks, IWANN 2023, Ponta Delgada, Portugal, June 19–21, 2023, Proceedings, Part II*, page 541–552, Berlin, Heidelberg, 2023. Springer-Verlag, ISBN 978-3-031-43077-0. https://doi.org/10.1007/978-3-031-43078-7_44. 28

[91] Deng, Houtao, George Runger, Eugene Tuv, and Martyanov Vladimir: *A time series forest for classification and feature extraction.* Information Sciences, 239:142–153, 2013, ISSN 0020-0255. https://www.sciencedirect.com/science/article/pii/S0020025513001473. 28

[92] Rodríguez, Juan José, Ludmila I. Kuncheva, and Carlos J. Alonso: *Rotation forest: A new classifier ensemble method.* IEEE Trans. Pattern Anal. Mach. Intell., 28(10):1619–1630, 2006. https://doi.org/10.1109/TPAMI.2006.211. 28

[93] Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby: *An image is worth 16x16 words: Transformers for image recognition at scale.* In *International Conference on Learning Representations*, 2021. https://openreview.net/forum?id=YicbFdNTTy. 29

[94] Tu, Zhengzhong, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li: *Maxvit: Multi-axis vision transformer.* In Avidan, Shai, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (editors): *Computer Vision – ECCV 2022*, pages 459–479, Cham, 2022. Springer Nature Switzerland, ISBN 978-3-031-20053-3. 29

[95] Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo: *Swin transformer: Hierarchical vision transformer using shifted windows.* In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002, 2021. 29

[96] Liu, Ze, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo: *Swin transformer v2: Scaling up capacity and resolution.* In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11999–12009, 2022. 29

[97] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun: *Deep residual learning for image recognition.* In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 29

[98] Xie, Saining, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He: *Aggregated residual transformations for deep neural networks.* In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017. 29

[99] Zagoruyko, Sergey and Nikos Komodakis: *Wide residual networks.* In Richard C. Wilson, Edwin R. Hancock and William A. P. Smith (editors): *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press, September 2016, ISBN 1-901725-59-6. `https://dx.doi.org/10.5244/C.30.87`. 29

[100] Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger: *Densely connected convolutional networks.* In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017. 29

[101] Simonyan, Karen and Andrew Zisserman: *Very deep convolutional networks for large-scale image recognition.* In Bengio, Yoshua and Yann LeCun (editors): *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. `http://arxiv.org/abs/1409.1556`. 29

[102] Iandola, Forrest N., Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer: *Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5MB model size*, 2017. `https://openreview.net/forum?id=S1xh5sYgx`. 29

[103] Tan, M., B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le: *Mnasnet: Platform-aware neural architecture search for mobile.* In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2815–2823, Los Alamitos, CA, USA, jun 2019. IEEE Computer Society. `https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00293`. 29

[104] Sandler, M., A. Howard, M. Zhu, A. Zhmoginov, and L. Chen: *Mobilenetv2: Inverted residuals and linear bottlenecks.* In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, Los Alamitos, CA, USA, jun 2018. IEEE Computer Society. `https://doi.ieeecomputersociety.org/10.1109/CVPR.2018.00474`. 29

[105] Howard, Andrew, Mark Sandler, Bo Chen, Weijun Wang, Liang Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, Yukun Zhu, Ruoming Pang, Hartwig Adam, and Quoc Le: *Searching for mobilenetv3.* In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019. 29

[106] Tan, Mingxing and Quoc V. Le: *Efficientnet: Rethinking model scaling for convolutional neural networks.* In Chaudhuri, Kamalika and Ruslan Salakhutdinov (editors): *Proceedings of the 36th International Conference on Machine Learning,*

ICML 2019, 9-15 June 2019, Long Beach, California, USA, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019. http://proceedings.mlr.press/v97/tan19a.html. 29

[107] Tan, Mingxing and Quoc V. Le: *Efficientnetv2: Smaller models and faster training.* In Meila, Marina and Tong Zhang (editors): *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 10096–10106. PMLR, 2021. http://proceedings.mlr.press/v139/tan21a.html. 29

[108] Ma, Ningning, Xiangyu Zhang, Hai Tao Zheng, and Jian Sun: *Shufflenet v2: Practical guidelines for efficient cnn architecture design.* In Ferrari, Vittorio, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (editors): *Computer Vision – ECCV 2018*, pages 122–138, Cham, 2018. Springer International Publishing, ISBN 978-3-030-01264-9. 29

[109] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton: *Imagenet classification with deep convolutional neural networks.* Commun. ACM, 60(6):84–90, may 2017, ISSN 0001-0782. https://doi.org/10.1145/3065386. 29

[110] Liu, Zhuang, Hanzi Mao, Chao Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie: *A convnet for the 2020s.* In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976, 2022. 29

[111] Radosavovic, Ilija, Raj Prateek Kosaraju, Ross B. Girshick, Kaiming He, and Piotr Dollár: *Designing network design spaces.* In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10425–10433. Computer Vision Foundation / IEEE, 2020. https://openaccess.thecvf.com/content_CVPR_2020/html/Radosavovic_Designing_Network_Design_Spaces_CVPR_2020_paper.html. 29, 52

[112] Contributors, Optuna: *Optuna: A hyperparameter optimization framework.* https://optuna.readthedocs.io/en/stable/, 2018. Accessed: May 18, 2024. 27

[113] Kingma, Diederik P. and Jimmy Ba: *Adam: A method for stochastic optimization.* In Bengio, Yoshua and Yann LeCun (editors): *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. http://arxiv.org/abs/1412.6980. 27

[114] developers, The imbalanced-learn: *imbalanced-learn documentation.* https://imbalanced-learn.org/stable/, 2024. Accessed: May 18, 2024. 31

[115] Faouzi, Johann and Hicham Janati: *pyts: A python package for time series classification.* Journal of Machine Learning Research, 21(46):1–6, 2020. http://jmlr.org/papers/v21/19-763.html. 31

[116] Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan

Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala: *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates Inc., Red Hook, NY, USA, 2019. 33

[117] developers scikit-learn: *scikit-learn*. `https://scikit-learn.org/`, 2024. Accessed: May 18, 2024. 33

[118] Jean Brouwers, Ludwig Haehne, Robert Schuppenies: *Pympler*. `https://pympler.readthedocs.io/en/latest/`, 2020. Accessed: May 18, 2024. 33

[119] Yang, Tien-Ju, Andrew G. Howard, Bo Chen, Xiao Zhang, Alec Go, Mark Sandler, Vivienne Sze, and Hartwig Adam: *Netadapt: Platform-aware neural network adaptation for mobile applications*. In Ferrari, Vittorio, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (editors): *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part X*, volume 11214 of *Lecture Notes in Computer Science*, pages 289–304. Springer, 2018. `https://doi.org/10.1007/978-3-030-01249-6_18`. 41

[120] Szegedy, C., Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich: *Going deeper with convolutions*. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, Los Alamitos, CA, USA, jun 2015. IEEE Computer Society. `https://doi.ieeecomputersociety.org/10.1109/CVPR.2015.7298594`. 60

[121] Dempster, Angus, Daniel F. Schmidt, and Geoffrey I. Webb: *Hydra: competing convolutional kernels for fast and accurate time series classification*. Data Min. Knowl. Discov., 37(5):1779–1805, may 2023, ISSN 1384-5810. `https://doi.org/10.1007/s10618-023-00939-3`. 60

[122] Chollet, F.: *Xception: Deep learning with depthwise separable convolutions*. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1800–1807, Los Alamitos, CA, USA, jul 2017. IEEE Computer Society. `https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.195`. 60

[123] Inc, Google: *Keras*. `https://keras.io/`. Accessed: May 18, 2024. 60

[124] Wen, Qingsong, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu: *Time series data augmentation for deep learning: A survey*. In Zhou, Zhi-Hua (editor): *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4653–4660. ijcai.org, 2021. `https://doi.org/10.24963/ijcai.2021/631`. 61

Appendices

Appendix A

Papers Resulting From This Thesis

- Guilherme Suzuki and Pedro Freitas. 2024. *On the Performance of Composite 1D-to-2D Projections for Signal Quality Assessment.* In Anais do XXIV Simpósio Brasileiro de Computação Aplicada à Saúde, junho 25, 2024, Goiânia/GO, Brasil. SBC, Porto Alegre, Brasil, 319-330. DOI: `https://doi.org/10.5753/sbcas.2024.2207`. (Qualis **A4**)

On the Performance of Composite 1D-to-2D Projections for Signal Quality Assessment

Guilherme Suzuki[1], **Pedro Garcia Freitas**[1]

[1] Department of Computer Science, University of Brasília, Brazil.

guilherme.chagas@aluno.unb.br, pedro.freitas@unb.br

Abstract. *Signal quality assessment is essential for health monitoring applications, as good signal quality is needed to reliably inform about the medical conditions of the patient. To achieve this, machine learning algorithms such as convolutional neural networks may be applied. However, the signal needs to be transformed into a 2D representation, which can be done using time series imaging techniques such as Gramian Angular Field (GAF), Markov Transition Field (MTF), and Recurrence Plot (RP), as well as by aggregating their results, which we refer to as Projection Mix. After preprocessing the dataset, Brno University of Technology Smartphone PPG (BUTPPG), into these images, various convolutional neural networks were trained and tested using such data, while also selecting hyperparameters through heuristic searching. The results indicate that our proposal performed better than the state-of-the-art methods.*

1. Introduction

Cardiovascular diseases such as coronary heart disease and stroke stand as a major global public health concern, claiming the lives of millions each year, as indicated by [Kaptoge et al. 2019]. In this context, the number of hypertensive patients will continue to increase worldwide, directly impacting any health system. In this scenario, the measurement of physiological parameters related to cardiac behavior, such as oximetry and heart rate, emerges as essential information that can be used for the control and prophylaxis of patients suffering from chronic diseases. While consistent monitoring of cardiovascular patterns is crucial for hypertensive individuals receiving home care or hospitalized patients, it necessitates precise and cost-effective equipment that prioritizes comfort. Currently, the gold standard method for conducting these measurements typically demands the expertise of a trained professional and provides an immediate assessment. However, it still exhibits various limitations attributed to its physical dimensions.

[Schmith et al. 2023] states that, considering public health concerns regarding cardiovascular diseases and practicality in clinical settings, numerous literature reports highlight automatic cuffless methods utilizing devices for measuring Photoplethysmogram (PPG) signals. With the advancement of Internet of Things (IoT) devices, wearable devices have become highly popular. They are now even smaller, smarter, more sensitive, scalable, and user-friendly. These devices can incorporate a wide range of sensing resources to continuously observe physiological functions. Among these sensing resources, PPG is a noteworthy sensor type since it can monitor various physiological parameters, such as heart rate, hypovolemia, blood oxygenation, respiration rate, and more. Moreover, PPG is worthy of attention because it is a simple optical method that consists of only two components placed on the skin. The first component is a light source used to

reflect light to the skin surface. The second component is a photodetector that collects the light reflection. As the PPG signal is composed of these optical components, it enables the continuous, non-invasive monitoring of physiological parameters. It is easy to implement, energy-efficient, and highly convenient for many clinical and commercial wearable devices.

Despite its undeniable advantages, PPG signals can be compromised by noises stemming from motion artifacts, signal acquisition, and other environmental sources. These noises have the potential to adversely affect signal quality, compromising the reliability of extracted health parameters. Such unreliable measurements could result in false alarms with potentially life-threatening consequences in healthcare applications. Consequently, Signal Quality Assessment (SQA) has emerged as an active research field in recent years. Several researchers have proposed techniques for evaluating PPG signals to analyze and optimize signal quality assessment performance.

In the literature, researchers investigated the PPG Signal Quality Index (SQI) by proposing assessment methods to discriminate 'reliable' and 'unreliable' parts of the signal. Among these methods, [Elgendi 2016] presents a comparison of eight evaluation metrics for signal quality based on perfusion, kurtosis, skewness, power, stationarity, zero crossing, entropy, and the matching of systolic wave detectors. It is a purely statistic-based approach. [Vadrevu and Manikandan 2019], [Reddy et al. 2020], and [Alam et al. 2021] described SQA using different signal features, also based on statistics, but considering additional constraints to enable the use of their methods in real-time applications on low-resource devices. Other studies presented machine learning approaches. For instance, [Li and Clifford 2012] employed a multi-layer perceptron to classify the quality status of PPG signals. [Pereira et al. 2019] investigated a set of machine learning approaches for quality assessment in 30-s segments of PPG, including k-nearest neighbors, decision trees, and support vector machines. Recently, deep learning methods based on 2D projections have emerged, converting the one-dimensional temporal-series raw PPG signal into bi-dimensional representations. [Naeini et al. 2023] employ one customized one-dimensional and three 2D Convolutional Neural Networks (CNN) to train models to assess PPG signals. In a similar fashion, [Freitas et al. 2023b, Freitas et al. 2023a] proposed a SQA method that converts PPG signals into two-dimensional representations, similar to 2D images. Subsequently, a vision transformer assesses the quality of these representations.

In this paper, we adopt a similar approach to the previously mentioned studies. Specifically, we extend the work of [Freitas et al. 2023a, Freitas et al. 2023b, Naeini et al. 2023] by incorporating all GAF, MTF, and RP projections within the same framework. Unlike previous works, our proposed approach combines these projection methods to create a composite, transforming them into a multi-channel (hyperspectral) image. In other words, we convert the temporal-series problem into a hyperspectral Computer Vision (CV) problem. To the best of our knowledge, no previous work has explored this composition of projections yet. Moreover, in contrast to the preceding papers, we consider the Computer Vision Classifier (CVC) as a hyperparameter of the overall framework. This means that, instead of using only Visual Transformer (ViT) as a classifier, we explore how different classifiers impact the performance of the overall framework, expanding the amount of possible CVC models that could be used for SQA.

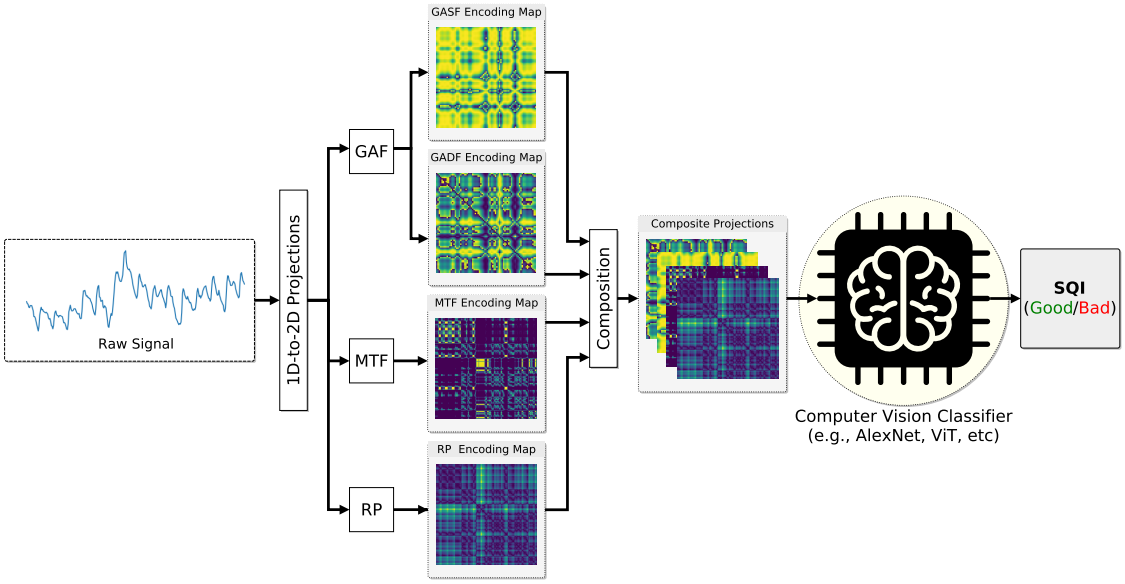


Figure 1. Block diagram of the proposed signal quality assessment pipeline. Different algorithms, namely GAF (with two variants, GASF and GADF), MTF, and RP project the one-dimensional raw signal onto two-dimensional image representations. Then, this hyperspectral image feeds a computer vision classifier. After training and evaluating the classifier, it can predict the signal quality indices (SQIs).

2. Proposed Method

Figure 1 depicts the summarized pipeline of the proposed method. This pipeline contains three main steps: (1) application of multiple 1D-to-2D signal projections; (2) hyperspectral image representation; and (3) image classification that employs a CVC to perform quality index estimation.

2.1. 1D-to-2D Projections

Mathematically, we can represent a PPG signal as a continuous function $X(t)$, where t is time. $X(t)$ represents the amplitude (intensity) of the PPG at any given point in time. In practice, sensors often sample the signal at discrete time points. Specifically, we may have a discrete representation $X[i]$, where i is the sample index. Thus, the PPG signal can be described as a time series $x = x_1, x_2, \cdots, x_n$ of n samples, where all values are in the interval $[-1, 1]$ and $x_i \in \mathbb{R}$. These one-dimensional temporal series can be projected onto a two-dimensional representation, such as GAF, MTF, and RP, offering advantages in visualization of temporal patterns, interpretability, applicability to computer vision techniques, and invariance to time warping.

Due to these advantages, in this paper, we combine three frameworks used to encode 1D PPG signals into 2D projections. The first type of projection, RP, represents a time series in a recurrence state in a phase space. It facilitates the representation of the phase-space trajectory in 2D, relying on its recurrence. The relationship between instants i and j is encoded in a 2D matrix, with elements assuming values of 0 or 1. The second type of projection, MTF, aims to generate a Markov matrix based on quantile bins, encoding transition probabilities within a quasi-Gramian Matrix. The third type of projection, GAF, represents time series data, such as a 1D signal, in a two-dimensional space by encoding

the pairwise angles between vectors in the original time series, visualized as an image.

2.1.1. Recurrence Plots

[Eckmann et al. 1987] introduced Recurrence Plots (RP) in 1987. In the original paper, the authors explored the use of recurrence as a tool for visualizing and analyzing the behavior of dynamical systems. Since then, RPs have become a valuable technique in the field of nonlinear time series analysis and have found applications in various scientific disciplines [Marwan 2008].

A RP is a projection that depicts the distances between trajectories extracted from the original time series. In a RP, the matrix elements correspond to the times at which a state of a dynamical system recurs. Specifically, columns and rows correspond to specific time pairs, capturing instances when the trajectory of the dynamical system traverses a region within the phase space that is approximately consistent.

Mathematically, for a signal x, the extracted trajectories are represented as $\vec{x}i = x_i, xi + \tau, \cdots, x_{i+\tau\cdot(m-1)}$, where $i \in 1, \cdots, n - (m-1)\cdot\tau$, n is the number of samples, m is the dimension of the trajectories, and τ is a time delay. The recurrence linking at instants i and j can be expressed by the following formula:

$$R_{ij} = \mathcal{H}(\varepsilon - \|\vec{x}_i - \vec{x}_j\|) = \begin{cases} 1 & \text{if } \|\vec{x}_i - \vec{x}_j\| \leq \varepsilon \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $\mathcal{H}$ is the Heaviside step function, ε is a recurrence threshold, and $\|\cdot\|$ is a norm. Applying this expression creates a chart R that represents $\vec{x}_i = \vec{x}_j$ on the horizontal and vertical axes, respectively. It encodes recurrences through a binary depiction, where $Rij = 1$ signifies the presence of recurrence, and $Rij = 0$ indicates its absence.

2.1.2. Markov Transition Field

[Campanharo et al. 2011] introduced the MTF model to encode the statistics of dynamical transitions, preserving sequential Markov transition probabilities to retain temporal information. For the PPG signal x, the MTF identifies m quantile bins and assigns each x_i to the corresponding bin q_j. Consequently, the MTF constructs a weighted adjacency matrix W of size $m \times m$ by tracking transitions among quantile bins, akin to a first-order Markov chain along the time axis. The coefficients w_{ij} denote the frequency with which a point in quantile q_i succeeds a point in quantile q_j. Following normalization, where $\sum_j w_{ij} = 1$, the result is the Markov transition matrix. This matrix is designed to be insensitive to the signal distribution and temporal dependencies on time steps t_i. To retain crucial temporal dependencies and prevent excessive information loss in matrix W, MTF is given by

$$M = \begin{bmatrix} w_{ij|x_1\in q_i, x_1\in q_j} & \cdots & w_{ij|x_1\in q_i, x_n\in q_j} \\ w_{ij|x_2\in q_i, x_1\in q_j} & \cdots & w_{ij|x_2\in q_i, x_n\in q_j} \\ \vdots & \ddots & \vdots \\ w_{ij|x_n\in q_i, x_1\in q_j} & \cdots & w_{ij|x_n\in q_i, x_n\in q_j} \end{bmatrix} \tag{2}$$

The construction of the $m \times m$ Markov transition matrix W involves partitioning the magnitude into m quantile bins. The symbols q_i and q_j denote the quantile bins corresponding to the magnitude at instants i and j, respectively. This implies that the element M_{ij} in MTF matrix represents the probability transition from q_i to q_j. To clarify, the transition probability along the magnitude axis in the matrix W is translated into the MTF matrix by accounting for temporal positions.

2.1.3. Gramian Angular Field

[Wang and Oates 2015] introduced GAF as a technique for converting time-series data onto image data to obtain spatial correlation on time-series signals. This technique encompasses both the Gramian Angular Summation Field (GASF) and the Gramian Angular Difference Field (GADF). GASF represents the summation of the cosine values of the pairwise angles formed by vectors in the time series data. It provides a 2D projection where each element of the matrix corresponds to the cosine of the angle between the corresponding time points in the original series. GASF emphasizes the global patterns and similarities in the time series data. The resulting GASF projection is symmetric and captures overall trends. On the other hand, GADF represents the absolute differences between the sine values of the pairwise angles, focusing on capturing local variations and changes in the time series and highlighting details in the time series that might indicate transitions or fluctuations, making it sensitive to changes in the dynamics of the system. Through the conversion of one-dimensional time-series data into two-dimensional image data, the resultant projection image preserves the temporal correlations inherent in the original data. This enables the application of deep representation learning techniques to conduct deep feature learning and classification on the transformed image data.

Mathematically, The method initially rescales the original time-series data range to $[-1, 1]$ using

$$\tilde{x_i} = \frac{(x_i - \max(X)) + (x_i - \min(X))}{\max(X) - \min(X)}. \tag{3}$$

Then, the scaled data is converted into a polar coordinate system as expressed by

$$\begin{cases} \phi_i = \arccos(\tilde{x_i}), & -1 \leq \tilde{x_i} \leq 1 \\ r = \dfrac{t_i}{N}, & t_i \in \mathbb{N}, \end{cases} \tag{4}$$

where t_i is the timestamp, r stands for the polar coordinate radius, and N is a regularization factor to stretch over the polar coordinate system. Since the data scaling range is $[-1, 1]$, then the transformed angle range $\phi_i \in [0, \pi]$. It means that this conversion is a bijection, implying that it produces a unique result in the polar coordinate system and, therefore, it has a distinctive inverse mapping. Subsequently, this conversion executes a process akin to the inner product operation in Cartesian coordinates. The relationships between time points are established by computing the angles (both sum and difference)

between distinct sample points as follows:

$$
\begin{aligned}
GASF &= \tilde{x}^{\intercal} \cdot \tilde{x} - \left(\sqrt{\mathbb{1} - \tilde{x}^2}\right)^{\intercal} \cdot \sqrt{\mathbb{1} - \tilde{x}^2} \\
&= \{\cos(\phi_i + \phi_j)\}_{i,j} \\
&= \begin{bmatrix} \cos(\phi_1 + \phi_1) & \cdots & \cos(\phi_1 + \phi_n) \\ \cos(\phi_2 + \phi_1) & \cdots & \cos(\phi_2 + \phi_n) \\ \vdots & \ddots & \vdots \\ \cos(\phi_n + \phi_1) & \cdots & \cos(\phi_n + \phi_n) \end{bmatrix},
\end{aligned} \tag{5}
$$

and

$$
\begin{aligned}
GADF &= \left(\sqrt{\mathbb{1} - \tilde{x}^2}\right)^{\intercal} \cdot \tilde{x} - \tilde{x}^{\intercal} \cdot \sqrt{\mathbb{1} - \tilde{x}^2} \\
&= \{\sin(\phi_i - \phi_j)\}_{i,j} \\
&= \begin{bmatrix} \sin(\phi_1 - \phi_1) & \cdots & \sin(\phi_1 - \phi_n) \\ \sin(\phi_2 - \phi_1) & \cdots & \sin(\phi_2 - \phi_n) \\ \vdots & \ddots & \vdots \\ \sin(\phi_n - \phi_1) & \cdots & \sin(\phi_n - \phi_n) \end{bmatrix},
\end{aligned} \tag{6}
$$

where $\mathbb{1} = [1, 1, \cdots, 1]^{\intercal}$ the unit row vector with the same length of $\tilde{x}$. Following the conversion of the temporal-series signal into the polar coordinate system, two categories of GAF can be established utilizing the inner products $\langle x, y\rangle = x \cdot y - \sqrt{1 - x^2} \cdot \sqrt{1 - y^2}$ and $\langle x, y\rangle = \sqrt{1 - x^2} \cdot y - x \cdot \sqrt{1 - y^2}$. These inner products are quasi-Gramian matrices due to the non-linearity of the defined functions $\langle x, y\rangle$ within the inner-product space.

2.2. Signal Quality Classification

Figure 2 depicts two examples of PPG signals and how these temporal-series signals relate to the bi-dimensional projections. In this figure, the first row contains the waveforms of an 'unreliable' and a 'reliable' signal. The second row depicts their corresponding 2D projections produced via RP, MTF, and GAF algorithms. From these pictures, it is possible to notice that both RP, MTF, and GAF projections produced from the 'unreliable' PPG signal present more irregularity and chaotic visual patterns (higher entropy). On the other hand, the 'reliable' signal generates regular and symmetric visual patterns. These RP, MTF, and GAF encoding maps illustrate how distinct the 'reliable' and 'unreliable' quality status of the PPG signals can be compared to their 2D projections.

The correlation among the encoding maps of RP, MTF, and GAF with the quality of PPG signals was noted by [Freitas et al. 2023a, Freitas et al. 2023b]. In these works, the authors investigated how each of these projections, used individually, performs in the task of signal quality classification when employed as input for a ViT classifier. Despite the promising results, the authors investigated the performance of each projection individually. However, the combination of these projections remained an open question. In this paper, we investigate how this combination affects the performance of different classifiers in discriminating the quality of 'reliable' or 'unreliable' signals. For this purpose, we computed the projections separately for each of the MTF, GAF, and RP algorithms. In the case of GAF, we considered both GASF and GADF variants, described, respectively, in Equations 5 and 6 of Section 2.1.3.

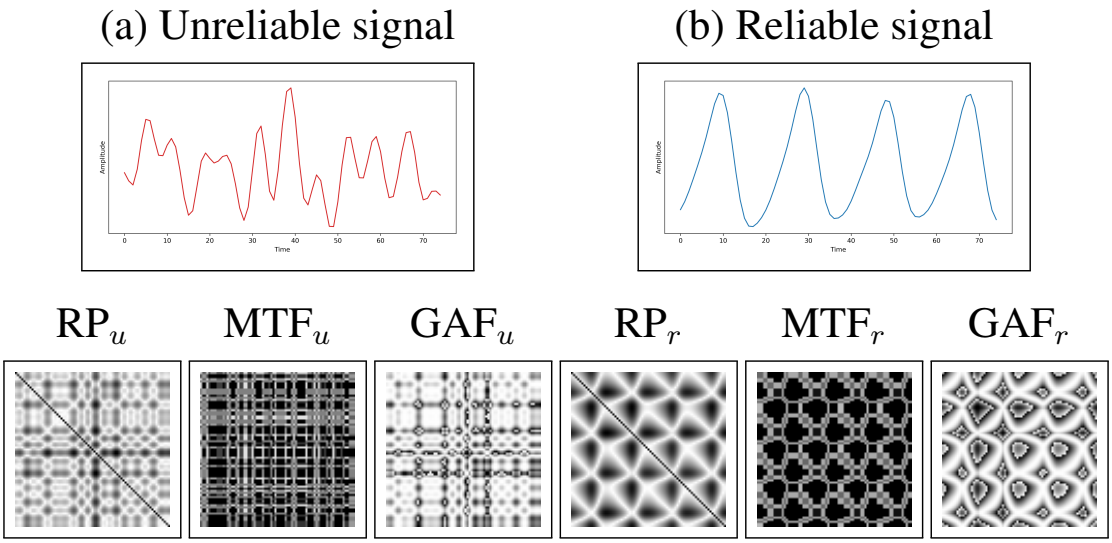


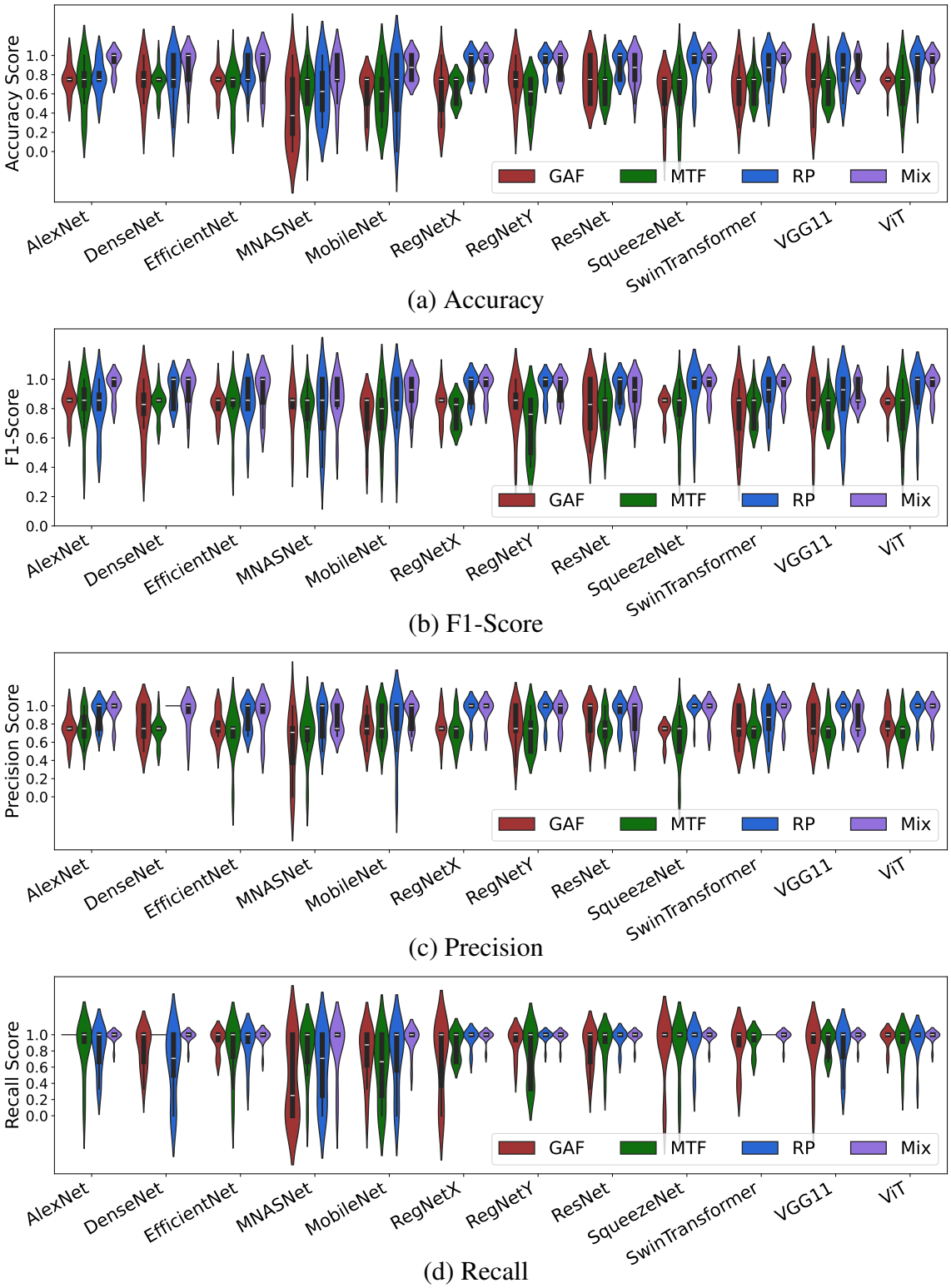
Figure 2. Example of signals and their corresponding 1D-to-2D encoding maps (projections). An 'unreliable' signal (a) generates asymmetrical encoding maps with less redundancy and higher visual variation, as we can observe from its RP_u, MTF_u, and GAF_u encoding maps. On the other hand, a 'reliable' signal (b) produces symmetric encoding maps with a more redundant pattern for all its RP_r, MTF_r, and GAF_r projections.

After generating the four independent projections (1 MTF, 1 RP, and 2 GAFs, one for GASF and one for GADF), the proposed method stacks those projections to compose a hyperspectral image (multi-channel). In other words, for a signal with n samples, the method transforms each of these four independent projections with dimensions $n \times n \times 1$ into a tensor of dimension $n \times n \times 4$. Then, we input these tensors into a CVC algorithm to predict the actual quality index (i.e., 'reliable' or not). In this work, we focus on deep-learning based CVCs, but other types could also be considered. To train the quality classifier, we map the input tensor to actual quality labels provided in the quality databases: $Q(PPG) = \mathfrak{C}(y, \mathcal{M})$, where $\mathfrak{C}(\cdot, \cdot)$ represents a CVCs algorithm, $\mathcal{M}$ is the trained model, and $Q(\cdot)$ is the quality score predicted using the model.

3. Experimental Results

We performed the tests using the BUTPPG database proposed by [Nemcova et al. 2021]. In this database, 48 PPG signals were recorded from the index fingers of 12 subjects. The recordings included three sessions while the subjects were seated and one session while they were walking [Nemcova et al. 2021]. This database can be accessed through the Physionet interface using the `wfdb` Python package. The PyTS library [Faouzi and Janati 2020] provided implementations for generating GAF, MTF, and RP representations. We used the PyTorch library [Paszke et al. 2019] for training and classification operations, focusing on classifying PPG signals. Additionally, we employed hyper-parameter optimization using the Optuna library [Akiba et al. 2019]. To balance the training dataset, we applied random oversampling using the imbalanced-learn library [Lemaître et al. 2017]. Furthermore, we utilized transfer learning by initializing the weights of neural networks pre-trained on the ImageNet dataset and fine-tuning them using the BUTPPG data.

The neural models chosen in our study were AlexNet, DenseNet, EfficientNet,



(a) Accuracy

(b) F1-Score

(c) Precision

(d) Recall

Figure 3. Violinplots illustrating the distribution of performance metrics for different models. Plots depict the variability scores across the model ensemble.

MNASNet, MobileNet, RegNetX, RegNetY, SqueezeNet, SwinTransformer, VGG11, and ViT. All these models were provided by TorchVision. Since the Projection Mix approach requires 4 channels in the input layer, we modified the models to meet this requirement.

Table 1. Performance Metrics for Neural Models with Different Projections. Considering each metric separately, the best results in terms of maximum absolute value are boldfaced, while the best results for each neural model contains an asterisk.

Neural model	Projection	Accuracy	F1 Score	Precision	Recall	Neural model	Projection	Accuracy	F1 Score	Precision	Recall
AlexNet	GAF	0.750	0.849	0.750	**1.000***	RegNetY	GAF	0.729	0.823	0.771	0.910
	MTF	0.708	0.827	0.773	0.833		MTF	0.583	0.690	0.729	0.757
	RP	0.771	0.819	0.910	0.812		RP	**0.938***	**0.955***	0.944*	0.979*
	Mix	**0.938***	**0.955***	0.944*	0.979		Mix	0.917	0.943	0.924	0.979*
DenseNet	GAF	0.729	0.795	0.799	0.819	ResNet	GAF	0.750	0.807	0.861	0.819
	MTF	0.729	0.837	0.729	**1.000***		MTF	0.708	0.799	0.771	0.889
	RP	0.771	0.917	**1.000***	0.646		RP	0.896*	0.926*	0.924*	0.951
	Mix	0.896*	0.932*	0.910	0.979		Mix	0.854	0.908	0.868	0.979*
EfficientNet	GAF	0.729	0.828	0.778	0.924	SqueezeNet	GAF	0.604	0.819	0.700	0.833
	MTF	0.667	0.806	0.694	0.812		MTF	0.625	0.794	0.646	0.861
	RP	0.812	0.856	0.882	0.889		RP	0.896	0.914	0.972*	0.903
	Mix	0.875*	0.916*	0.896*	0.972*		Mix	**0.938***	**0.955***	0.944	0.979*
MNASNet	GAF	0.438	0.812	0.552	0.458	SwinTransformer	GAF	0.667	0.765	0.771	0.847
	MTF	0.646	0.798	0.681	0.819		MTF	0.688	0.806	0.736	0.924
	RP	0.625	0.787	0.843	0.590		RP	0.833	0.897	0.833	**1.000***
	Mix	0.771*	0.866*	0.848*	0.861*		Mix	**0.938***	**0.955***	0.944*	0.979
MobileNet	GAF	0.625	0.758	0.765	0.743	VGG11	GAF	0.729	0.840	0.811	0.833
	MTF	0.604	0.777	0.806	0.583		MTF	0.667	0.790	0.729	0.896
	RP	0.667	0.838	0.818	0.715		RP	0.833*	0.855	0.944*	0.840
	Mix	0.875*	0.908*	0.910*	0.938*		Mix	0.833*	0.895*	0.840	0.979*
RegNetX	GAF	0.625	0.831	0.778	0.708	ViT	GAF	0.750	0.844	0.785	0.951
	MTF	0.646	0.773	0.729	0.868		MTF	0.688	0.790	0.729	0.889
	RP	0.917	0.938	0.944*	0.951		RP	0.896	0.913	0.944*	0.924
	Mix	**0.938***	**0.955***	0.944*	0.979*		Mix	**0.938***	**0.955***	0.944*	0.979*

Table 2. Memory usage of each model in terms of its parameters and buffers.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
AlexNet	228	MNASNet	3	RegNetY	15	SwinTransformer	110
DenseNet	27	MobileNet	16	ResNet	44	VGG11	515
EfficientNet	16	RegNetX	20	SqueezeNet	2	ViT	349

For most of them, we added a convolution layer with a kernel size of 1 and 4 input channels so that its output dimensions match those of the original first layer. In the case of RegNet, we adjusted the number of input channels in the stem layer. For the standard projection methods, we replicated the projection into the 3 input channels without adding extra layers.

For the purpose of testing those models, we separated the dataset using a k-fold cross-validation training-test strategy. This strategy divides the dataset into k sets of equal size. The model is then trained using $k-1$ of the folds and validated in the remaining kth fold. Figure 3 and Table 1 present the results of this approach. Figure 3 contains violin plots where the correlation score of each fold for a given metric is a point of the distribution. In that sense, if the shape of the violin is larger at a certain location and its length is shorter, it indicates consistent model performance across different measurements. Such behavior is very recurrent in the Mix method, which achieves results where most fold scores concentrate in the median of all folds in all metrics. Furthermore, for a given model and metric, that median often achieves higher scores than the other projection

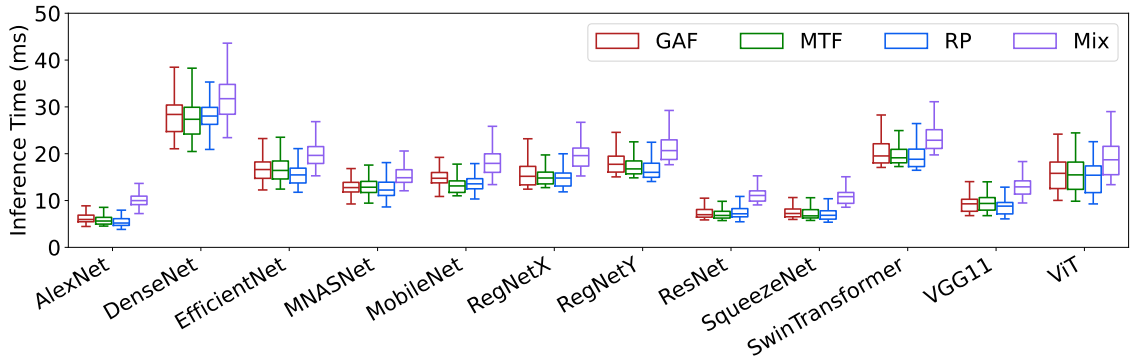


Figure 4. Boxplots where each point is the total inference time in milliseconds of each combination of neural model and projection method. The measured time includes the 1D-to-2D projection step.

methods, as seen in the accuracy score of AlexNet. These signs indicate that the proposed mix/composite method achieves higher performance. A visible exception occurs for the ResNet model, which achieves the best results with RP.

On the other hand, Table 1 reveals that, even though the PC mostly outperforms for the majority of the models, there are cases where other projections surpass it, such as for ResNet and RegNetY, where the Recurrence Plot obtains better results for most metrics. Additionally, the same table shows that the Mix method achieves the globally best scores for Accuracy and F1-Score, which contrasts with the fact that the other methods achieve the best Precision and Recall scores, but without achieving high scores in both metrics simultaneously.

Additionally, we measured the inference time and the memory consumption, as shown in Figure 4 and Table 2, respectively. Figure 4 was generated using 500 prediction/assessing time measurements. From this figure, it is noticeable that the speed of certain neural models stands out, such as the speed of AlexNet, ResNet, SqueezeNet, and VGG11. As for memory consumption, we see that some models consume a low amount of memory, such as MNASNet and SqueezeNet.

Considering all the information presented, a special case stands out: the combination of SqueezeNet and the Projection Composition (PC). In this case, Figure 3 shows that the PC is less scattered and has higher values than other approaches, showing a close competition with RP. However, Table 1 demonstrates that the PC has higher metric values than RP, except for Precision, where RP surpasses the PC, despite the PC having a value greater than 90%. Also, in Table 1, integrating SqueezeNet and PC gives the best average for the metrics Accuracy and F1 Score compared to all other combinations. Moreover, all metrics have values exceeding 90%. Finally, as seen before, this combination has high inference speed and low memory consumption.

The acceptance of these analyses needs to consider a series of limitations present in this procedure. Firstly, the dataset size is insufficient for training deep-learning models, which typically require large amounts of data. Secondly, we balanced the dataset using a simple technique, without exploring other alternatives specifically developed for time series data. Additionally, we did not explore the hyperparameters of the projections, such as the number of dimensions in the RP and the number of quantile bins in the MTF. Fur-

thermore, we only explored one of the multiple possible ways of constructing a composite projection from these individual projections.

4. Conclusion

This work presented a SQA method for PPG signals by projecting them into 2 dimensions using the RP, GAF, and MTF. In addition to these projection methods, we also proposed the use of a composition of them. Results indicate that the composition improves certain classification metrics and the generalization capabilities of the ensemble.

Acknowledgment

This work was supported by the Fundação de Apoio a Pesquisa do Distrito Federal (FAP-DF), by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), and by the University of Brasília (UnB). We thank Prof. Ricardo Lopes de Queiroz and Prof. Edson Mintsu Hung for letting us use their computing resources.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2623–2631, New York, NY, USA. Association for Computing Machinery.

Alam, S., Gupta, R., and Sharma, K. D. (2021). On-board signal quality assessment guided compression of photoplethysmogram for personal health monitoring. *IEEE Transactions on Instrumentation and Measurement*, 70:1–9.

Campanharo, A. S., Sirer, M. I., Malmgren, R. D., Ramos, F. M., and Amaral, L. A. N. (2011). Duality between time series and networks. *PloS one*, 6(8):e23378.

Eckmann, J.-P., Kamphorst, S. O., and Ruelle, D. (1987). Recurrence plots of dynamical systems. *Europhysics Letters*, 4(9):973.

Elgendi, M. (2016). Optimal signal quality index for photoplethysmogram signals. *Bioengineering*, 3(4):21.

Faouzi, J. and Janati, H. (2020). pyts: A python package for time series classification. *Journal of Machine Learning Research*, 21(46):1–6.

Freitas, P., Lima, R., Lucafo, G., and Penatti, O. (2023a). Photoplethysmogram signal quality assessment via 1d-to-2d projections and vision transformers.

Freitas, P. G., De Lima, R. G., Lucafo, G. D., and Penatti, O. A. B. (2023b). Assessing the quality of photoplethysmograms via gramian angular fields and vision transformer. In *2023 31st European Signal Processing Conference (EUSIPCO)*, pages 1035–1039.

Kaptoge, S., Pennells, L., De Bacquer, D., Cooney, M. T., Kavousi, M., Stevens, G., Riley, L. M., Savin, S., Khan, T., Altay, S., et al. (2019). World health organization cardiovascular disease risk charts: revised models to estimate risk in 21 global regions. *The Lancet global health*, 7(10):e1332–e1345.

Lemaître, G., Nogueira, F., and Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5.

Li, Q. and Clifford, G. D. (2012). Dynamic time warping and machine learning for signal quality assessment of pulsatile signals. *Physiological measurement*, 33(9):1491.

Marwan, N. (2008). A historical review of recurrence plots. *The European Physical Journal Special Topics*, 164(1):3–12.

Naeini, E. K., Sarhaddi, F., Azimi, I., Liljeberg, P., Dutt, N., and Rahmani, A. M. (2023). A deep learning–based ppg quality assessment approach for heart rate and heart rate variability. *ACM Transactions on Computing for Healthcare*, 4(4):1–22.

Nemcova, A., Vargova, E., Smisek, R., Marsanova, L., Smital, L., and Vitek, M. (2021). Brno university of technology smartphone ppg database (but ppg): Annotated dataset for ppg quality assessment and heart rate estimation. *BioMed Research International*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates Inc., Red Hook, NY, USA.

Pereira, T., Gadhoumi, K., Ma, M., Liu, X., Xiao, R., Colorado, R. A., Keenan, K. J., Meisel, K., and Hu, X. (2019). A supervised approach to robust photoplethysmography quality assessment. *IEEE journal of biomedical and health informatics*, 24(3):649–657.

Reddy, G. N. K., Manikandan, M. S., and Murty, N. N. (2020). On-device integrated ppg quality assessment and sensor disconnection/saturation detection system for iot health monitoring. *IEEE Transactions on Instrumentation and Measurement*, 69(9):6351–6361.

Schmith, J., Kelsch, C., Cunha, B. C., Prade, L. R., Martins, E. A., Keller, A. L., and de Figueiredo, R. M. (2023). Photoplethysmography signal quality assessment using attractor reconstruction analysis. *Biomedical Signal Processing and Control*, 86:105142.

Vadrevu, S. and Manikandan, M. S. (2019). Real-time ppg signal quality assessment system for improving battery life and false alarms. *IEEE transactions on circuits and systems II: express briefs*, 66(11):1910–1914.

Wang, Z. and Oates, T. (2015). Imaging time-series to improve classification and imputation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 3939–3945.

SBCAS2024
XXIV Simpósio Brasileiro de Computação Aplicada à Saúde

CERTIFICADO

Certificamos, para os fins que se fizerem necessários, que

Guilherme Chagas Suzuki

apresentou o trabalho intitulado

On the Performance of Composite 1D-to-2D Projections for Signal Quality Assessment

na **Trilha Principal** do XXIV Simpósio Brasileiro de Computação Aplicada à Saúde (SBCAS 2024), realizado em Goiânia-GO, Brasil, nos dias 25 a 28 de junho de 2024.

Prof. Dr. Sérgio Teixeira de Carvalho
Coordenador Geral do Evento

SBC Sociedade Brasileira de Computação
INF INSTITUTO DE INFORMÁTICA
UFG UNIVERSIDADE FEDERAL DE GOIÁS