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 Brazilian Federal Agency for Suport and Evaluation of Graduate Education (CAPES), the National Council for Scientific and Technological Development (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. Esses aplicativos costumam utilizar dispositivos vestíveis para captar sinais úteis no diagnóstico das condições de saúde de um indivíduo, como o fotopletismograma. O método de obtenção desse tipo de sinal, a fotopletismografia, é compacto, não-invasivo e econômico. Apesar desses benefícios, a fotopletismografia é particularmente suscetível a artefatos de movimento e interferências ambientais. Esses problemas podem deteriorar a qualidade do sinal, o que prejudica significativamente a eficácia dos aplicativos que o consomem. Portanto, avaliar a qualidade dos sinais é essencial nas aplicações de monitoramento de saúde. Para esse fim, 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 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*". Essas projeções foram combinadas com vários modelos de visão computacional. Então, esses modelos foram treinados e testados na base de dados de sinais fotopletismográficos de *smartphones* da Universidade de Brun, com hiperparâmetros selecionados através de busca heurística. Os resultados indicaram que o *Recurrence Plot* e o *Projection Mix* geralmente superaram as outras projeções usadas no estudo. Além disso, os métodos baseados em projeção alcançaram resultados comparáveis a 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% (remapeada de $[-1, 1]$ para $[0, 1]$) com um desvio padrão de 0,101.

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

Abstract

With the rapid rise in popularity of wearable devices like smartwatches, health monitoring applications are gaining traction. Those applications commonly utilize wearable devices to record signals that are useful in the individual's health condition diagnostic, such as the photoplethysmogram. That signal extraction method, the photoplethysmography, is compact, non-invasive, and economical. Despite those benefits, the photoplethysmography is particularly susceptible to motion artifacts and environmental interferences. Those issues can greatly impair quality of the signal, which compromises the performance of the applications that consume it. Therefore assessing the signal quality is essential for enabling health monitoring applications. 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 Plot, and by aggregating their results, which is referred to as "Projection Mix". We combined those projections with several computer vision models. Then, we trained and tested them on the Brno University of Technology smartphone PPG database, 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.

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

Contents

List of Figures

List of Tables

Acronyms

ABP arterial blood pressure

ACC accuracy

AF atrial fibrillation

AI artificial intelligence

BACC balanced accuracy

bpm beats per minute

BUTPPG Brno University of Technology smartphone PPG database

CA cardiac arrhythmia

CNN convolutional neural network

CV computer vision

DL deep learning

DTW dynamic time warping

ECG electrocardiogram

EDSS end diastole slope sum

FFNN feed-forward neural network

FN false negative

FP false positive

GADF Gramian Angular Difference Field

GAF Gramian Angular Field

GASF Gramian Angular Summation Field

HR heart rate

HRV heart rate variability

IBI inter-beat interval

IoT internet of things

KNN k-nearest neighbors

LED light-emitting diode

LOSO leave-one-subject-out

MaxViT Multi-axis Vision Transformer

MCC Matthew's correlation coefficient

ML machine learning

MLP multi-layer perceptron

MTF Markov Transition Field

PMix Projection Mix

PPG photoplethysmograph

RISEC random interval spectral ensemble classifier

ROC AUC area under receiver operating characteristics curve

RP Recurrence Plot

SESS slow ejection slope sum

SQA signal quality assessment

SQI signal quality index

SVM support vector machine

SwinT Shifted Windows Transformer

SwinTV2 Shifted Windows Transformer Version 2

TDE temporal dictionary ensemble

TN true negative

TP true positive

ViT Vision Transformer

Chapter 1

Introduction

The human civilization constantly seeks improvements in life longevity and quality. One of the main research areas in that regard is 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 strokes [1]. For that reason, it is important to periodically search for professional medical advice to detect diseases at early stages. Some diseases depend on early diagnosis to be even treatable, such as cardiac amyloidosis [2]. However, individual professional service is expensive for some people and is unresponsive to sudden changes when the patient is not in a hospital dependecy. 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. For example, certain ECG patterns can indicate a post-ischemic state [4]. Examples of such data are body temperature, blood pressure and neural activity. We can obtain those indicators in the patient's daily life through wearable devices that resemble quotidian objects, with the shape of belts, bracelets, rings, shoe soles, clothing, etc. [5]. A popular example of such wearables is the smartwatch that, similarly to the smartphone, can execute multiple applications, such as recording physiological data. Since devices like those support wireless connection, they can send that data 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. Photoplethysmograph (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 [6], 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 those reasons, 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 and photoplethysmograph. 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.

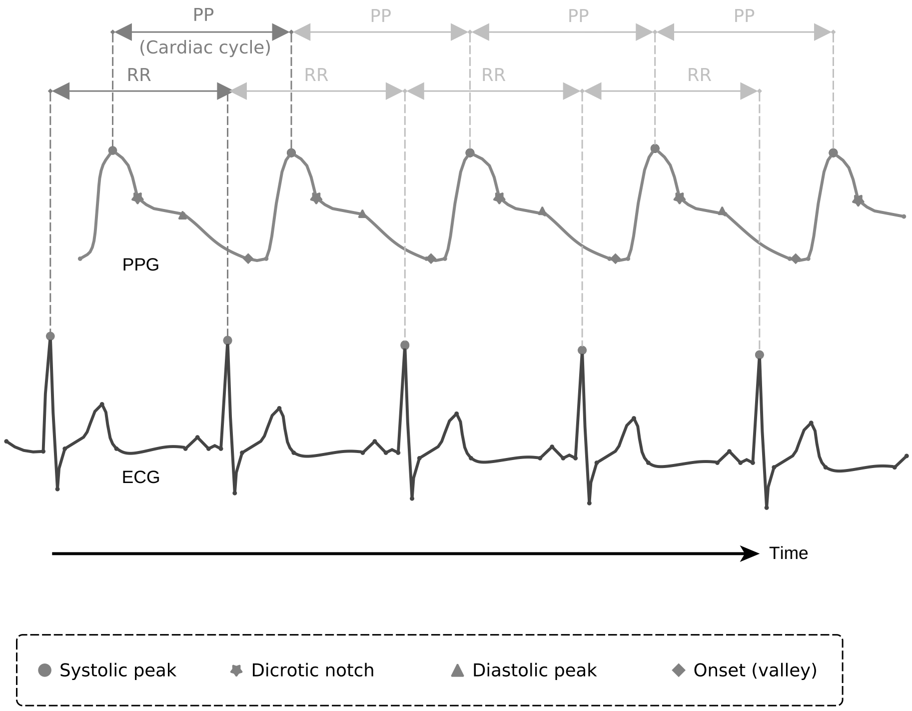


Figure 1.1: Inter-beat interval estimation using RR interval from electrocardiogram and the corresponding peak-to-peak interval from photoplethysmograph.

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 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, as illustrated in Figure 1.2. 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.

Typically, PPG signals are prone to degradation due to various factors, being motion artifact a relevant one among them [6]. 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. In that figure, the reliable

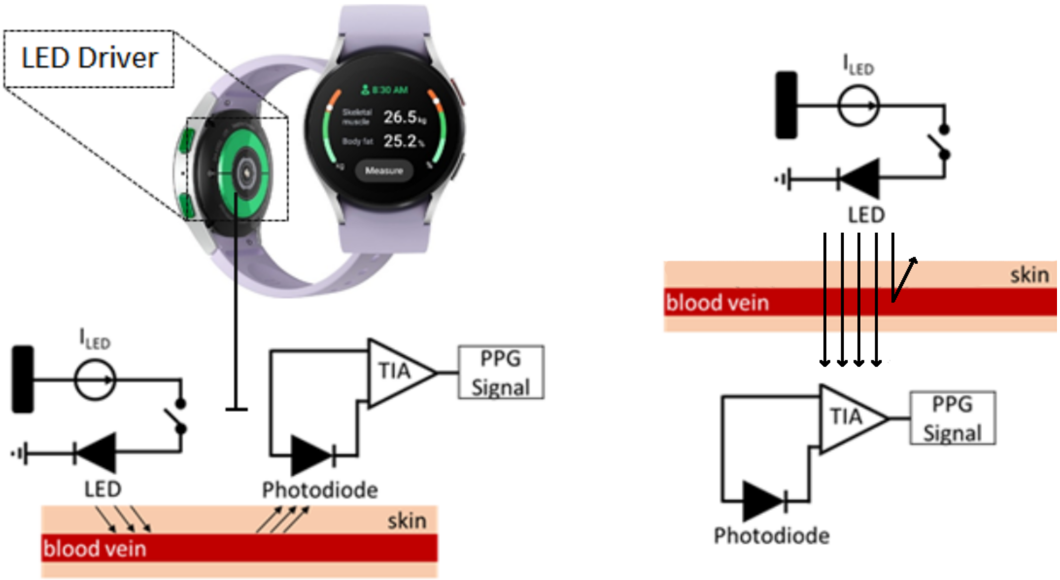


Figure 1.2: Schematic illustration of a photoplethysmograph sensor. On the left, the signal is obtained through the reflectivity of light by human tissue, while on the right, it is obtained through the permeability of light through human tissue. This figure is a courtesy of Lucafó et al. [7].

signals exhibits symmetrical, well-defined, and more consistent patterns. In contrast, the distorted signals present irregular, asymmetrical patterns with reduced consistency and greater variability between periods. Those 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.

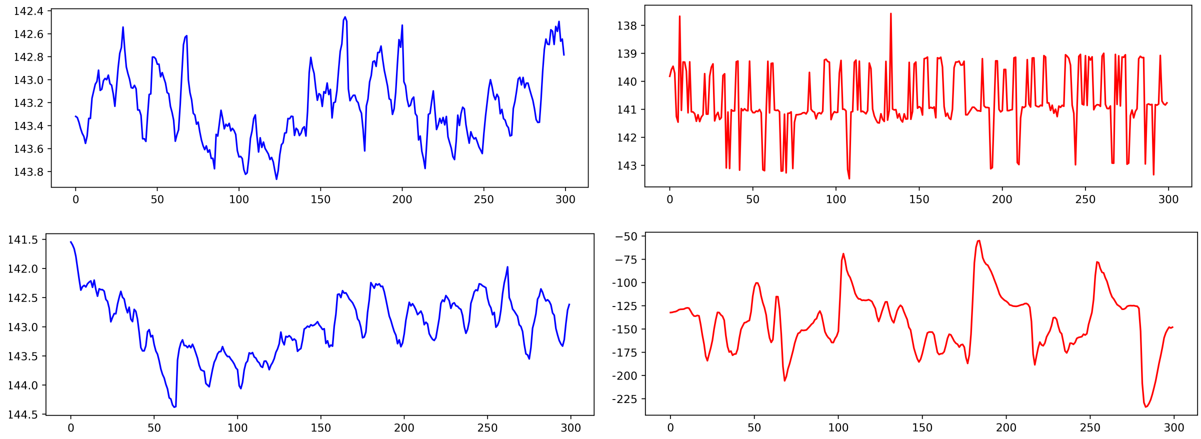


Figure 1.3: Example of reliable (in blue) and distorted (in red) photoplethysmograph signals. These signals belong to the Brno University of Technology smartphone PPG database [8].

1.2 Contributions of this work

This work elaborates a quality assessment framework for PPG signals which leads to 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 computer vision (CV) models, which previous works have not done. This evaluation provides insights into which types of models perform well with 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 [9]. Finally, the thesis reports experiments conducted on a publicly available labeled dataset, named Brno University of Technology smartphone PPG database (BUTPPG) [8], 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 five chapters, including this introduction. Chapter 2 contains an overview of the SQA ecosystem and its quality assessment methods. [blue]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 and works in the area of signal quality assessment (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 information theory in the 1920s and 1930s. One example of such foundational publications is the work of Rice [10], in which he analyzes the statistical properties of communication device noises. Another example is the work of Shannon [11], which introduces fundamental concepts in communication systems. In that millennium, studies already used the concept of SQA [12, 13]. We can see samples of this in the 1980s, such as the work of Stehle [12], which proposed an algorithm for assessing the quality of shortwave broadcast signals, trying to objectively measure the human perception of the signal message intelligibility. 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 had several works in the subject. One of them is the work of Wang et al. [14], which proposed a ECG SQA method based on the difference between the areas of distinct QRS complexes. The work proposed comparing the cumulative histograms of different ECG leads to assess its qualities. Later, Li et al. [15] suggested the combination of multiple quality indices and heart rate (HR) to assess ECG. Another work, by Deshmane et al. [16], posed the thresholding based on the Hjorth parameters [17] to assess the quality of PPG signals. Afterwards, Zhang et al. [18] elaborated an arterial blood pressure (ABP) signal quality assessment metric based on the end diastole slope sum (EDSS) and slow ejection slope sum (SESS) features. Through those works, the researchers proposed quantifying the SQA in a metric [15, 16, 18]. A name that they used to refer to this metric was signal quality index (SQI) [15, 16].

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 [19]. There are plenty of works in the SQA of ECG signals. In one of them, Naseri et al. [20] 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. Therefore, the ECG is present in the SQA literature.

Also, there are other publications on the ECG SQA. For instance, in 2017, Orphanidou et al. [21] introduced a feature based on the extraction of the heart rate variability (HRV) 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 feeds a support vector machine (SVM), which classifies the signal as acceptable or not. Later, Shahriari et al. [22] 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. [23] proposed transforming the signal using the auto correlation function and extracting simple features from it. In 2024, Huerta et al. [24] 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 ranges multiple works.

Besides the relevance of the ECG, the PPG has increased in popularity as an alternative to it. In fact, its number of related articles published from 2013 to 2023 has increased by 176% [6]. However, as previously presented in Chapter 1, the PPG is also susceptible to noise, fact that led to the development of PPG SQA methods. A sample of this literature is the work of Li et al. [25], 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. [26], 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 DTW 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 [27], 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 [28, 29] and optical source-detector pairs with peak responses beyond the red-infrared wavelength spectrum [30]. Some studies generate reference noise signals internally from the impaired PPG segments, reducing the need for extra hardware [31, 32]. 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 [33].

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 allows the machine learning model to have access to more information in the learning process, but at the cost of more training computational expense. Multiple works in the SQA literature are supervised. Per example, Mohagheghian et al. [34] 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 neighbors (KNN) and discriminant analysis. Among those predictors, the AdaBoost presented the best performance in terms of area under receiver operating characteristics curve (ROC AUC) 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. [35] 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. [36], 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 of specialists guidance and its biases. As opposed to most works in the SQA literature, some works present unsupervised methods. For example, Singha et al. [37] propose extracting entropy and statistical features from the input signal and feeding them to a self-organizing map. Experiments on a private dataset resulted in 92.01% accuracy in ternary classification. As another example, Mahmoudzadeh et al. [38] proposed extracting features from the time and frequency domains and feeding them to a elliptical envelope algorithm. 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. [39], 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 [40]. 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 rejected as unreliable signals, leaving patients with the CA condition misdiagnosed. Pereira et al. [41] 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. [42] fed several features to three classifiers: SVM, KNN and decision tree. Similarly to their previous 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. [43], 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) have demonstrated the potential to achieve a higher accuracy when compared with feature-based models, even in the presence of CA [42]. On one hand, differently of hand-crafted features, DL automatically extracts 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 properly adjust the model parameters. Despite this, DL is worth exploring since it can provide the accuracy and robustness that the medical applications require.

In this context, several studies proposed the application convolutional neural networks (CNNs) receiving the one-dimensional input signal. For instance, Naeini et al. [44] designed a 1D 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. [45] employed a self-made 1D 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 1D CNNs can achieve high accuracy in SQA but also it is possible to transfer its learned features over different databases to improve its performance.

In a different light, some works go beyond the simple application of such CNNs. The research of Lucafo et al. [7], per example, introduced a hybrid-model for quality assessment, which combines a 1D CNN with a rule-based approach. This rule can bypass the utilization of the CNN by verifying if the min-to-max distance of the signal 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, combining the 1D CNN with other methods can achieve particular advantages.

Besides the 1D CNNs mentioned at this point, 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. [46], 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 light-weighted 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 seen in the work of Singha et al. [47]. 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. [48]. Even though three 2D CNNs achieved above 90% accuracy, F1-Score, and ROC AUC 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. Figure 2.1 shows some of those transformations. Some works used 2D CNNs as a classifier of the SQI. One of those, Chen et al. [49], proposed the construction of a short-time Fourier transform spectrogram as the signal transformation method. The researchers annotated the VitalDB database[1] [50] 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. [51], 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[2] [52] with labels from the researchers, scoring above baseline models and an existing DL method. Roh et al. [53] embedded the signal into a Recurrence Plot (RP) matrix for PPG SQA. On a private dataset, the method achieved 97.5% accuracy. Based on these results, it is noticeable that the application of time series image proved to be effective in the SQA literature.

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. [58] fed a Vision Transformer with a RP or a Markov Transition Field (MTF), achieving, respectively, 89.9% and 90.3% accuracy on a private dataset. Freitas et al. [59] also fed images with a Vision Transformer, but used GAFs. The proposed approach reached 92.2% accuracy on a private dataset. Liu et al. [56] proposed to input Multiscale Markov Transition Fields, MTF version which concatenates the signal first and second derivatives, to 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 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.

[1] Accessible at `https://osf.io/uq2p2/` or at `https://vitaldb.net/dataset/`.

[2] Accessible at `http://dx.doi.org/102.100.100/6914`.

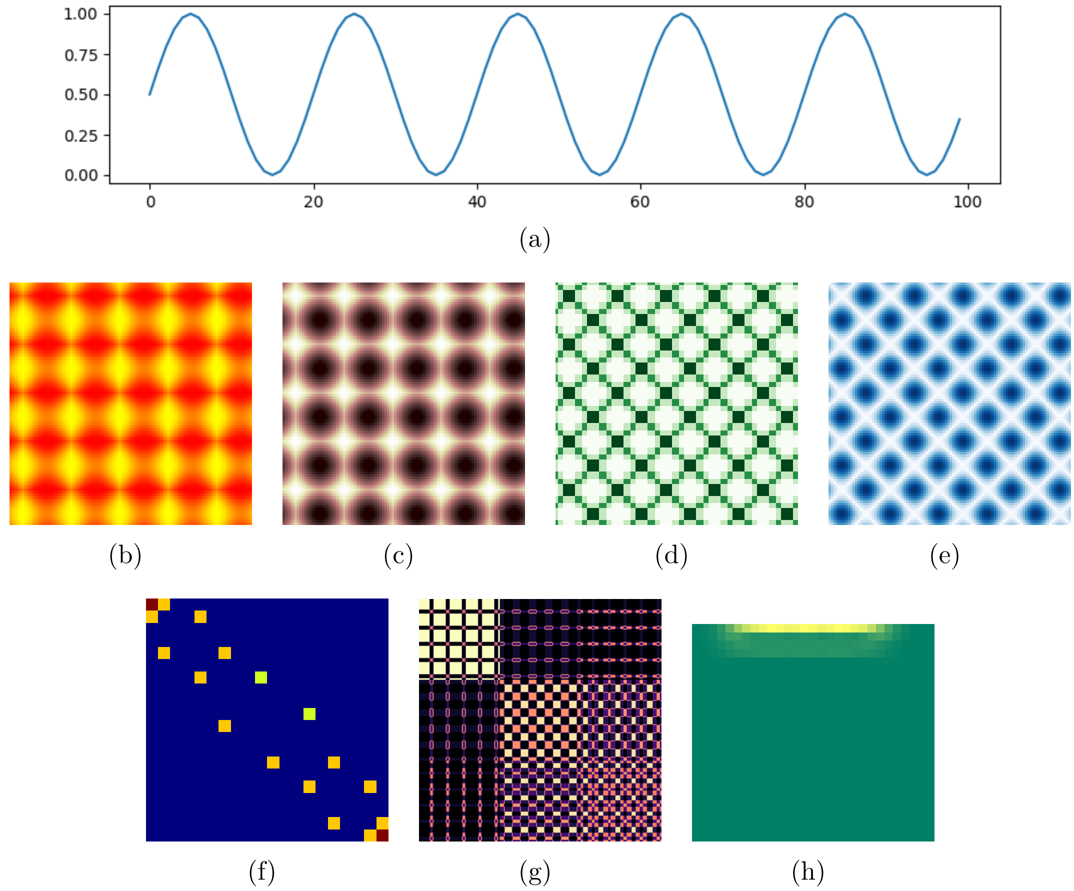


Figure 2.1: A signal (a) and its various projection obtained by several methods. They are: (b) Gramian Angular Diference Field [54], (c) Gramian Angular Summation Field [54], (d) Markov Transition Field [54] and (e) Recurrence Plot [55], (f) Poincaré Plot Density Map [24], (g) Multiscale Markov Transition Field [56] and (h) Short Time Fourier Transform Spectogram [57, 49].

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, none of the previous studies presented in this chapter have 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 used 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 a standardized series of experiments to test and validate the

2D DL approach against 1D ML methods. This thesis describes our efforts to fill that gap by experimenting with various existing methods for time series, aiming to conduct a comprehensive study of the field.

Chapter 3

The projection-based approach

As Chapter 2 presented, several works in the SQA literature employed the time series imaging method as a manner to allow CV models to be used in the SQI estimation. Figure 3.1 presents a generic framework of such method, which we qualify as "projection-based". It first converts the 1D signal into a 2D projection and afterwards feeds it to a CV model. Finally the CV estimates the SQI. The framework can use several projection algorithms, but we restrain our scope to four time series embedding methods presented by this chapter: Gramian Angular Field (GAF), which generates the variants Gramian Angular Summation Field (GASF) and Gramian Angular Difference Field (GADF); Markov Transition Field (MTF); and Recurrence Plot (RP). Moreover, this chapter introduces a new projection algorithm which we call Projection Mix (PMix).

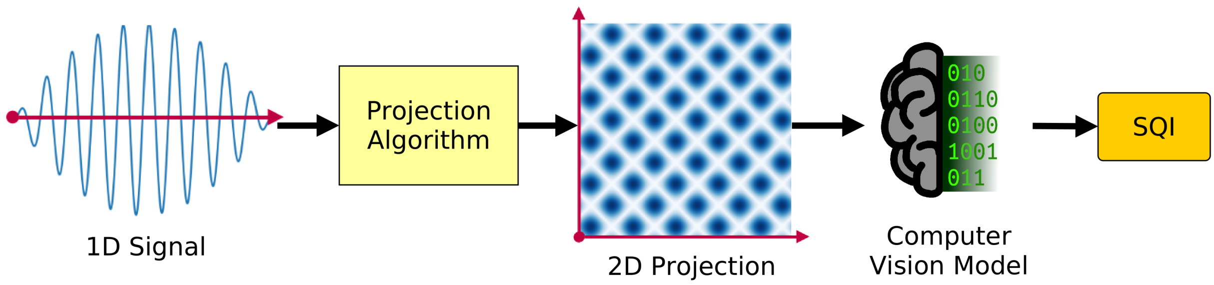


Figure 3.1: The time series imaging (or projection-based) approach.

3.1 Existing projection methods

This section provides an analysis of the existing 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.

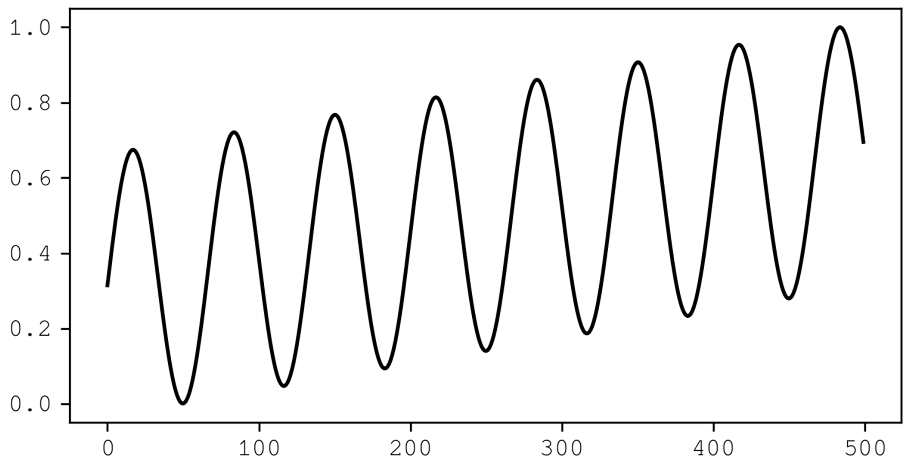


Figure 3.2: The example signal, with function $f(t) = \sin(\frac{15\pi t}{500}) + \frac{t}{500}$. The t variable correspond to the time instant, while the $f(t)$ function gives the magnitude of the signal.

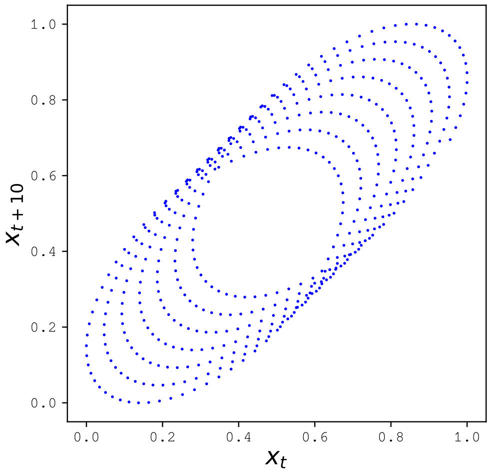
3.1.1 Recurrence Plot

The work of Eckmann et al. [55], 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 [60]. It is also applicable in life sciences, with attempts on identifying the presence of Parkinson's disease [61], epileptic seizure [62], fetal hypoxia [63], and Alzheimer's disease [64]. 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 length 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. Figure 3.3 depicts the phase space of the example signal of Figure 3.2.



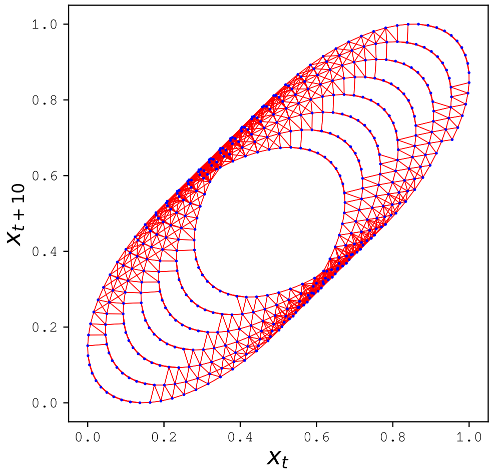


Figure 3.3: On the left, we have the signal of the figure 3.2 on the time delay phase space, without temporal information. Its parameters are dimension $d = 2$ and delay $\tau = 10$. On the right, we have almost the same figure, but with the recurrences represented by red lines $\vec{s_i} - \vec{s_j}$ 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 absence 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:

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

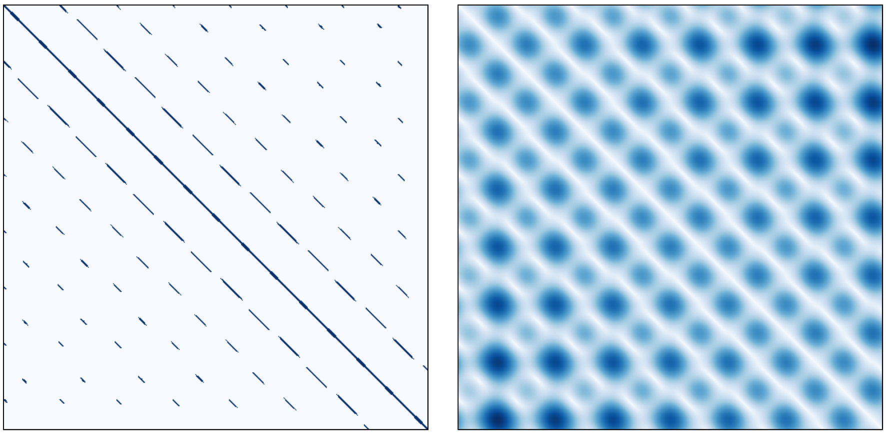
where $\mathcal{H} : \mathbb{R} \mapsto \{0, 1\}$ is the Heaviside function. Alternativelly, we can produce an unthresholded version $RP'_{m \times m}$ by attributing to each cell $RP'_{i,j} \in \mathbb{R}$ the points distance:

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

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

3.1.2 Gramian Angular Field

The work of Oates et al. [54] 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



(a) Thresholded (b) Unthresholded

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 (a) we have the thresholded version, while on the right (b) we have the unthresholded version.

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:

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

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

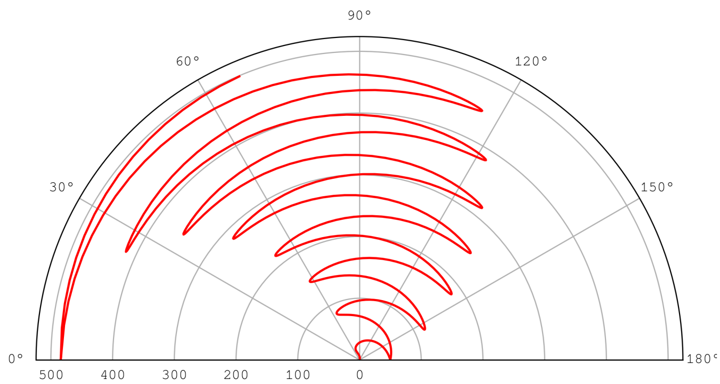


Figure 3.5: The example signal in its polar coordinate shape, with $N = 1$.

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

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

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 3.5. Thus, we can calculate GASF using matrix operations, as follows:

$$GASF = X^T \cdot X - \sqrt[\circ 2]{\mathbb{1} - X^{\circ 2}}^T \cdot \sqrt[\circ 2]{\mathbb{1} - X^{\circ 2}}, \tag{3.6}$$

where $M^{\circ 2}$ and $\sqrt[\circ 2]{M}$ represents the element-wise square power and square root of the matrix M, respectively, 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 $GADF_{n\times n}$:

$$\begin{aligned} GADF_{i,j} &= \sin(\phi_i - \phi_j) \\ &= \sin(\phi_i) \cdot \cos\phi_j - \cos(\phi_i) \cdot \sin(\phi_j) \\ &= \sqrt{1 - x_i^2} \cdot x_j - x_i \cdot \sqrt{1 - x_j^2}, \end{aligned} \tag{3.7}$$

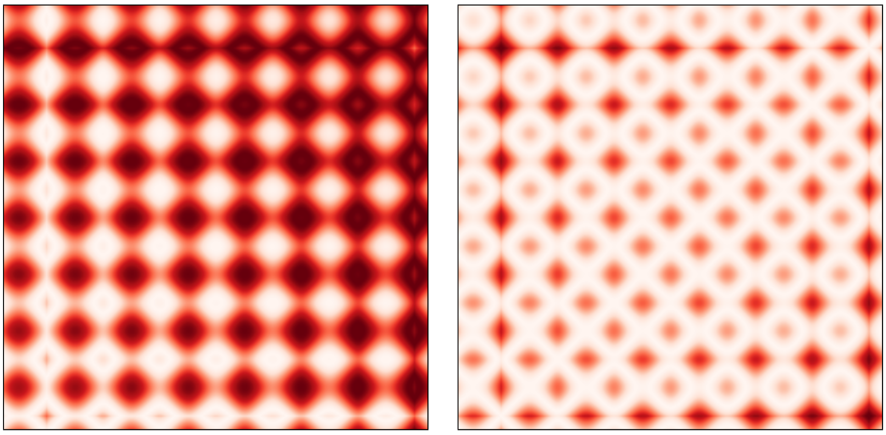
Also similarly, by the equality 3.7, we can express GADF by matrix operations:

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

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

3.1.3 Markov Transition Field

Oates et al. [54] proposed the MTF as well, based on a signal to graph mapping of Campanharo et al. [65]. 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 amount of samples x_i. Figure 3.7 pictures this concept for the example signal. The other graph component, its edges, are directed



(a) GADF (b) GASF

Figure 3.6: The example signal corresponding Gramian Angular Difference Field (a) and Gramian Angular Summation Field (b).

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:

$$W_{i,j} = \frac{\sum\limits_{x_k \in Q_i, x_{k+1} \in X} f_{in}(x_{k+1}, Q_j)}{\sum\limits_{Q_l \in Q} \sum\limits_{x_k \in Q_i, x_{k+1} \in X} f_{in}(x_{k+1}, Q_l)}, \tag{3.9}$$

where

$$f_{in}(x, Q) = \begin{cases} 0, & x \notin Q \\ 1, & x \in Q \end{cases}. \tag{3.10}$$

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 choosing a sample of the current quantile node and then transitioning to the next node according to the transitions probabilities. Algorithm 1 explicits that representation and 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 successful recovery is not certain.

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

$$MTF_{i,j} = W_{u,v} | x_i \in Q_u, x_j \in Q_v, \tag{3.11}$$

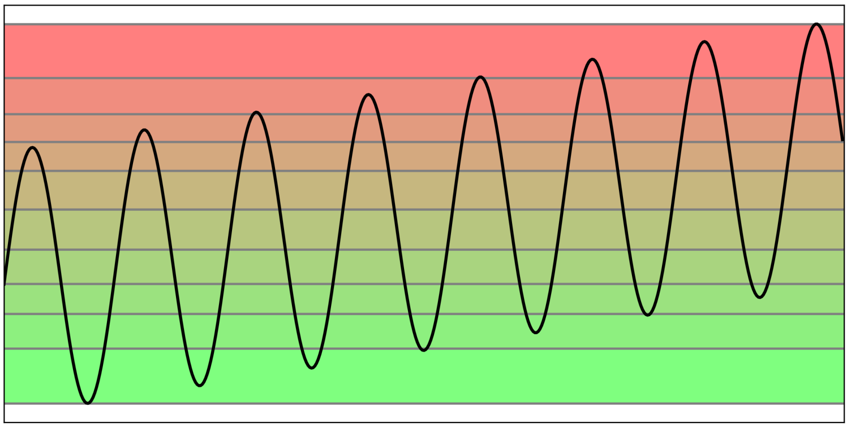


Figure 3.7: Original signal segmented into quantile bins.

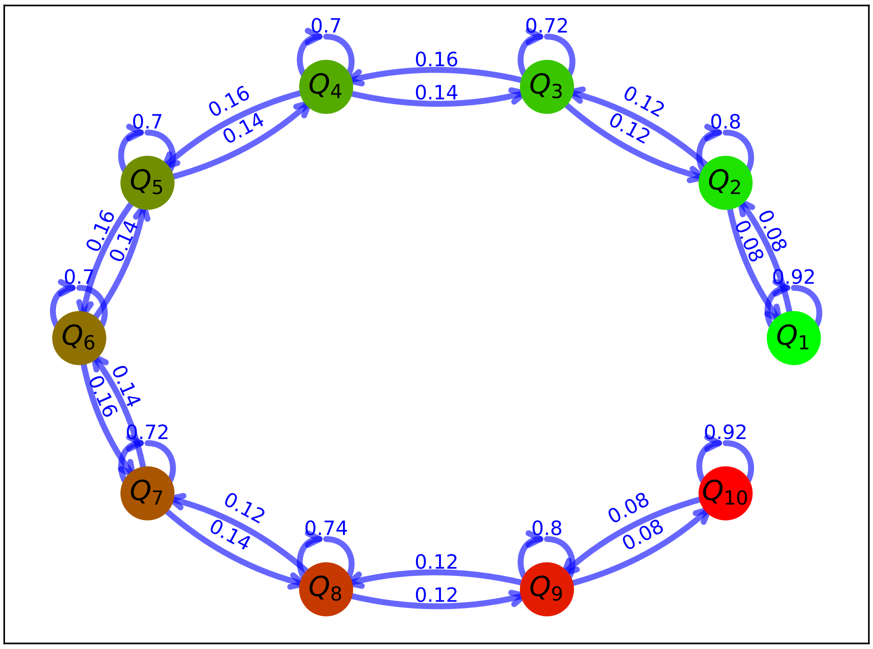


Figure 3.8: The graph of the Markov chain representing the transitions of the signal depicted in Figure 3.7.

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. Figure 3.9 pictures the final result of the method applied to the example signal.

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

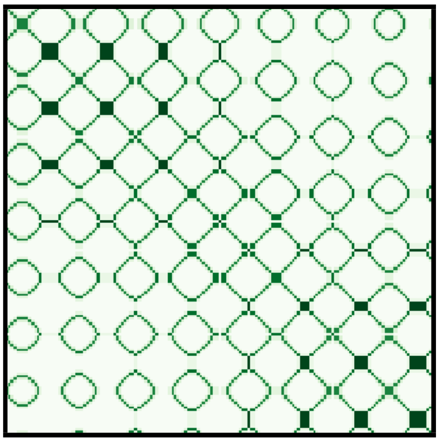


Figure 3.9: The Markov Transition Field of the example signal, with the number of quantile bins $m = 8$.

3.1.4 Projections comparison

The main idea of the projection methods is to reflect properties and shapes of the signal into visual patterns. That concept applies too to the inspection of the signal quality. 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 structures and the high-frequency noise figures contain small dot-like structures. Another observation is that noises concentrated in a certain region tend 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 noise and its characteristics as visual patterns that humans can recognize, raising the hypothesis that CV could learn them as well.

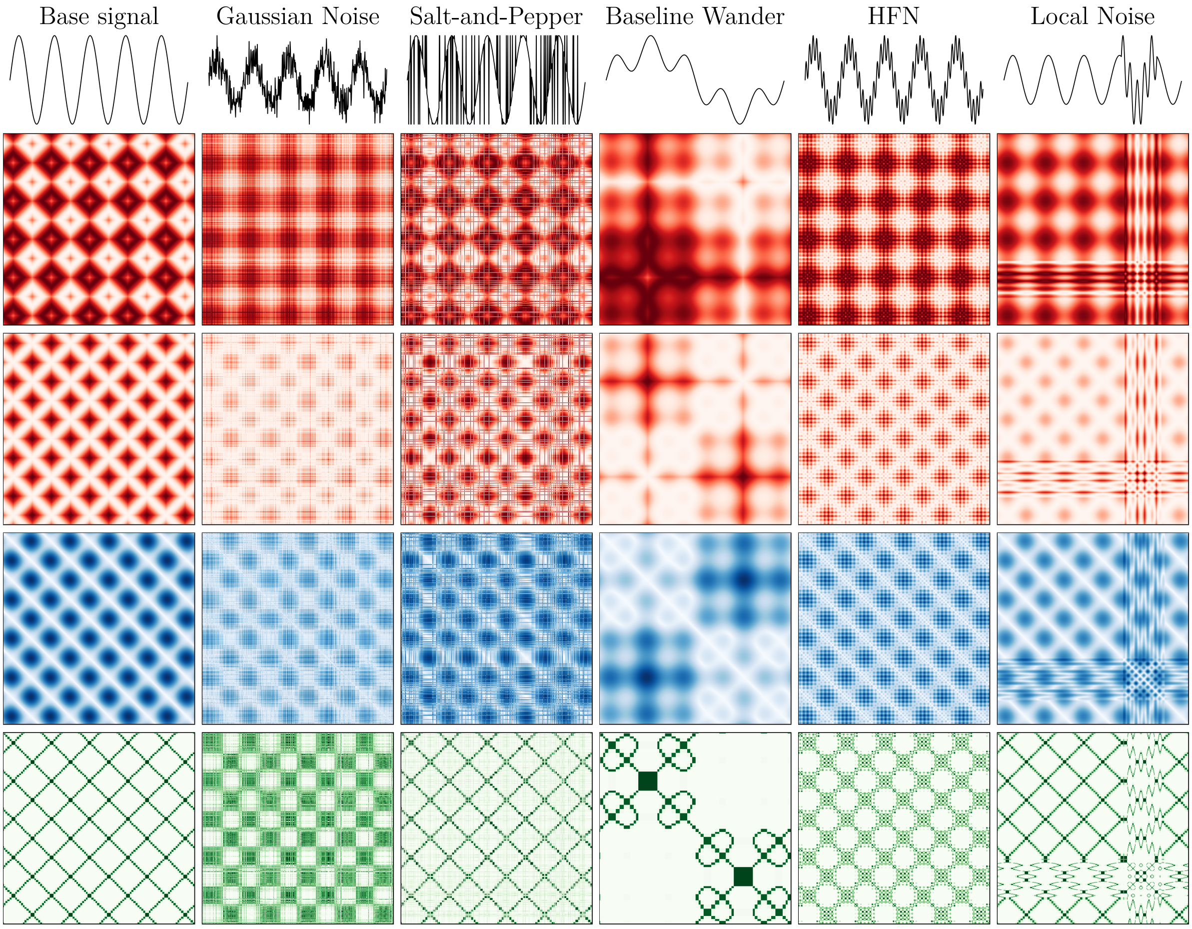


Figure 3.10: The signal, its impaired versions, and their corresponding 2D projections. From top to bottom: Gramian Angular Difference Field, Gramian Angular Summation Field, Recurrence Plot, and Markov Transition Field.

3.2 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 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 (1×1 kernel-sized) convolution operation. 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:

$$T_{in_{k,i,j}} = M_{k_{i,j}}. \tag{3.12}$$

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

$$T_{mix_{k,i,j}} = f_{actv}\left(\sum_{l=1}^{p} T_{in_{l,i,j}} \cdot w_{(k,i,j);l}\right), \tag{3.13}$$

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:

$$T_{mix_{k,i,j}} = f_{actv}\left(\sum_{l=1}^{p} M_{l_{i,j}} \cdot w_{(k,i,j);l}\right). \tag{3.14}$$

We can observe that, for each cell with indexes i, j, the summation $\sum\limits_{l=1}^{p} M_{l_{i,j}} \cdot w_{(k,i,j);l}$ "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 illustrates the implementation framework of that mixture in the context of models pre-trained in three-channeled datasets. That process may require resizing, since pointwise convolution does not change the width and height dimensions.

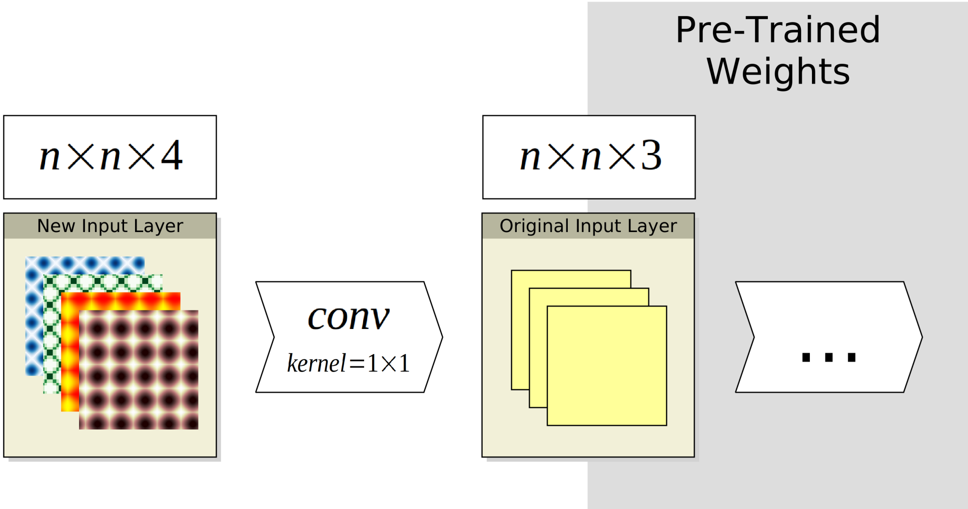


Figure 3.11: The three-channeled computer vision model feeding process. The figure begins in the left, in its input, and progresses to the right.

3.3 The proposed method

Based on the PMix projection method, this work proposes a new SQA framework presented by Figure 3.12. First, we transform the signal into four projections using the three before-mentioned algorithms: GAF, MTF, and RP. Afterwards, 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 [9] dataset. Finally, that model classifies the signal into a binary SQI, which indicates if the signal is "good" or "bad".

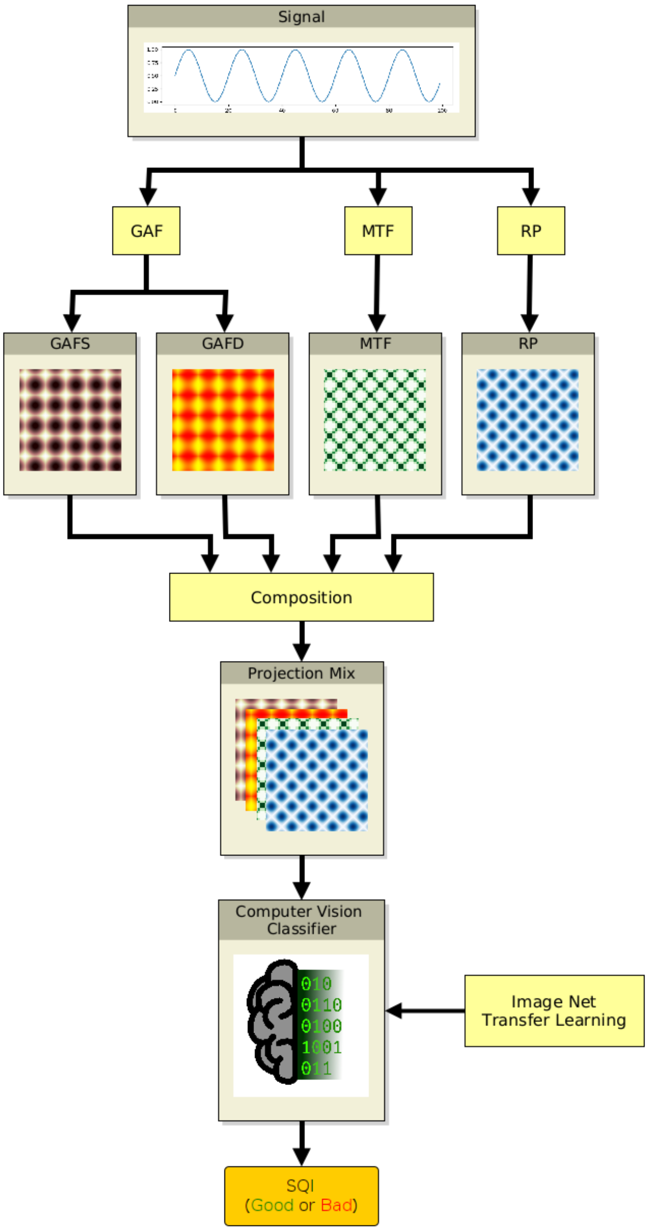


Figure 3.12: Proposed photoplethysmograph signal quality assessment framework.

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 [8] dataset was employed.

The Brno University of Technology smartphone PPG database is a publicly available[1] 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

[1] Available at `https://physionet.org/content/butppg/2.0.0/`

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 beats per minute (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 additional split is the validation dataset, used to select the best set of hyperparameters for the trained model or conducting the learning process. A direct way one could achieve the training-test split is randomly partitioning the dataset into both fragments. However, that method never uses the data in the training set for testing and vice-versa. Since the BUTPPG is small and it is desirable to reuse data, our experiments defined the training-test splits by 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. In our experiment, the smallest unit of division was the subject, not the individual signals associated with each subject, because having signals of the same subject in both training and testing datasets would introduce biases. This approach has the advantage of increasing the distinction between training and testing samples, since having the same subject in both training and test sets could also introduce biases into the results. Since the dataset is small, this splitting method increases the use of available resources by ensuring every sample is used as a test case at least once. Additionally, the training dataset was further divided using random partitioning to produce a validation dataset of size 3, used for early stopping.

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[2], 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[3], 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, which are briefly described in the following subsections of this section.

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 transforms 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 [96]. 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 [97]. 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 [98]. The SwinTV2 introduces several specific improvements over the earlier version [99].

[2]Documented at `https://www.aeon-toolkit.org/en/stable/`.
[3]Documented at `https://pytorch.org/docs/stable/index.html`.

Table 4.1: Non-computer vision models list, containing its references.

Classification	Model	Reference
Convolution-Based	Arsenal	[66]
	Rocket Classifier	[67]
Deep Learning	Zhao's CNN Classifier	[68]
	Wang's FCN Classifier	[69]
	Wang's MLP Classifier	[69]
	Inception Time Classifier	[70][71]
	Individual Inception Classifier	[70][71]
	LITE Time Classifier	[72]
Dictionary-Based	BOSS Ensemble	[73]
	Contractable BOSS	[74]
	Individual BOSS	[73]
	Individual TDE	[75]
	MUSE	[76]
	TemporalDictionaryEnsemble	[75]
	WEASEL	[77]
	WEASEL V2	[78]
	REDCOMETS	[79][80]
Distance-Based	Elastic Ensemble	[81]
	K-Neighbors Time Series Classifier	—
	Shape DTW	[82]
Feature-Based	Catch-22 Classifier	[83]
	Summary Classifier	—
	TS Fresh Classifier	[84]
Inverval-Based	Canonical Interval Forest Classifier	[85]
	DrCIFClassifier	[66]
	Random Interval Spectral Ensemble Classifier	[86]
	Supervised Time Series Forest	[87]
	Time Series Forest Classifier	[88]
	Random Interval Classifier	—
Shapelet-Based	Shapelet Transform Classifier	[89][90]
	RDST Classifier	[91][92]
Ordinal Classification	Individual Ordinal TDE	[93]
	Ordinal TDE	[93]
Other	Continuous Interval Tree	[94]
	Rotation Forest Classifier	[95]

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

Classification	Model	Reference
Transformer	Vision Transformer	[96]
	MaxViT	[97]
	Swin Transformer	[98]
	Swin Transformer V2	[99]
Residual Net	ResNet	[100]
	ResNeXt	[101]
	WideResNeXt	[102]
Extreme Net	DenseNet	[103]
	VGG	[104]
	SqueezeNet	[105]
Mobile-Oriented	MNASNet	[106]
	MobileNet V2	[107]
	MobileNet V3	[108]
Efficiency-Oriented	EfficientNet	[109]
	EfficientNet V2	[110]
	ShuffleNet V2	[111]
Diverse	AlexNet	[112]
	ConvNeXt	[113]
	RegNet	[114]

Residual nets

This thesis defines the residual nets as the ResNet model and its variations. ResNet introduced residual connections, which are links between non-adjacent layers that bypass intermediate layers [100]. 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 [102]. In contrast, ResNeXt employs a multipath approach, aggregating paths through an additive operation [101]. Instead of increasing width and depth, ResNeXt introduces an additional dimension for enhancement.

Mobile-oriented models

This thesis defines mobile-oriented models as networks that are designed specifically to address mobile hardware constraints. Three models were tested: MNASNet [106], MobileNet V2 [107], and MobileNet V3 [108]. MobileNet V2, introduced first, incorporates architectural changes to reduce memory usage while maintaining accuracy, including inverted residual blocks [107]. This alteration swaps high- and low-channel layers, connecting layers with fewer channels and thus reducing the number of parameters in the block [107]. MNASNet selects blocks to fit a predefined architectural skeleton, optimizing model performance on real-world mobile hardware [106]. MobileNet V3 combines these approaches and introduces additional changes, such as incorporating the NetAdapt [115] algorithm into the architectural search [108].

Extreme nets

This thesis defines as extreme nets neural models that focus on maximizing specific concepts, such as layer depth [104], model compression [105], or residual connections [103], include VGG, DenseNet, and SqueezeNet. VGG employs 3×3 filters to allow for deeper network architectures by adding more layers, thus increasing model depth [104]. 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 [103]. 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 [105]. This approach significantly reduces its number of parameters when comparing to convolutions applied over a layer with a high amount of channels [105].

Efficiency-oriented

This thesis defines efficiency-oriented networks as models designed for efficient resource utilization aiming to maximize performance with fewer parameters, such as ShuffleNet V2 [111], EfficientNet [109], and EfficientNet V2 [110]. ShuffleNet V2 is an advancement of ShuffleNet, introducing the channel shuffle operator to facilitate information exchange among channels [111]. It improves upon its predecessor by incorporating a channel split operation within each block, which avoids the use of costly grouped convolutions [111]. EfficientNet focuses on model scaling through a compound resizing method that proportionally increases multiple dimensions, such as depth, number of channels, and resolution [109]. This approach creates a highly efficient base model that can be scaled up to larger variants while preserving the original model's advantages [109]. EfficientNet V2 builds on the original EfficientNet by proposing non-proportional scaling and utilizing network architecture search [110]. It also introduces progressive learning, which involves gradually increasing dataset regularization [110].

Diverse

The remaining models not included in the anterior groups were reunited in this category. It 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 [112]. 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 [113]. 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 [114]. The experiments in that category encompassed networks with significant variations among them.

4.1.4 Defining the hyperparameters

However, defining the models alone is insufficient, as the selection of their hyperparameters is also required. Hyperparameters are parameters related to the learning process, rather to the model itself. For the Aeon models, the default hyperparameters provided by the library were used for convenience reasons, even though they are not the optimal ones. 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 to search for better hyperparameters than the defaults provided by the PyTorch library. The Optuna Python library[4] 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. It was chosen since maximizing that ratio is desired, as the more correct predictions, the better. The accuracy was measured on a validation dataset of size equal to 2 subjects, 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.

4.1.5 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, which our experiments used for convenience reasons, our experiments only established the fitting framework for the PyTorch computer vision models and the data feeding method. Our experiment involved feeding

[4] Documented at `https://optuna.readthedocs.io/en/stable/`.

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] [9] dataset. Such choice was made because this work hypothesizes that a model trained in a dataset with real word images might already be familiar with the basic shapes present in the 2D projection methods. Following that, we fitted the PyTorch models using the Adam optimization algorithm [116] to minimize the cross-entropy loss function. A reason to use that algorithm is that it surpassed some of the other options present in the PyTorch library when tested in several datasets [116]. The implementation of the training strategy performed this optimization cycle with a number of epochs determined by a median-deviation-based early stopping technique, through the assumption that a low dispersion on the last epochs indicates a convergence to a local-optimal in the search space. 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 $[f(i)]_{i=0}^{p}$ is the sequence generated by $f(i)$ when varying i from 0 to p. 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.

4.1.6 Performance measurements

Being established the training procedure, it is needed to choose metrics to evaluate the efficacy of the solution after the training of the model. 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, considering that they are capable of estimating the quality of binary classification through different perspectives. All of them can be evaluated using confusion matrix values, presented in Table 4.3. The Cohen kappa score [117], in binary classification tasks, measures the agreement between the obtained accuracy acc_o and the expected accuracy acc_e. The following equations define

[5] Available at `https://image-net.org/`.

those accuracies and the Cohen kappa score:

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

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

and

$$CohenKappa(R) = \frac{acc_o - acc_e}{1 - acc_e}, \tag{4.4}$$

where $N = TP + TN + FP + FN$ is the total number of samples. For the purpose of aligning this metric with others, we can rescale that metric from $[-1, 1]$ to $[0, 1]$:

$$CohenKappaRescaled(R) = \frac{CohenKappa(R) + 1}{2}. \tag{4.5}$$

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 implies that the predictor avoided mistakenly labeling "bad" signals as "good", 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:

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

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

and

$$F1 = HarmonicMean(Precision, Recall) = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}. \tag{4.8}$$

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

Table 4.3: Binary classification confusion matrix, where each cell resulting from the intersection between the i-th line and the j-th colum correspond to the amount of data which was predicted as the i-th line label and had the j-th column label as the true value.

Predicted label \ True label	Positive	Negative
Positive	True positive (TP)	False positive (FP)
Negative	False negative (FN)	True negative (TN)

memory usage. Additionally, inference time is important for achieving near-instantaneous evaluations, which enhances the application's responsiveness.

4.1.7 Overall schema

Figure 4.1 illustrates the experiment framework applied to each combination of computer vision model and projection algorithm. 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. So the experiment for non-CV models is 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 learning rate for the CV models. With the best learning rates 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.

4.1.8 The implementation details

The dataset sourcing procedure was carried out using the PyTorch multithreading data feeding solution, Data Loader[6]. This was configured to load batches of size 32 for all CV models to make the comparison more uniform, since that variable can change their performance. Prior to loading the batches, the training dataset was balanced using the Imbalanced Learn library[7] and its random oversampling method[8], since it is a open source implementation of an algorithm that equalizes the proportion of labels in the learning process, avoiding an label unbalance that our experiment design hypothesizes that it

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

[7]Documentation at `https://imbalanced-learn.org/stable/`.

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

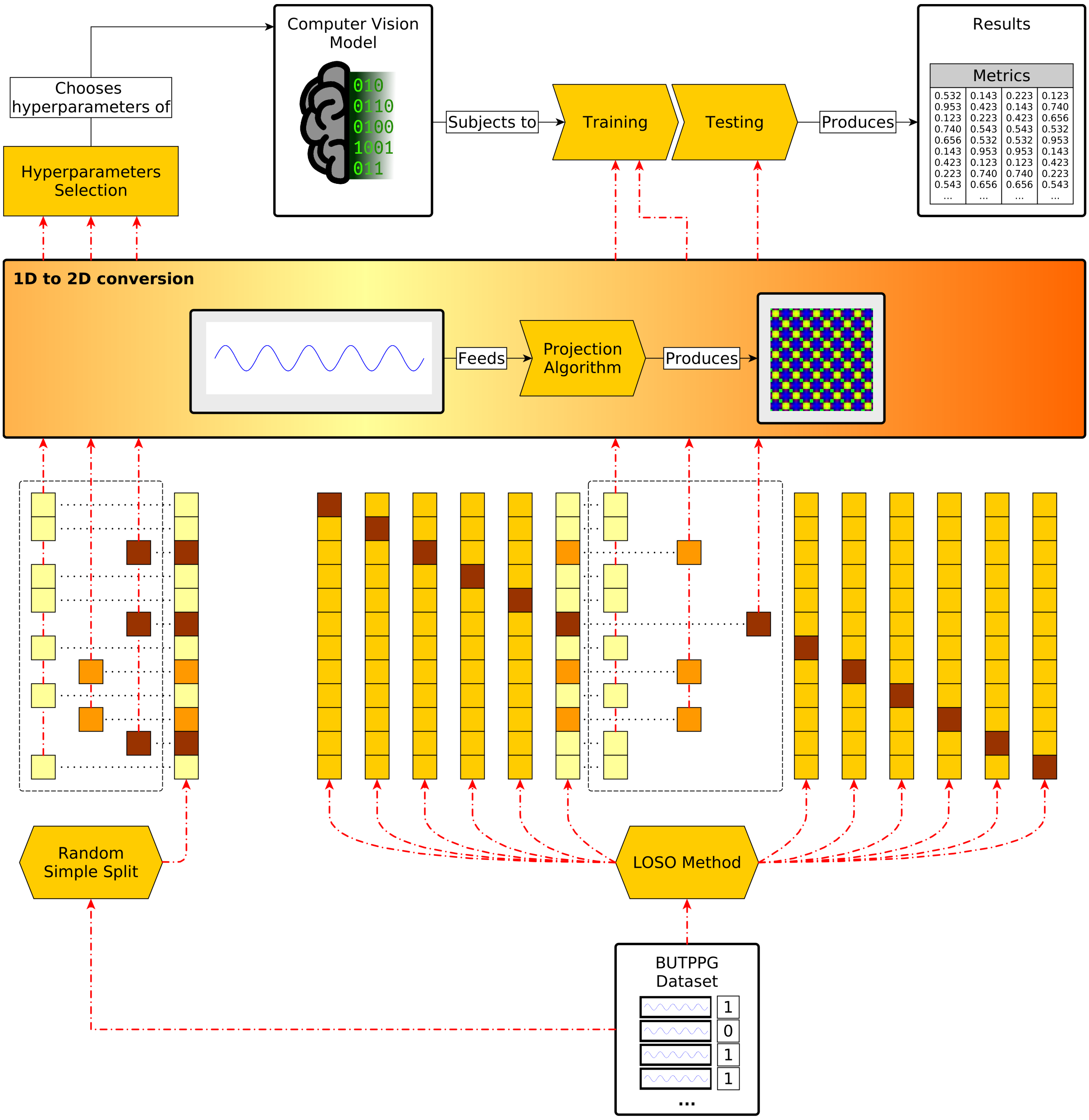


Figure 4.1: The framework of the experiment. Red dotted arrows indicates data flow that sources from the Brno University of Technology smartphone PPG database dataset, while the full black lines, labeled with an verb, represent a relationship "A do B", where the arrow starts on A and end on B. Notice that the figure presents the training-testing cycle for only one of the twelve folds.

could overfit the model to a specific label, which is undesirable. The batches were then transformed from 1D signals into 2D images using the projection algorithms of the PyTS library[9] [118], which allows the reproducibility of such transformations for being open source. 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[10] was applied to adjust the width and height. Additionally, a new convolutional layer corresponding to the PMix method was incorporated.

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[11], which uses the gradients evaluated by the Pytorch autograd engine [119]. The loss class (which, in our case, is the torch.nn.CrossEntropyLoss[12], used for being an open source implementation of the cross entropy loss function) backpropagates the gradients based on the model forward pass errors. For the models testing, our implementation used the Sklearn's[13] metrics[14], since they are open source. 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[15] of the Pympler library[16]. Finally, we describe the inference time measurement, for which our implementation extracted 500 measurement samples. For the non-CV models, our implementation used the `time` method[17] of the Python's time module, from its standard library, to measure two time instants: the moment right before the model testing predictions, when the model is already trained; and the moment right after those predictions. Our implementation evaluates the time interval between those instants to estimate the inference time of the non-CV model. For the CV models, our implementation marked the time instants by using CUDA events interface provided by Pytorch[18], while, before measuring, performing 500 iterations to warm-up the GPU.

[9] Documentation available at `https://pyts.readthedocs.io/en/stable/`.
[10] Documentation available at `https://pytorch.org/vision/stable/generated/torchvision.transforms.Resize.html`.
[11] Documentation available at `https://pytorch.org/docs/stable/generated/torch.optim.Adam.html#torch.optim.Adam`
[12] Documentation available at `https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html#torch.nn.CrossEntropyLoss`
[13] Documented at `https://scikit-learn.org/`.
[14] Documentation at `https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics`
[15] Documentation at `https://pympler.readthedocs.io/en/latest/library/asizeof.html`
[16] Documented at `https://pympler.readthedocs.io/en/latest/`.
[17] Documentation at `https://docs.python.org/3/library/time.html#time.time`
[18] Documentation at `https://pytorch.org/docs/stable/generated/torch.cuda.Event.html`

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. The score metrics are presented like the example in Table 4.4, in which the results of each combination of model and projection are shown in the format $mean \pm std$, with values up to three decimal places. Additionally, considering a particular column of mean or standard deviation, it is used a coloring system where the red color represents values worse or equal than the first quartile, the green color highlights values better or equal than the third quartile, and the blue color paints values in between the other colors. 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.

Table 4.4: Example of the score-displaying system.

Model	Projection	Example metric 1	Example metric 2
Example model 1	Example projection 1	0.999 ± 0.333	0.777 ± 0.111
Example model 2	Example projection 2	0.333 ± 0.999	0.999 ± 0.333
Example model 3	Example projection 3	0.777 ± 0.111	0.111 ± 0.777
Example model 4	Example projection 4	0.111 ± 0.777	0.333 ± 0.999

4.2.1 Analysis by computer vision model family

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

Transformers

One metric table was generated for each type of transformer. Table 4.5 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 4.6 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 4.7 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 4.8 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 4.5, 4.6, 4.7, and 4.8, 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 4.9 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. Therefore, we select the following methods for this section:

- SwinT V2 S with PMix;
- and ViT B 32 with PMix.

Table 4.5: Averages and standard deviations of the folds evaluation for the Vision Transformer variants.

Model	Projection	Cohen Kappa	F1 Score	Precision
ViT: 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 (proposed)	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
ViT: 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 (proposed)	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
ViT: 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 (proposed)	0.833 ± 0.222	0.931 ± 0.087	0.903 ± 0.146
ViT: 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 (proposed)	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130
ViT: 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 (proposed)	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130

Table 4.6: Averages and standard deviations of the folds evaluation for the MaxViT variants.

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 (proposed)	**0.875** ± **0.169**	0.921 ± **0.098**	**0.944** ± **0.130**

Table 4.7: Averages and standard deviations of the folds evaluation for the Swin Transformer V2 variants.

Model	Projection	Cohen Kappa	F1 Score	Precision
SwinT: 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 (proposed)	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129
SwinT: 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 (proposed)	0.792 ± 0.234	0.919 ± 0.087	0.882 ± 0.148
SwinT: 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 (proposed)	**0.896** ± 0.167	0.938 ± 0.093	**0.944** ± 0.130

Table 4.8: Averages and standard deviations of the folds evaluation for the Swin Transformer V2 variants.

Model	Projection	Cohen Kappa	F1 Score	Precision
SwinTV2: 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 (proposed)	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
SwinTV2: 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 (proposed)	**0.917** ± 0.163	**0.955** ± **0.083**	0.944 ± 0.130
SwinTV2: 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 (proposed)	0.500 ± **0.000**	0.837 ± 0.090	0.729 ± 0.129

Table 4.9: Memory size in Mega Bytes of each Transformers family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
SwinT: T	110.083712	SwinT: B	346.981336
SwinTV2: T	110.336672	SwinTV2: B	347.632024
MaxViT	122.144800	ViT: B 32	349.827128
SwinT: S	195.355424	ViT: H 14	1213.214776
SwinTV2: S	195.880352	ViT: L 16	1213.214776
ViT: B 16	343.200824	ViT: L 32	1222.049848

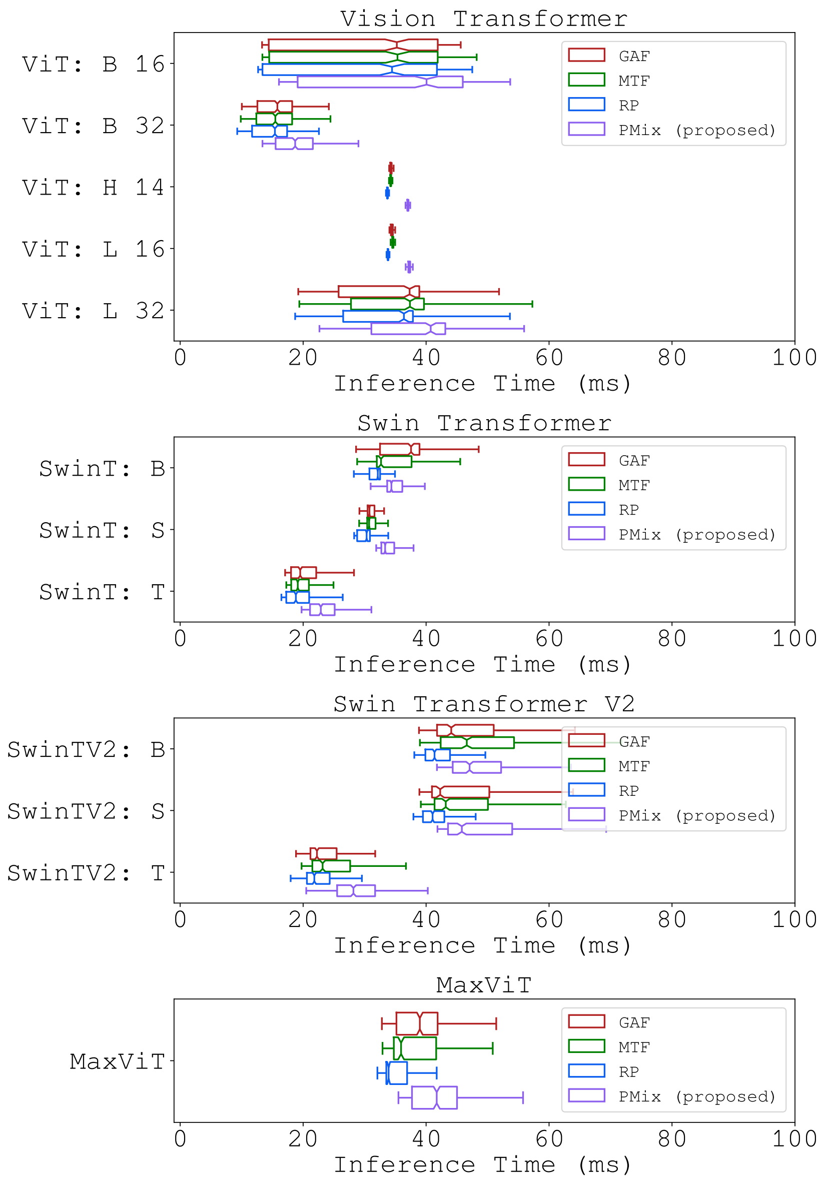


Figure 4.2: Inference time in milliseconds of each Transformers family model variant.

Residual nets

The experiments produced three score tables. Table 4.10 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 4.11 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 4.12 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 4.10, 4.11, and 4.12 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, according to Table 4.13, and was the fourth slowest in inference time, as seen in Figure 4.3. An alternative with nearly the second-best scores but significantly lower memory usage and inference time is the ResNet 50 with PMix. Thus, the two methods below were chosen for this section:

- ResNet 50 with PMix;
- and Wide ResNet 101-2 with PMix.

Table 4.10: Averages and standard deviations of the folds evaluation for the ResNet variants.

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 (proposed)	**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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130

Table 4.11: Averages and standard deviations of the folds evaluation for the ResNeXt variants.

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 (proposed)	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 (proposed)	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 (proposed)	0.792 ± 0.234	0.919 ± 0.087	0.882 ± 0.148

Table 4.12: Averages and standard deviations of the folds evaluation for the Wide ResNet variants.

Model	Projection	Cohen Kappa	F1 Score	Precision
WiResNet: 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 (proposed)	**0.955** ± **0.101**	**0.967** ± **0.078**	0.944 ± 0.130
WiResNet: 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 (proposed)	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140

Table 4.13: 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	WiResNet: 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	WiResNet: 101-2	499.369720

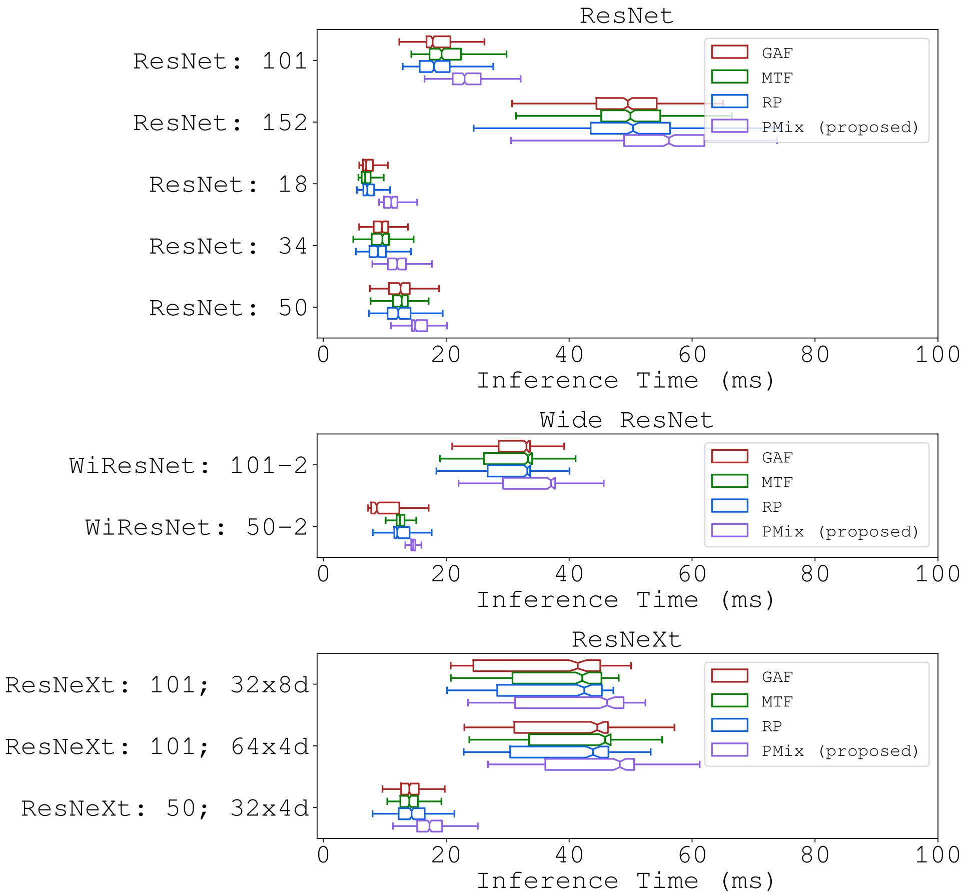


Figure 4.3: Inference time in milliseconds of each Residual Nets family model variant.

Mobile-oriented

Three metric tables were generated for mobile-oriented family of CV models. Table 4.14 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 4.15 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 4.16 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 4.14, 4.15 and 4.16, the combination of MNASNet 1.0 with PMix emerges as the overall best case. This combination demonstrates a competent inference time when comparing to the other models in the category, as shown in Figure 4.4, but its memory consumption was not among the best models in the category, according to Table 4.17. Nonetheless, we elect only one method as the best models of this section:

- MNASNet 1.0 with PMix.

Table 4.14: Averages and standard deviations of the folds evaluation for the MNASNet variants.

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 (proposed)	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 (proposed)	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 (proposed)	**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 (proposed)	0.604 ± 0.249	0.884 ± 0.107	0.750 ± 0.312

Table 4.15: Averages and standard deviations of the folds evaluation for the MobileNet V2 variants.

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 (proposed)	0.854 ± 0.249	**0.951 ± 0.086**	0.889 ± 0.296

Table 4.16: Averages and standard deviations of the folds evaluation for the MobileNet V3 variants.

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 (proposed)	**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 (proposed)	0.667 ± 0.246	0.822 ± 0.174	0.833 ± 0.207

Table 4.17: 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

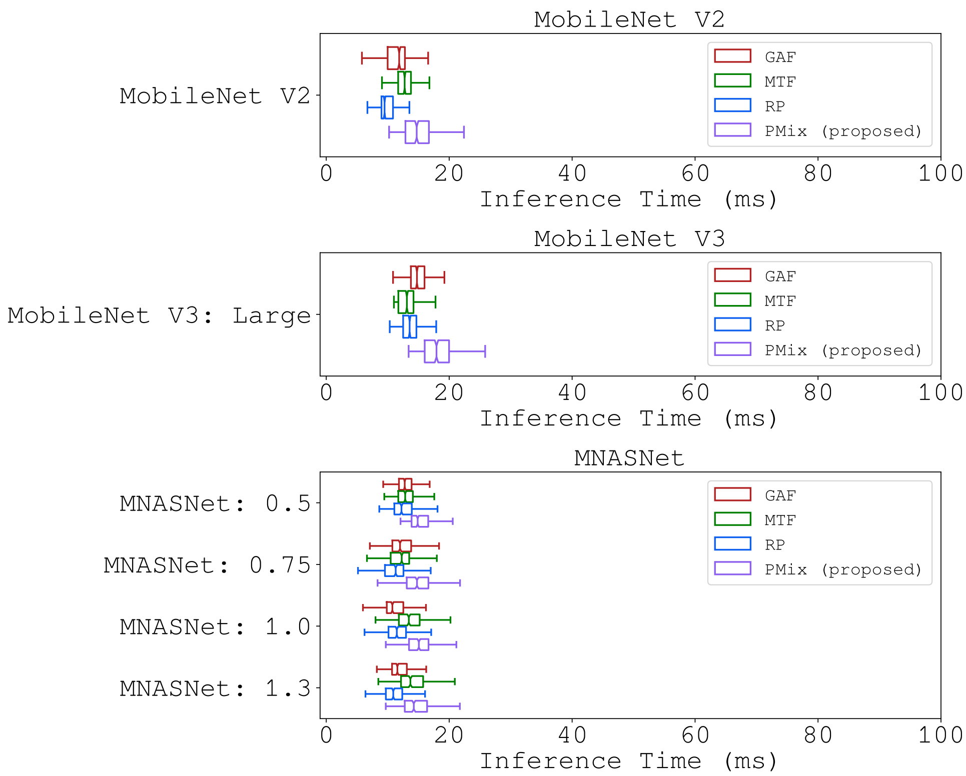


Figure 4.4: Inference time in milliseconds of each Mobile-Oriented family model variant.

Extreme nets

Three metric tables were constructed, each corresponding to a model within the CV family. Table 4.18 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 4.19 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 4.20 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 4.21 shows that this combination also ranks as the smallest in memory consumption. Thence, we choose two methods as representative of this section:

- VGG 16 BN with PMix;
- and SqueezeNet 1.1 with PMix.

Table 4.18: Averages and standard deviations of the folds evaluation for the DenseNet variants.

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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	0.875 ± 0.199	**0.943** ± 0.086	0.924 ± 0.140

Table 4.19: Averages and standard deviations of the folds evaluation for the SqueezeNet variants.

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 (proposed)	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 (proposed)	**0.917** ± 0.163	**0.955** ± 0.083	0.944 ± 0.130

Table 4.20: Averages and standard deviations of the folds evaluation for the VGG variants.

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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	0.708 ± 0.257	0.885 ± 0.123	0.833 ± 0.195

Table 4.21: 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

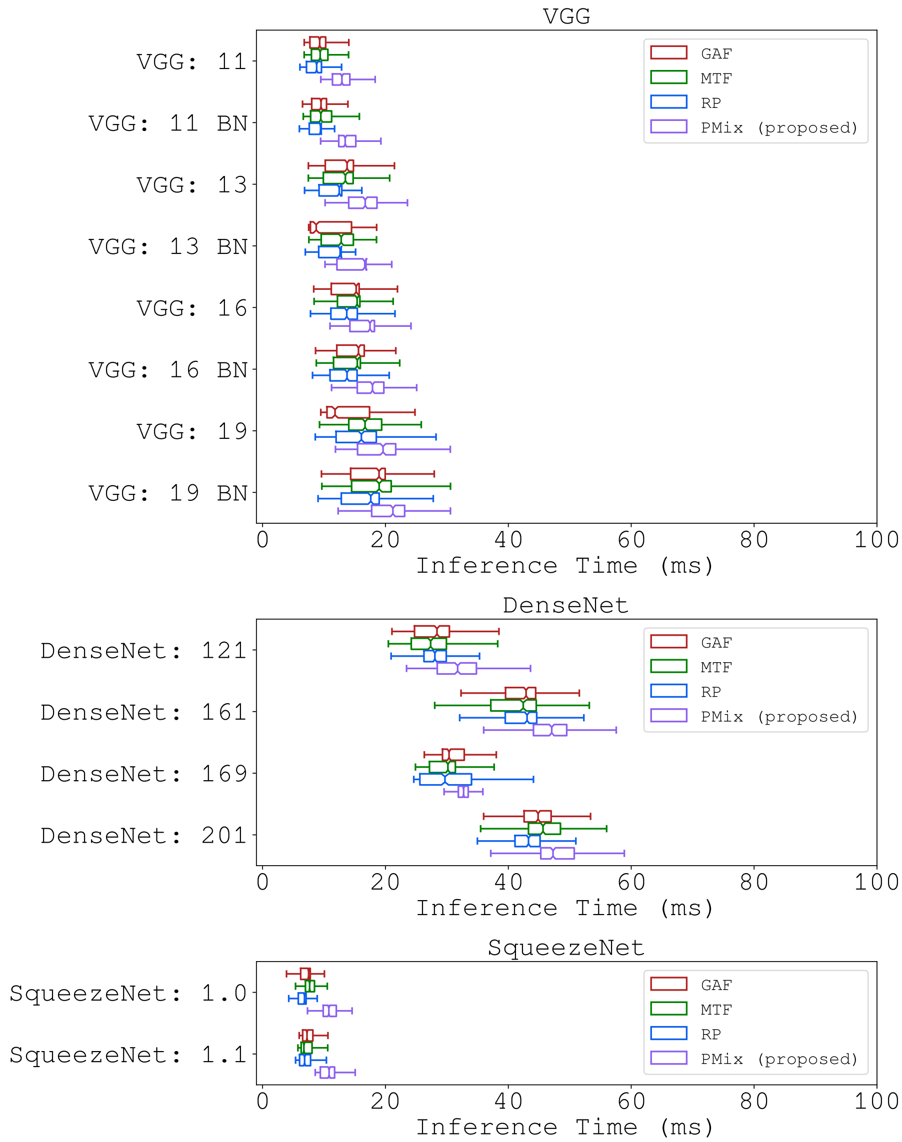


Figure 4.5: Inference time in milliseconds of each Extreme Nets family model variant.

Efficiency-oriented

Three score tables were constructed for the Efficiency-Oriented CV family. Table 4.22 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 4.23 presents the scores for EfficientNet V2, with the PMix projection outperforming all other methods. Table 4.24 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 4.22, 4.23, and 4.24, 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 4.25, while being one of the fastest in inference, as visible in the Figure 4.6. Thus, only the following method was selected for this section:

- ShuffleNet V2$\times$0.5 with PMix.

Table 4.22: Averages and standard deviations of the folds evaluation for the EfficientNet variants.

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 (proposed)	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 (proposed)	**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 (proposed)	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 (proposed)	0.604 ± 0.225	0.788 ± 0.181	0.792 ± 0.209

Table 4.23: Averages and standard deviations of the folds evaluation for the EfficientNet V2 variants.

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 (proposed)	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± **0.130**

Table 4.24: Averages and standard deviations of the folds evaluation for the ShuffleNet V2 variants.

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 (proposed)	**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 (proposed)	**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 (proposed)	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 (proposed)	0.729 ± 0.249	0.896 ± 0.106	0.847 ± 0.173

Table 4.25: 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		

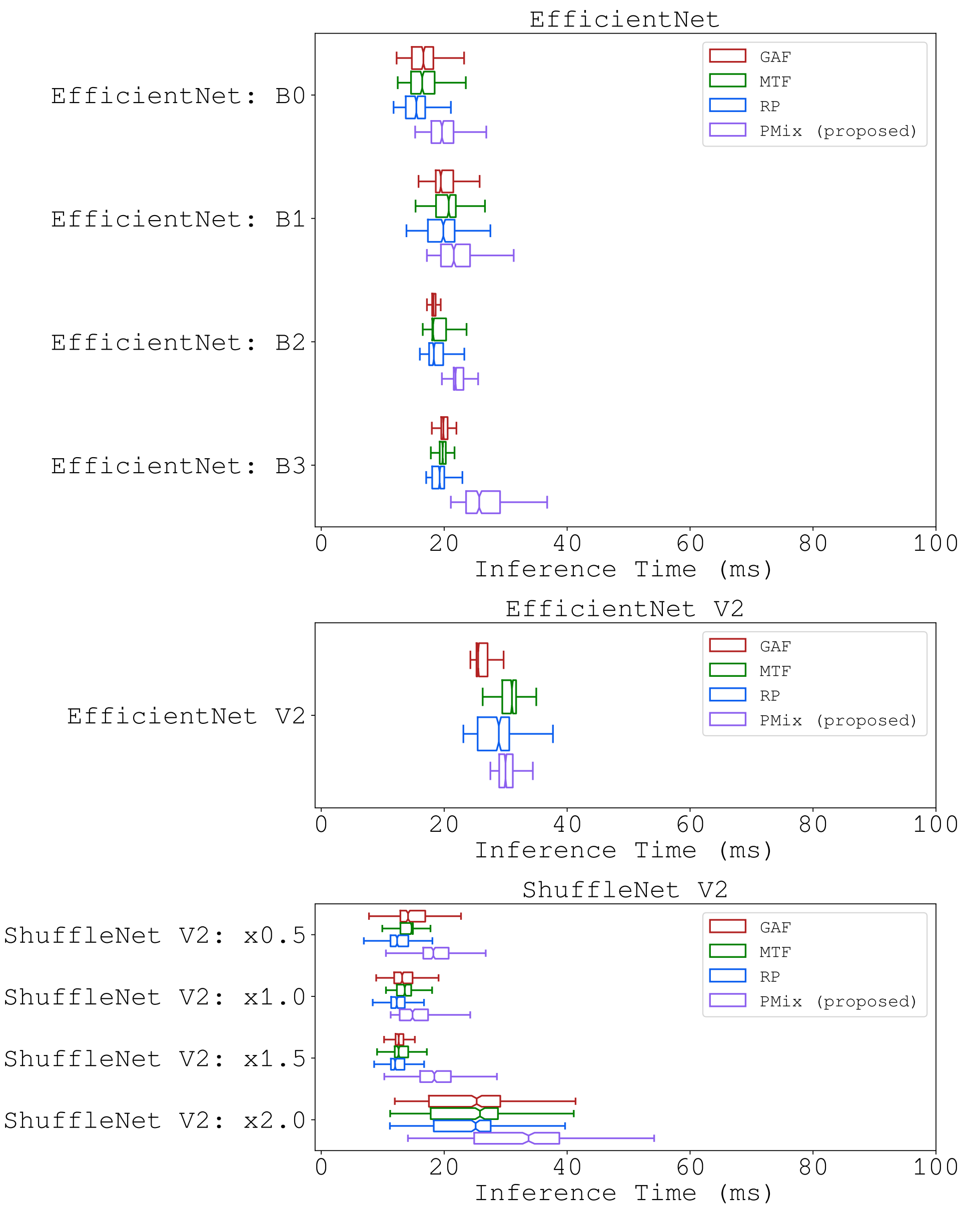


Figure 4.6: Inference time in milliseconds of each Efficiency-Oriented family model variant.

Further architectures

Three score tables were generated for the remaining computer vision models. The Table 4.27 contains the scores of AlexNet. Notably, the PMix projection earned the best scores for all metrics in that table. Table 4.28 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 4.26 exhibits the results for RegNet variants, categorized into RegNetX and RegNetY design spaces [114], 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 4.29, 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 4.29, while AlexNet had the fastest inference times across all projections, as illustrated in Figure 4.7. Hence, two methods were chosen for this section:

- AlexNet with PMix;
- and RegNet Y 400 MF with RP.

Table 4.26: Averages and standard deviations of the folds evaluation for the RegNet variants.

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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	**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 (proposed)	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 (proposed)	**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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	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 (proposed)	**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 (proposed)	0.771 ± 0.249	0.908 ± 0.109	0.868 ± 0.176

Table 4.27: Averages and standard deviations of the folds evaluation for the AlexNet variants.

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 (proposed)	**0.917** ± 0.163	**0.955** ± **0.083**	**0.944** ± **0.130**

Table 4.28: Averages and standard deviations of the folds evaluation for the ConvNeXt variants.

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 (proposed)	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 (proposed)	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 (proposed)	**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 (proposed)	0.875 ± 0.199	**0.951** ± **0.086**	0.939 ± 0.135

Table 4.29: 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

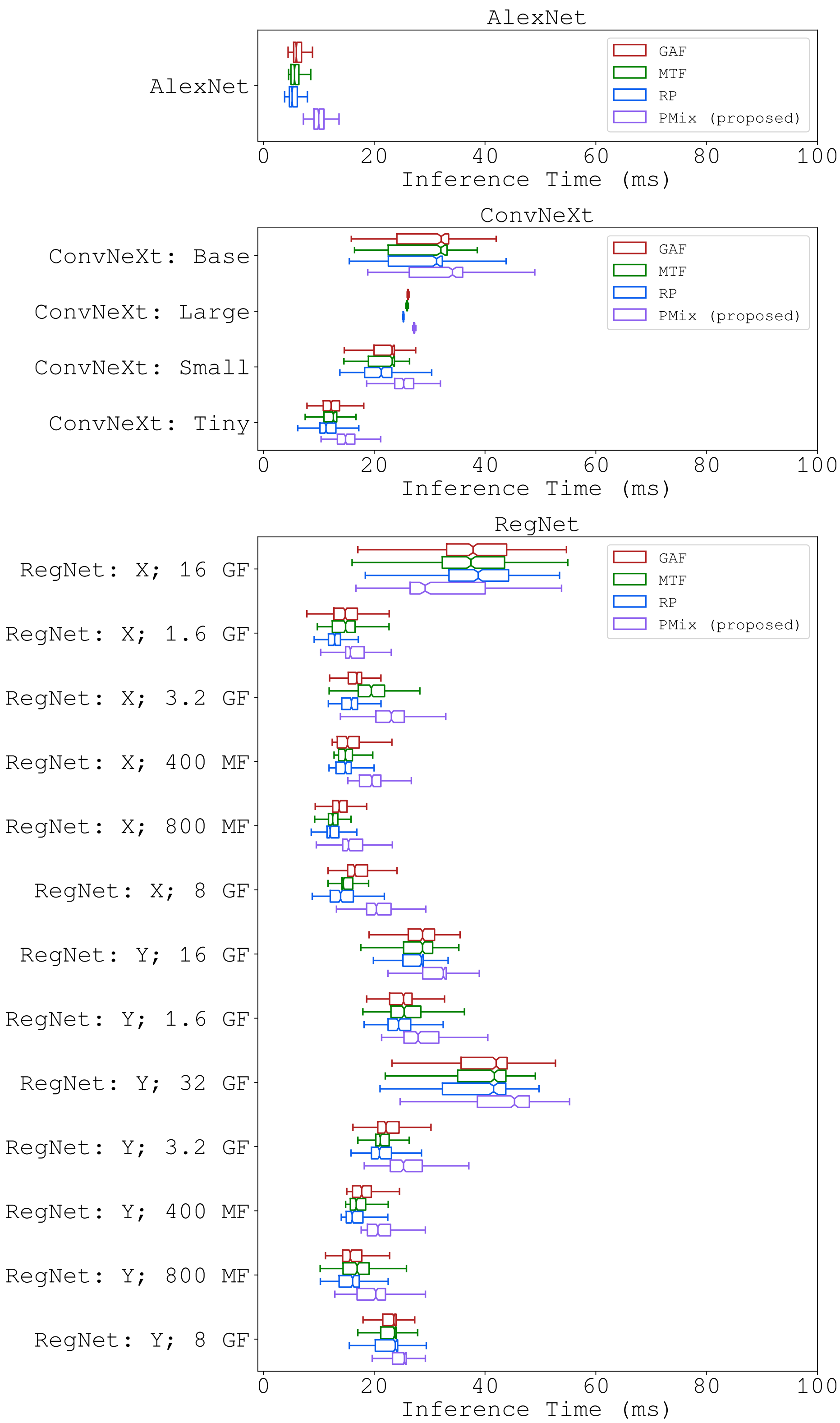


Figure 4.7: Inference time in milliseconds of each Diverse family model variant.

4.2.2 Non-computer vision models comparison

Table 4.30 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 4.31. So, we elect the two methods below as the best models of this section:

- RISEC;
- and TDE.

Table 4.30: Averages and standard deviations of the folds evaluation for the Non-CV variants.

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
RISEC	**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
TDE	**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

Table 4.31: 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
TDE	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
RISEC	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		

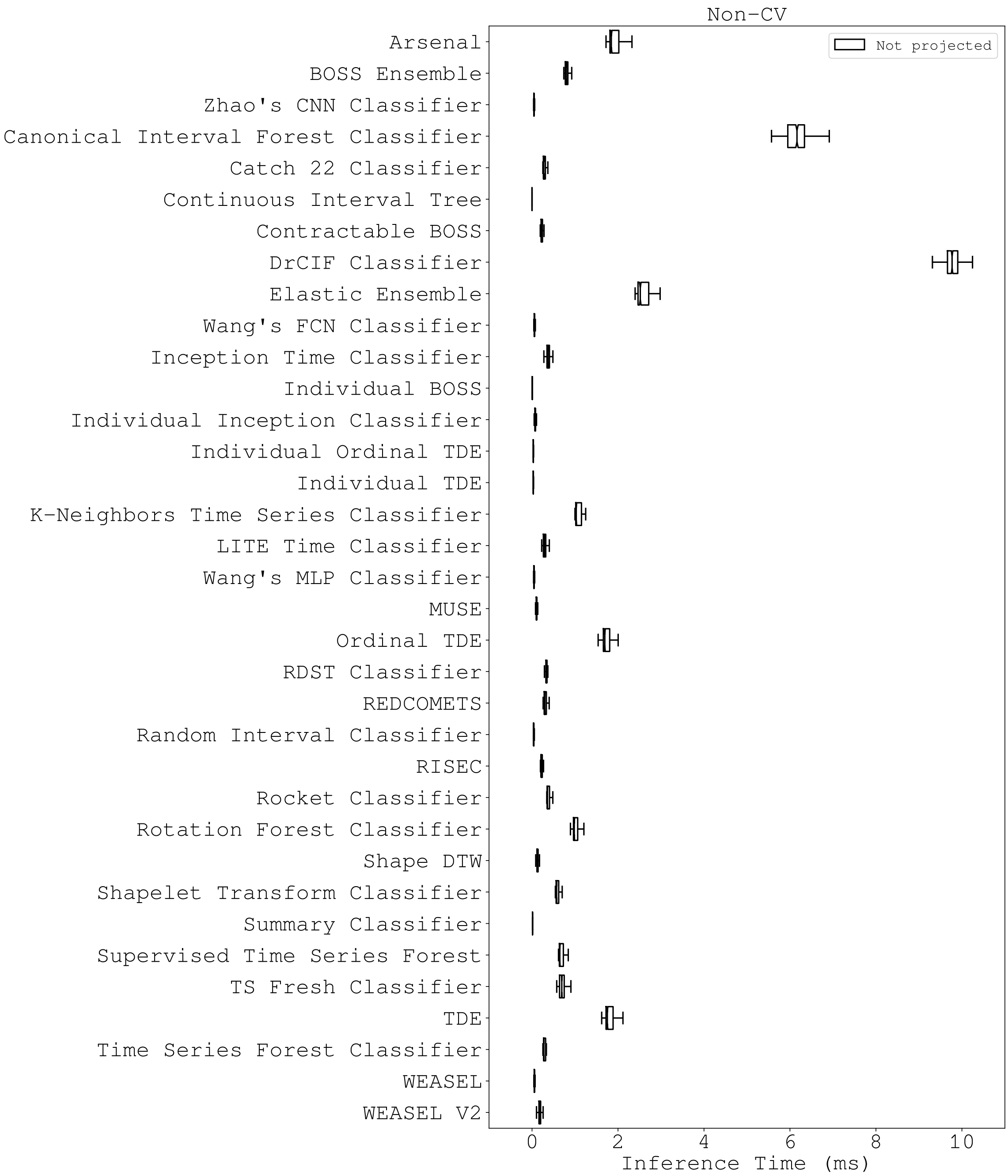


Figure 4.8: Inference time in milliseconds of each Non-CV family model variant.

4.2.3 Comparison of the top-performing models

The best combinations between models and projections from previous analyses are aggregated in Table 4.32. 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 4.33. 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.

Table 4.32: Averages and standard deviations of the folds evaluation for the best models variants.

Model	Projection	Cohen Kappa	F1 Score	Precision
AlexNet	PMix (proposed)	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
MNASNet: 1.0	PMix (proposed)	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
RISEC	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 (proposed)	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
ShuffleNet V2: x0.5	PMix (proposed)	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
SqueezeNet: 1.1	PMix (proposed)	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
SwinTV2: S	PMix (proposed)	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
TDE	Not projected	0.938 ± 0.155	**0.971 ± 0.068**	**0.972 ± 0.096**
VGG: 16 BN	PMix (proposed)	0.896 ± 0.198	0.960 ± 0.075	0.951 ± 0.115
ViT: B 32	PMix (proposed)	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
WiResNet: 101-2	PMix (proposed)	**0.955 ± 0.101**	0.967 ± 0.078	0.944 ± 0.130

Table 4.33: Memory size in Mega Bytes of each best models family model variant.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
TDE	0.586264	ResNet: 50	94.049840
RISEC	1.171312	SwinTV2: S	195.880352
ShuffleNet V2: x0.5	1.376760	AlexNet	228.048184
SqueezeNet: 1.1	2.894136	ViT: B 32	349.827128
MNASNet: 1.0	12.420792	WiResNet: 101-2	499.369720
RegNet: Y; 400 MF	15.617376	VGG: 16 BN	537.109104

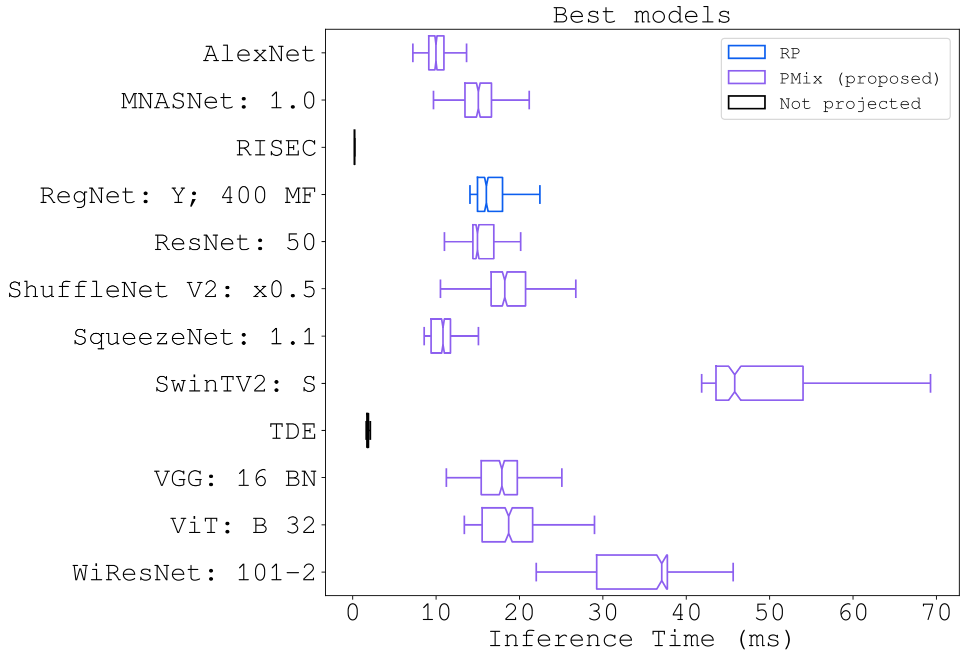


Figure 4.9: Inference time in milliseconds of each best models family model variant.

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 library, and the Hydra Classifier [121], from the Aeon library. Additionally, models external to these libraries, such as Xception [122] available in the Keras library[19], 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 [123], 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.

[19] Accessible at `https://keras.io/`.

Chapter 5

Conclusion

This work, named "Projection-Based Photoplethysmography Signal Quality Assessment", presented a study on SQA for PPG signals, mainly focused on the 1D-to-2D-projection approach. The investigated projection-based approach involved transforming 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. Nonetheless, the method of this work was published on the *Anais do XXIV Simpósio Brasileiro de Computação Aplicada à Saúde* through the peer-reviewed work "*On the Performance of Composite 1D-to-2D Projections for Signal Quality Assessment*" [124], in which we tested the proposed method and the RP, MTF, and GAF methods in the same BUTPPG dataset with a smaller set of CV models.

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 4.32, 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 4.32, 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 that the proposed method is 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.

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

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 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. [56]. 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] Thenappan, T., J. A. Raza, and A. Movahed: *Review: Aortic atheromas: Current concepts and controversies—a review of the literature.* Echocardiography, 25(2):198–207, 2008. https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-8175.2007.00568.x. 1

[2] Kwok, C. S. and W. E. Moody: *The importance of pathways to facilitate early diagnosis and treatment of patients with cardiac amyloidosis.* Therapeutic Advances in Cardiovascular Disease, 17:17539447231216318, 2023. https://doi.org/10.1177/17539447231216318, PMID: 38099406. 1

[3] Aouedi, O., T. H. Vu, A. Sacco, D. C Nguyen, K. Piamrat, G. Marchetto, and Q. V. Pham: *A survey on intelligent internet of things: applications, security, privacy, and future directions.* IEEE Communications Surveys & Tutorials, 2024. 1

[4] Nikus, Kjell, Olle Pahlm, Galen Wagner, Yochai Birnbaum, Juan Cinca, Peter Clemmensen, Markku Eskola, Miquel Fiol, Diego Goldwasser, Anton Gorgels, Samuel Sclarovsky, Shlomo Stern, Hein Wellens, Wojciech Zareba, and Antoni Bayés de Luna: *Electrocardiographic classification of acute coronary syndromes: a review by a committee of the international society for holter and non-invasive electrocardiology.* Journal of Electrocardiology, 43(2):91–103, 2010, ISSN 0022-0736. https://www.sciencedirect.com/science/article/pii/S0022073609002829. 1

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

[6] Scardulla, F., G. Cosoli, S. Spinsante, A. Poli, G. Iadarola, R. Pernice, A. Busacca, S. Pasta, L. Scalise, and L. 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. 2, 3, 8

[7] Lucafó, G. D., P. Freitas, R. Lima, G. da Luz, R. Bispo, P. Rodrigues, F. Cabello, and O. Penatti: *Signal quality assessment of photoplethysmogram signals using hybrid rule-and learning-based models.* Journal of Health Informatics, 15(Especial), 2023. 4, 12

[8] Nemcova, A., E. Vargova, R. Smisek, L. Marsanova, L. Smital, and M. 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`. 5, 27

[9] Deng, J., W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei: *Imagenet: A large-scale hierarchical image database*. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 5, 26, 34

[10] Rice, S. O.: *Mathematical analysis of random noise*. The Bell System Technical Journal, 23(3):282–332, 1944. 7

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

[12] Stehle, R. H.: *A double-sideband shortwave-broadcast signal-quality estimation algorithm*. IEEE Transactions on Broadcasting, 34(2):263–282, 1988. 7

[13] Bayya, A. and M. Vis: *Objective measures for speech quality assessment in wireless communications*. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 1, pages 495–498 vol. 1, 1996. 7

[14] Wang, J. Y.: *A new method for evaluating ecg signal quality for multi-lead arrhythmia analysis*. In *Computers in Cardiology*, pages 85–88, 2002. 7

[15] 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`. 7, 8

[16] Deshmane, A. V.: *False arrhythmia alarm suppression using ecg, abp, and photoplethysmogram*. Master's thesis, Massachusetts Institute of Technology, 2009. 7, 8

[17] Hjorth, B.: *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`. 7

[18] Zhang, P., J. Liu, X. Wu, X. Liu, and Q. 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. 7

[19] Berkaya, S. K., A. K. Uysal, E. S. Gunal, S. Ergin, S. Gunal, and M. B. 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`. 8

[20] 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. 8

[21] Orphanidou, C. and I. 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. 8

[22] Shahriari, Y., R. Fidler, M. M. Pelter, Y. Bai, A. Villaroman, and X. Hu: *Electrocardiogram signal quality assessment based on structural image similarity metric.* IEEE Transactions on Biomedical Engineering, 65(4):745–753, 2018. 8

[23] Moeyersons, J., E. Smets, J. Morales, A. Villa, W. De Raedt, D. Testelmans, B. Buyse, C. Van Hoof, R. Willems, S. Van Huffel, and C. 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`. 8

[24] Huerta, Á., A. Martinez-Rodrigo, V. Bertomeu-González, Ó. Ayo-Martin, J. J. Rieta, and R. 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`. 8, 14

[25] 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`. 8

[26] Papini, G. B., P. Fonseca, X. L. Aubert, S. Overeem, J. W. M. Bergmans, and R. 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. 9

[27] Such, O.: *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. 9

[28] Nabavi, S. and S. Bhadra: *A robust fusion method for motion artifacts reduction in photoplethysmography signal.* IEEE Transactions on Instrumentation and Measurement, 69(12):9599–9608, 2020. 9

[29] Tăuţan, A. M., A. Young, E. Wentink, and F. 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. 9

[30] Zhang, Y., S. Song, R. Vullings, D. Biswas, N. Simões-Capela, N. Van Helleputte, C. Van Hoof, and W. Groenendaal: *Motion artifact reduction for wrist-worn photoplethysmograph sensors based on different wavelengths.* Sensors, 19(3):673, 2019. 9

[31] Ram, M. R., K. V. Madhav, E. H. Krishna, N. R. Komalla, and K. A. 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. 9

[32] Raghuram, M., K. Sivani, and K. A. 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. 9

[33] Janiesch, C., P. Zschech, and K. Heinrich: *Machine learning and deep learning.* Electronic Markets, 31(3):685–695, 2021. 9

[34] Mohagheghian, F., D. Han, A. Peitzsch, N. Nishita, E. Ding, E. L. Dickson, D. DiMezza, E. M. Otabil, K. Noorishirazi, J. Scott, D. Lessard, Z. Wang, C. Whitcomb, K. V. Tran, T. P. Fitzgibbons, D. D. McManus, and K. H. Chon: *Optimized signal quality assessment for photoplethysmogram signals using feature selection.* IEEE Transactions on Biomedical Engineering, 69(9):2982–2993, 2022. 9

[35] Tiwari, A., G. Gray, P. Bondi, A. Mahnam, and T. 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`. 10

[36] Miranda, J. A., C. López-Ongil, and J. 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. 10

[37] Roy, M. S., R. Gupta, and K. D. Sharma: *Photoplethysmogram signal quality evaluation by unsupervised learning approach.* In *2020 IEEE Applied Signal Processing Conference (ASPCON)*, pages 6–10, 2020. 10

[38] Mahmoudzadeh, A., I. Azimi, A. M. Rahmani, and P. 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. 10

[39] Feli, M., I. Azimi, A. Anzanpour, A. M. Rahmani, and P. 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`. 10

[40] Antzelevitch, C. and A. Burashnikov: *Overview of basic mechanisms of cardiac arrhythmia.* Cardiac electrophysiology clinics, 3(1):23–45, 2011. 10

[41] Pereira, T., K. Gadhoumi, M. Ma, R. Colorado, K. J. Keenan, K. Meisel, and X. 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. 10

[42] Pereira, T., K. Gadhoumi, M. Ma, X. Liu, R. Xiao, R. A. Colorado, K. J. Keenan, K. Meisel, and X. Hu: *A supervised approach to robust photoplethysmography quality assessment*. IEEE Journal of Biomedical and Health Informatics, 24(3):649–657, 2020. 11

[43] Pereira, T., C. Ding, K. Gadhoumi, N. Tran, R. A. Colorado, K. Meisel, and X. 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`. 11

[44] Naeini, E. K., I. Azimi, A. M. Rahmani, P. Liljeberg, and N. 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. 11

[45] Zanelli, S., M. A. El Yacoubi, M. Hallab, and M. 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. 11

[46] Gao, H., C. Zhang, S. Pei, and X. 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`. 12

[47] Roy, M. S., B. Roy, R. Gupta, and K. D. 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. 12

[48] Naeini, E. K., F. Sarhaddi, I. Azimi, P. Liljeberg, N. Dutt, and A. 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`. 12

[49] Chen, J., K. Sun, Y. Sun, and X. 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. 13, 14

[50] Lee, H. C. and C. W. Jung: *Vital recorder—a free research tool for automatic recording of high-resolution time-synchronised physiological data from multiple anaesthesia devices*. Scientific reports, 8(1):1527, 2018. `https://www.doi.org/10.1038/s41598-018-20062-4`. 13

[51] Chatterjee, T., A. Ghosh, and S. 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. 13

[52] Liu, D., M. Görges, and S. A. Jenkins: *University of queensland vital signs dataset: Development of an accessible repository of anesthesia patient monitoring data for research.* Anesthesia & Analgesia, 114(3):584–589, 2012. `https://www.doi.org/10.1213/ANE.0b013e318241f7c0`. 13

[53] Roh, D. and H. 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`. 13

[54] Wang, Z. and T. 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. 14, 18, 20

[55] Eckmann, J. P., S. O. Kamphorst, and D. Ruelle: *Recurrence plots of dynamical systems.* World Scientific Series on Nonlinear Science Series A, 16:441–446, 1995. 14, 17

[56] Liu, J., S. Hu, Y. Wang, Q. Hu, D. Wang, and C. 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, 14, 68

[57] Gröchenig, K.: *Foundations of time-frequency analysis.* Springer Science & Business Media, 2013. 14

[58] Freitas, P. G., R. G. De Lima, G. D. Lucafo, and O. 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

[59] Freitas, P. G., R. G. De Lima, G. D. Lucafo, and O. 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

[60] Marwan, N.: *A historical review of recurrence plots.* The European Physical Journal Special Topics, 164(1):3–12, 2008. 17

[61] Afonso, L. C. S., G. H. Rosa, C. R. Pereira, S. A. T. Weber, C. Hook, V. H. C. Albuquerque, and J. 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`. 17

[62] Shankar, A., H. K. Khaing, S. Dandapat, and S. 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. 17

[63] Zhao, Z., Y. Zhang, Z. Comert, and Y. 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. 17

[64] Li, X., T. Zhou, and S. 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. 17

[65] Campanharo, A. S. L. O., M. I. Sirer, R. D. Malmgren, F. M. Ramos, and L. A. N. Amaral: *Duality between time series and networks.* PLOS ONE, 6(8):1–13, August 2011. https://doi.org/10.1371/journal.pone.0023378. 20

[66] Middlehurst, M., J. Large, M. Flynn, J. Lines, A. Bostrom, and A. 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. 30

[67] Dempster, A., F. Petitjean, and G. 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. 30

[68] Zhao, B., H. Lu, S. Chen, J. Liu, and D. Wu: *Convolutional neural networks for time series classification.* Journal of Systems Engineering and Electronics, 28(1):162–169, 2017. 30

[69] Wang, Z., W. Yan, and T. 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. 30

[70] Ismai-Fawaz, H., B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P. A. Muller, and F. 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. 30

[71] Ismail-Fawaz, A., M. Devanne, J. Weber, and G. 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. 30

[72] Ismail-Fawaz, A., M. Devanne, S. Berretti, J. Weber, and G. 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. 30

[73] Schäfer, P.: *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. 30

[74] Middlehurst, M., W. Vickers, and A. Bagnall: *Scalable dictionary classifiers for time series classification*. In Yin, H., D. Camacho, P. Tino, A. J. Tallón-Ballesteros, R. Menezes, and R. Allmendinger (editors): *Intelligent Data Engineering and Automated Learning – IDEAL 2019*, pages 11–19, Cham, 2019. Springer International Publishing, ISBN 978-3-030-33607-3. 30

[75] Middlehurst, M., J. Large, G. Cawley, and A. 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. 30

[76] Schäfer, P. and U. Leser: *Multivariate time series classification with WEASEL+MUSE*. CoRR, abs/1711.11343, 2017. http://arxiv.org/abs/1711.11343. 30

[77] Schäfer, P. and U. 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. 30

[78] Schäfer, P. and U. 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. 30

[79] Bennett, L. A. and Z. S. Abdallah: *RED comets: An ensemble classifier for symbolically represented multivariate time series*. In Ifrim, G., R. Tavenard, A. J. Bagnall, P. Schäfer, S. Malinowski, T. Guyet, and V. 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. 30

[80] Abdallah, Z. S. and M. M. 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. 30

[81] Lines, J. and A. 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. 30

[82] Zhao, J. and L. Itti: *shapedtw: Shape dynamic time warping.* Pattern Recognition, 74:171–184, 2018, ISSN 0031-3203. https://www.sciencedirect.com/science/article/pii/S0031320317303710. 30

[83] Lubba, C. H., S. S. Sethi, P. Knaute, S. R. Schultz, B. D. Fulcher, and N. 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. 30

[84] Christ, M., N. Braun, J. Neuffer, and A. 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. 30

[85] Middlehurst, M., J. Large, and A. 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. 30

[86] Lines, J., S. Taylor, and A. 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. 30

[87] Cabello, N., E. Naghizade, J. Qi, and L. Kulik: *Fast and accurate time series classification through supervised interval search.* In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 948–953, 2020. 30

[88] Deng, H., G. Runger, E. Tuv, and M. 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. 30

[89] Hills, J., J. Lines, E. Baranauskas, J. Mapp, and A. 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. 30

[90] Bostrom, A. and A. 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. 30

[91] Guillaume, A., C. Vrain, and W. Elloumi: *Random dilated shapelet transform: A new approach for time series shapelets.* In El-Yacoubi, M. A., E. Granger, P. C. Yuen, U. Pal, and N. 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. 30

[92] Guillaume, A., C. Vrain, and W. 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. 30

[93] Ayllón-Gavilán, R., D. Guijo-Rubio, P. A. Gutiérrez, and C. 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. 30

[94] Deng, H., G. Runger, E. Tuv, and M. 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. 30

[95] Rodríguez, J. J., L. I. Kuncheva, and C. 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. 30

[96] Dosovitskiy, A., L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. 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, 31

[97] Tu, Z., H. Talebi, H. Zhang, F. Yang, P. Milanfar, A. Bovik, and Y. Li: *Maxvit: Multi-axis vision transformer.* In Avidan, S., G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner (editors): *Computer Vision – ECCV 2022*, pages 459–479, Cham, 2022. Springer Nature Switzerland, ISBN 978-3-031-20053-3. 29, 31

[98] Liu, Z., Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo: *Swin transformer: Hierarchical vision transformer using shifted windows.* In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002, 2021. 29, 31

[99] Liu, Z., H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, F. Wei, and B. 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, 31

[100] He, K., X. Zhang, S. Ren, and J. Sun: *Deep residual learning for image recognition.* In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 31

[101] Xie, S., R. Girshick, P. Dollár, Z. Tu, and K. He: *Aggregated residual transformations for deep neural networks.* In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017. 31

[102] Zagoruyko, S. and N. 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`. 31

[103] Huang, G., Z. Liu, L. Van Der Maaten, and K. Q. Weinberger: *Densely connected convolutional networks.* In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017. 31, 32

[104] Simonyan, K. and A. Zisserman: *Very deep convolutional networks for large-scale image recognition.* In Bengio, Y. and Y. 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`. 31, 32

[105] Iandola, F. N., S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer: *Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5MB model size*, 2017. `https://openreview.net/forum?id=S1xh5sYgx`. 31, 32

[106] 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`. 31, 32

[107] 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`. 31, 32

[108] Howard, A., M. Sandler, B. Chen, W. Wang, L. C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le: *Searching for mobilenetv3.* In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019. 31, 32

[109] Tan, M. and Q. V. Le: *Efficientnet: Rethinking model scaling for convolutional neural networks.* In Chaudhuri, K. and R. 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`. 31, 32

[110] Tan, M. and Q. V. Le: *Efficientnetv2: Smaller models and faster training.* In Meila, M. and T. 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`. 31, 32

[111] Ma, N., X. Zhang, H. T. Zheng, and J. Sun: *Shufflenet v2: Practical guidelines for efficient cnn architecture design.* In Ferrari, V., M. Hebert, C. Sminchisescu, and Y. Weiss (editors): *Computer Vision – ECCV 2018*, pages 122–138, Cham, 2018. Springer International Publishing, ISBN 978-3-030-01264-9. 31, 32

[112] Krizhevsky, A., I. Sutskever, and G. 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`. 31, 33

[113] Liu, Z., H. Mao, C. Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie: *A convnet for the 2020s.* In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976, 2022. 31, 33

[114] Radosavovic, I., R. P. Kosaraju, R. Girshick, K. He, and P. 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, 2020. 31, 33, 56

[115] Yang, T. J., A. G. Howard, B. Chen, X. Zhang, A. Go, M. Sandler, V. Sze, and H. Adam: *Netadapt: Platform-aware neural network adaptation for mobile applications.* In Ferrari, V., M. Hebert, C. Sminchisescu, and Y. 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`. 32

[116] Kingma, D. P. and J. Ba: *Adam: A method for stochastic optimization.* In Bengio, Y. and Y. 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`. 34

[117] Cohen, J.: *A coefficient of agreement for nominal scales.* Educational and Psychological Measurement, 20(1):37–46, 1960. `https://doi.org/10.1177/001316446002000104`. 34

[118] Faouzi, J. and H. 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`. 38

[119] Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala: *PyTorch: An Imperative Style, High-Performance Deep Learning Library.* Curran Associates Inc., Red Hook, NY, USA, 2019. 38

[120] Szegedy, C., W. Liu, Y. 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`. 64

[121] Dempster, A., D. F. Schmidt, and G. 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`. 64

[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`. 65

[123] Wen, Q., L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. 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`. 65

[124] Suzuki, G. C. and P. Freitas: *On the performance of composite 1d-to-2d projections for signal quality assessment.* In *Anais do XXIV Simpósio Brasileiro de Computação Aplicada à Saúde*, pages 319–330, Porto Alegre, RS, Brasil, 2024. SBC. `https://sol.sbc.org.br/index.php/sbcas/article/view/28828`. 66