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

Projection Based Photopletysmography Signal Processing for Its Quality Assessment

Guilherme C. 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 Photopletysmography Signal Processing for Its Quality Assessment

Guilherme C. Suzuki

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

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

Prof. Dr. Donald Knuth
Stanford University

Dr. Leslie Lamport
Microsoft Research

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

Brasília, @ dia @ de @ mês @ de 2024

Dedicatória

Na *dedicatória* o autor presta homenagem a alguma pessoa (ou grupo de pessoas) que têm significado especial na vida pessoal ou profissional. Por exemplo (e citando o poeta): *Eu dedico essa música a primeira garota que tá sentada ali na fila. Brigado!*

Agradecimentos

Nos *agradecimentos*, o autor se dirige a pessoas ou instituições que contribuíram para elaboração do trabalho apresentado. Por exemplo: *Agradeço aos gigantes cujos ombros me permitiram enxergar mais longe. E a Google e Wikipédia.*

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), por meio do Acesso ao Portal de Periódicos.

Resumo

O *resumo* é um texto inaugural para quem quer conhecer o trabalho, deve conter uma breve descrição de todo o trabalho (apenas um parágrafo). Portanto, só deve ser escrito após o texto estar pronto. Não é uma coletânea de frases recortadas do trabalho, mas uma apresentação concisa dos pontos relevantes, de modo que o leitor tenha uma ideia completa do que lhe espera. Uma sugestão é que seja composto por quatro pontos: 1) o que está sendo proposto, 2) qual o mérito da proposta, 3) como a proposta foi avaliada/validada, 4) quais as possibilidades para trabalhos futuros. É seguido de (geralmente) três palavras-chave que devem indicar claramente a que se refere o seu trabalho. Por exemplo: *Este trabalho apresenta informações úteis a produção de trabalhos científicos para descrever e exemplificar como utilizar a classe LaTeX do Departamento de Ciência da Computação da Universidade de Brasília para gerar documentos. A classe* `UnB-CIC` *define um padrão de formato para textos do CIC, facilitando a geração de textos e permitindo que os autores foquem apenas no conteúdo. O formato foi aprovado pelos professores do Departamento e utilizado para gerar este documento. Melhorias futuras incluem manutenção contínua da classe e aprimoramento do texto explicativo.*

Palavras-chave: Projeção de sinais, IA, Aprendizado de máquina, Aprendizagem profunda

Abstract

Signal quality assessment is essential to health monitoring applications since good signal quality is needed to inform reliably the medical conditions of the patient. In order to do so, machine learning algorithms such as convolutional neural networks may be applied. However, the signal needs to be transformed into a 2D representation, which can be done by the use of time series imaging techniques, such as Gramian Angular Field, Markov Transition Field, and Recurrence Plot, and also by aggregating their results, which we called Projection Mix. After preprocessing the dataset, BUT PPG, into those images, various convolutional neural networks were trained and tested using such data, while also choosing hyperparameters using heuristic searching. The results reveal that the projections Recurrence Plot and Projection Mix performed generally better than the Gramian Angular Field and Markov Transition Field, despite the existence of some exceptions. An implementation of the method described in this paper can be found at https://gitlab.com/lisa-unb/projection-based-biological-signal-processing.

Keywords: photoplethysmography, convolutional neural networks, signal quality assessment, biological signals, time series imaging

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introduction

Capítulo 2

Literature Review

This chapter will present the state of the literature of the Signal Quality Assessment of cardiological signals problem, using a top to bottom approach, finishing on this thesis work scope. Then, this chapter will expose this work contribution.

2.1 Signal Quality Assessment

The Signal Quality Assessment (SQA) challenge is not exclusive to the medical domain. Actually, its origins are related to the old comunication systems, when researchers published their first works on the theory of information in the 1920s and 1930s. One example of such works is the "Mathematical analysis of random noise", by Stephen O. Rice, in which he analyzes the statistical properties of comunication device noises [1]. Another example is the work of Claude E. Shannon, "A mathematical theory of communication", which introduces fundamental concepts in comunication systems [2]. In that millennium, studies already used the concept of SQA. We can see samples of this in the 1980s, such as the work of R. H. Stehle, which proposed an algorithm for assessing the quality of shortwave broadcast signals, trying to objectively measure the human perception of the signal mensage intelligibility [3]. Some of its conclusions are useful in the SQA in clinical contexts, such as the high degree of subjectivity in the human idea of quality. This makes labeling datasets properly fundamental to reflect this concept of quality in the proper evaluation of the developed SQA algorithms.

Past the 20th century, the SQA of physiological signals begin to become popular. Particularly, the early 2000s decade had several works in the subject. One of them is the work of **(author?)**, which proposed a ECG assessment method based on the difference between the areas of distinct QRS complexes [4]. The work propose comparing the cummulative histograms of different ECG leads to assess its qualities. Later, **(author?)** suggested the combination of different quality metrics by measuring their inner and between ECG lead

ammount of agreement, giving one final metric [5]. Another work, by **(author?)**, posed the thresholding based on the Hjorth parameters to assess the quality of PPG signals [6]. Afterwards, **(author?)** elaborated a ABP signal quality assessment metric based on the End Diastole Slope Sum and Slow Ejection Slope Sum features. Through those works, researchers proposed quantifying the SQA in a metric. A common name that they used to refer to this metric was Signal Quality Indice (SQI).

2.1.1 The Signal Types

Among the variety of physiological signals, the Electrocardiogram (ECG) is prevalent in the literature. This signal has multiple applications, such as disease classification, heartbeat type detection, biometric detection and emotion recognition [8]. There are a plenty of works in the SQA of ECG signals. In one of them, **(author?)** 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 chosing a target lead, feeding a FFNN with array of derivatives of all leads to reconstruct the targeted lead and finally comparing the original target lead to its artificial version with correlation analysis. In 2017, **(author?)** introduced a feature based on the extraction of the Heart Rate Variability of ECG signals. The method decomposes this new signal into wavelets with different frequency ranges and calculates the each of them entropy, forming a feature vector. This vector feed a SVM, which classifies the signal as acceptable or not. Later, **(author?)** 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, **(author?)** proposed transforming the signal using the Auto Correlation Function and extracting simple features from it [12]. In the current year, **(author?)** generated phase space plots, such as Poincaré plots and First Order Difference graphs, and discretized them into a grid where each cell is the logarithm of the number of points contained in that cell [13]. Thus, the SQA literature for ECG is already well developed.

However, an alternative to the ECG, Photoplethysmograph (PPG), has only increased in popularity. In fact, its number of related articles published from 2013 to 2023 has increased in 176% [14]. This signal also elevated in presence in the SQA literature. A sample of this literature is the work of **(author?)**, which poses the measurement of a SQI through the application of the Dynamic Time Warping techinique [15]. It finds the optimal path cost on a distance matrix which encodes the differences between each pair of points of the input and a template, reference of a good signal. **(author?)** added this techinique SQI to others and feeded them to a MLP and to a self-made function,

predicting a unique SQI. Experiments on private annotations on the MIMIC II dataset resulted on the MLP achieving the highest accuracy, 95.2% [15]. Therefore, it is possible to achieve good accuracy in the SQA of PPGs.

In the SQA literature, the PPG showed to be usefull in the Cardiac Arrhythmia identification applications. The Cardiac Arrhythmia (CA) is the presence of an anomalous cardiac rate, such as fast, slow or irregular HR moments. Its cause is abnormalities in the cardiac nervous system. 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, such as the naive HR estimation based on QRS complex peaks. This assumption can result in signals with arrhythmia being reject as unreliable signals, leaving those special patients undiagnosed. **(author?)** conducted experiments on a private dataset that contains cases of Atrial Fibrillation (AF), a type or arrhythmia [16]. In this experiment, 40 features used in previous studies fed a SVM, which achieved an average accuracy superior to 94% [16]. That accuracy was far higher than other existing methods, which the researchers also tested [16]. Therefore, abundant feeding a machine learning algorithm with features and training it on datasets with arrhytmia 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, **(author?)** feeded several features to three classifiers: SVM, K-NN and Decision Tree [17]. Similarly to the antecedent study, experiments on a private dataset demonstrated that the SVM was the best of all classifiers and it obtained above 95% suprassing the methods of other studies. The second study also included the SVM method, but added to the experiment deep learning models [18]. The study contemplated both 1d and 2d deep learning models, with the first recieving the raw signal and the second recieving 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 supprass conventional methods even with the presence of arrythmic events. The next section will further explore the use of deep learning in the literature.

2.1.2 The Deep Learning Approach

As already mentioned, Deep Learning (DL) has the potential to achieve a higher accuracy than feature based models, even in the context of CA. On one hand, differently of hand-crafted features, DL automatically extract features from the input signal, creating models that are adaptable to different dataset training contexts. Additionally, a high quality dataset can provide resources for the DL model to be robust to variations on the signal conditions. On the other hand, not only it creates a black-box that does not explain the

reasons why the model atributed a certain SQI, but also requires large ammounts of data to proper adjust the model parameters. Depite this, DL are worth exploring since it can provide the accuracy and robustness that the medical applications require.

In this context, several studies proposed the application of CNNs, one-dimensional DL models. For instance, **(author?)** applied a self-designed CNN to extract a binary SQI [19]. Tests on a private dataset with data of three devices lead to 85% F1-score for the "Reliable" class. Alike, **(author?)** employed a self-made CNN, but examined the effect of transfer learning as well [20]. They conducted the experiment on three private datasets. It beggan training the model mostly on one dataset, in which it achieved an accuracy of 99.8%. Then, for each remaining database, they fine-tuned the the model with little training and tested on it. This procedure resulted on the 93% and 81% accuracies on the second and third datasets, respectively. Additionally, the model trained solely on the second database scored lower, 86%. Those results indicate that not only we can use CNN to achieve high accuracy but also we can transfer its learned features over different databases to improve its performance.

However, some works go beyond the simple application of CNNs. The research of **(author?)**, per example, introduced a hybrid-model for quality assessment, which combines a CNN with a rule-based approach. This rule can bypass the utilization of the CNN by verifying if the signal min to max distance is less than an threshold. They determined the threshold by methods such as Last Value Thresholding and Nearest Value Thresholding. The researchers did it to avoid the unecessary power usage 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 mantaining similar prediction scores if compared to the CNN without the rule component [21]. Therefore, we can achieve particular advantages by combining the CNN with other methods.

Besides CNNs, works explored the use of alternative DL models such as recurrent networks. **(author?)** proposed the application of a Long Short-Term Memory network for real time SQA, giving a SQI for each point in the signal. For the experiments, they labeled private and public datasets by applying Blind Source Separation to generate from each PPG signal one high-quality signal and one low-quality signal. When compared to baseline SQIs and existing models, it achieved competitive accuracy, while being lightweighted and enoughly fast to predict in real-time [22]. Therefore, recurrent networks can suit well for real-time SQA tasks.

Additionally, other researchers proposed the application of 2D CNNs. One of those, **(author?)**, proposed, initially, the transformation of the input signal into a Short-Time Fourier Transform spectogram. Then, this image feeds a 2D CNN which classifies the quality of the signal. The researchers annotated the VitalDB database and, then, con-

ducted experiments that lead to the proposed method achieving a better accuracy than four chosen baseline models. Its value was 98,3%. Therefore, transforming the input into a image can result in improved accuracy. The next section will explore the time series imaging scope of the SQA literature.

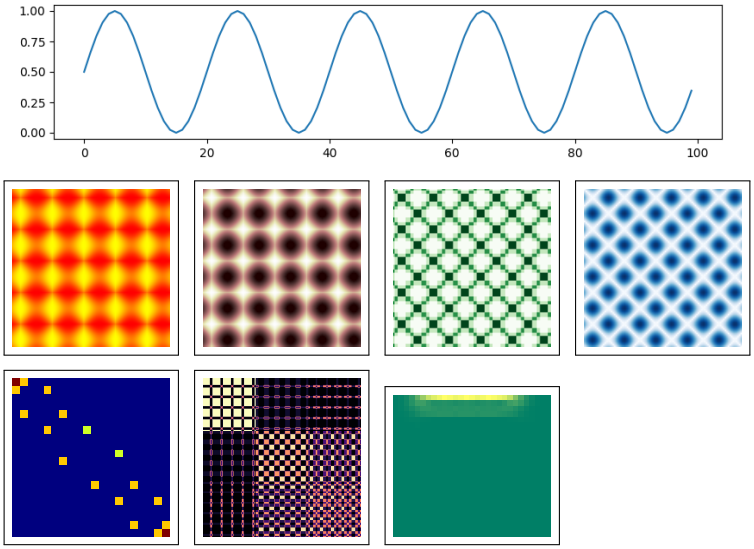

Figura 2.1: A signal and its various projection obtained by several methods. In the first line, from the left to the right, the methods are: Gramian Angular Diference Field, Gramian Angular Summation Field, Markov Transition Field and Recurrence Plot. The methods of the second line are, from the left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

2.1.3 The Time Series Imaging Techinique

Several works in the literature propose transforming the input signal into a image and, then, feeding it to a computer vision model. The figure 2.1 shows some of those transformations. The works that this paragraph mentions all used CNNs as a classifier of the SQI. **(author?)** transformed the signal into Quantum Pattern Recognition images for PPG SQA. This resulted in 98,3% accuracy on the University of Queensland vital signs dataset with labels from the researchers, scoring above baseline models and an existing DL method [24]. **(author?)** embeded the signal into a Recurrence Plot matrix for PPG SQA. On a private datset, the method achieved 97,5% accuracy [25]. Therefore, 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, **(author?)** feeded a Vision Transformer with a Recurrence Plot or a Markov Transition Field, achieving, respectivelly, 89,9% and 90,3% accuracy on a private dataset [26]. **(author?)** also feeded images with a Vision Transformer, but used Gramian Angular Fields. The proposed approach reached 92,2% accuracy on a private dataset [27]. **(author?)** proposed the use of Multiscale Markov Transition Fields, version which concatenates the signal first and second derivatives. This projection feeded a self-made computer vision 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 [28]. Thus, the combination of a generic computer vision model with a projection method, such as Recurrence Plot, Gramian Angular Field or Markov Transition Field, can achieve a decent accuracy, while it is possible to apply the multiscaling techinique to improve some of those projections accuracy.

2.2 This work contribution

This section will explain this work contribution. Firstly, this work proposes a new and effective approach for encoding time series. It aggregates different projections into a composite hyperspectral image. This aggregation showed to be better than using one of the projections methods alone. Secondly, this work also evaluates the proposed approach in combination with a great ammount of computer vision models, which none of the above-mentioned works did. That gives insights on which types of models are ideal for the time series matrix embedding techinique. Thirdly, this thesis also attemp a novel idea of transfer learning with source in a dataset out of the SQA domain, in this case, the ImageNet dataset. That has the potential to reduce the need of a large ammount of labeled PPG data. Finally, this thesis did experiments on a publicly avalliable and

labeled dataset, Brno University of Technology Smartphone PPG Database (BUTPPG), which ensures the reproductibility and comparability of this work experiment. The lack of reproductibility was a noticeable problem in the literature that this chapter reviewed.

Capítulo 3

Method

Capítulo 4

Experiments

4.1 Experimental Setup

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

4.1.1 The dataset

As for every machine learning task, we need a dataset to provide data to feed the predictive models for their parameters fitting, molding them to the domain of the specific task. In this work, the task of assessing the quality of the signal is cleary a supervised classification problem, that is, can be described as the problem of finding a function that best defines a predefined set of pairs of variables and label, (X, y). In this scope, the pair corresponds to the signal itself mapped to its quality label, "Good" or "Bad". For the purpose of training the predictive methods and evaluating their ability to fit to the problem of classifying the quality of heartbeat time series, I employed the BUTPPG [29] dataset.

The Brno University of Technology Smartphone PPG Database (BUTPPG) is a publicly available database produced by the Department of Biomedical Engineering of the the Brno University of Technology. It contains samples of PPG signals, its quality labels and its heat rate estimations. Those signals were extracted using a low-cost method: recording with the camera of a smartphone. To be more precise, they recorded the subject's index finger, covering the lens of the camera and its LED light. Then, they measured, for each video frame, the average of the intensities of the red channel of every image pixel, resulting in a time series of averages. Finally, they inverted the signal. The Figure 4.1 shows examples of the results of such a sequence of procedures.

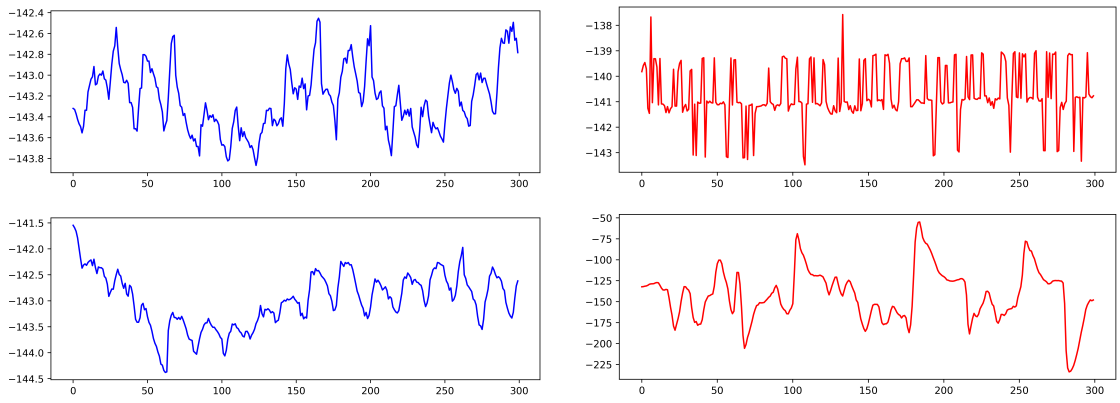

Figura 4.1: Samples of the four signals of the subject with id equal to 11 (where the first subject has id 0). The "Good" signals color is blue, while the "Bad" signals are red. In those graphs, the vertical dimension is the average of all pixels intensity in a frame, while the horizontal dimension is the time instant in frames. Notice that, since those records are 10 seconds long, it is implied that the sampling frequency is 30 Hz.

They done this method of obtaining PPG signals 48 times, ammount distributed equally between 12 subjects. That is, for each subject, they did 4 measurements. Moreover, they did those recordings in two possible situations: one which the subject sat down and stayed static, case which the quality label "Good" was probable; and other in which the subject walked, hence, case likely to be a "Bad" recording. That distinction is relevant, since the walking situation occurred only 1 time for each subject, biasing the labels proportion to be nearly 25% of "Bad" ones. Therefore, this dataset is imbalanced, factor that we need to handle in our experiment.

As for the definition of the signal quality labels, they designated specialists to estimate the heart rate associated with the PPG signals, with the help of a software specialized for the analysis. Then, the organizers compared the number they gave with the one given by a gold standard method that, instead of using the PPG signal, used an ECG recording. The measurer syncronized the ECG mannually. If the specialist measured with an error less or equal than 5 bpm, then the organizers considered the estimative correct. Finally, if 3 specialists of 5 gave correct estimations, the organizers considered the quality of the PPG signal as correct. Thus, the dataset "Good" labels, in essence, identify if a signal is human-readable.

4.1.2 The dataset split

Machine learning tasks also requires the separation the dataset into fragments. One of them is the training dataset, used for the models parameters adjustment. Another is the test dataset, used for evaluating the models efficiency. An optional one is the validation

dataset, used to choose the set of best hyperparameters of a trained models. In this experiment case, to define the training-test splits, I used a cross validation method named Leave One Subject Out (LOSO), which partitions the dataset into k pieces and k train-test splits. It makes the i-th train-test split assigning the i-th piece as the test dataset, lefting the other k-1 pieces as the training dataset. In the BUTPPG case, the k value is 12, the number of subjects. Notice that the smaller unity of division is the subject, not the signals that are associated with it. I did it to increase the difference between training samples and testing samples. Since the dataset is small, such a split method allows the maximum provect of the avaliable resources, because it uses every sample in the dataset as a test object at least once, without biasing the results. Additionally, the training datset was divided producing a validation dataset of size 3, with usage that will be explained later.

4.1.3 The models

To evaluate the proposed projection-based framework and find particular overperforming cases, it is necessary to involve a big ammount of machine learning models. Firstly, I compared the projection-based framework with other existing approaches, by utilizing the Aeon-toolkit python library [30], with models listed in table 4.2. Furthermore, I combined the proposed method with a wide variet of classification CV models. For that, I used the Pytorch python library [31], which aggregates a wide variety of neural network design strategies. Those strategies varies from simply convolutional networks to vision transformers. The table 4.1 lists all CV models involved in the experiment. With that, I submited the method to an abrangent set of scenarios.

However, even though I defined the models, my setting lacks the choice of their hyperparameters. Those kind of parameters are high level parameters that does not change during the training procedure. For the Aeon models, I chosed the default hyperparameters provided by the library. But, for the computational vision model, while I set most of them to default, I did hyperparameter search for the learning rate hyperparameter, used by the optimizing algorithm. I did that seach by applying the Optuna python library [82], which does heuristic search over the set of all parameters requested dynamically in the user code. That libray prunes ramifications of the search-space tree with a variety of methods, of which I chose the median prunning. In our case, the score that guides this heuristic is the accuracy score, the ratio of hits over the number of samples. I measured it in the validation dataset of size 2, that resulted from a simple random split. This functionallity allowed me to find a near-optimal combination of parameters without testing cases exhaustively, using as heuristic the score of the model in the validation dataset.

Classification	Model	Reference
Transformer	Vision Transformer	[32]
	MaxViT	[33]
	Swin Transformer	[34]
	Swin Transformer V2	[35]
Residual Net	ResNet	[36]
	ResNeXt	[37]
	WideResNeXt	[38]
Extreme Net	DenseNet	[39]
	VGG	[40]
	SqueezeNet	[41]
Mobile Oriented	MNASNet	[42]
	MobileNet V2	[43]
	MobileNet V3	[44]
Efficiency Oriented	EfficientNet	[45]
	EfficientNet V2	[46]
	ShuffleNet V2	[47]
Diverse	AlexNet	[48]
	ConvNeXt	[49]
	RegNet	[50]

Tabela 4.1: Computer Vision models list, containing its citations.

4.1.4 The training strategy

Given the before-mentioned models, the dataset and its divisions, I need to estabilish the training method for the models parameters adjustment. Since the Aeon implementation already contained a default training procedure, I only estabilished the fitting framework for the pytorch computational vision models and the data feeding method. I feed the models by loading the signal data and, before transforming them, applyed random over sampling, since the dataset was unbalanced. After doing the projection transform, I begin by loading pre-trained model weights provided by the Pytorch. This learn transfering originated from training the models in the ImageNet [83] dataset. Following that, I did the pytorch models fitting using the Adam optimizing algorithm [84] to minimize the Cross Entropy loss function. My implementation of the training startegy did this optimization cycle with an ammount of epochs depending on an median-based early stop technique. The formula bellow gives the score of the n-th epoch:

$$\textit{Early-Stop-Score}(n) = M^{9}_{i=0}(|l_{n-i} - M^{9}_{i=0}(l_{n-i})|) \tag{4.1}$$

Where l_k is the loss value (the Cross Entropy Loss) of the k-th epoch and $M^{k}_{i=0}(f(i))$ is the median of the values of $f(i)$ that result from the variation of i from 0 to k. In other

Classification	Model	Reference
Convolution-Based	Arsenal	[51]
	Rocket Classifier	[52]
Deep Learning	CNN Classifier	[53]
	FCN Classifier	[54]
	MLP Classifier	[54]
	Inception Time Classifier	[55][56]
	Individual Inception Classifier	[55][56]
	LITE Time Classifier	[57]
Dictionary-Based	BOSS Ensemble	[58]
	Contractable BOSS	[59]
	Individual BOSS	[58]
	Individual TDE	[60]
	MUSE	[61]
	TemporalDictionaryEnsemble	[60]
	WEASEL	[62]
	WEASEL V2	[63]
	REDCOMETS	[64][65]
Distance-Based	Elastic Ensemble	[66]
	K-Neighbors Time Series Classifier	—
	Shape DTW	[67]
Feature-Based	Catch-22 Classifier	[68]
	Summary Classifier	—
	TS Fresh Classifier	[69]
Hybrid	HIVE-COTE V1	[70][51]
	HIVE-COTE V1	[51]
Inverval-Based	Canonical Interval Forest Classifier	[71]
	DrCIFClassifier	[51]
	Random Interval Spectral Ensemble Classifier	[72]
	Supervised Time Series Forest	[73]
	Time Series Forest Classifier	[74]
	Random Interval Classifier	—
Shapelet-Based	Shapelet Transform Classifier	[75][76]
	RDST Classifier	[77][78]
Ordinal Classification	Individual Ordinal TDE	[79]
	Ordinal TDE	[79]
Other	Continuous Interval Tree	[80]
	Rotation Forest Classifier	[81]

Tabela 4.2: Non-Computer Vision models list, containing its references.

words, the formula calculates the median absolute deviation of the last 10 losses values using the median as the central value. If *Early-Stop-Score(n)* ≤ 0.1, the training stops in the n-th epoch. With that estabilished, I am only left to determine the metrics to be measured.

4.1.5 The measurements

Finally, I chose the metrics, used to evaluate objectivelly efficacy of the solution. I separated the metrics that I used into two groups: prediction metrics, which measures the quality of the models signal quality assessments; and benchmarking metrics, which measures the resources usage and the model speed. As the prediction metrics, I used the Cohen kappa score, the f1-score, and the precision. The Cohen kappa score, in binary classification tasks, measures the relation between the obtained accuracy acc_o and the expected accuracy acc_e. The following equations define those accuracies and the Cohen Kappa score, in terms of confusion matrix cell values:

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

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

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

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

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

In sequence, the Precision is a metric that measures the ratio of hits in the set of positive predictions. In our context, a higher Precision imply that the predictor aproved a low ammount of "Bad" signals, which is desireable in applications where we do not want to show to the user wrong results. 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 ammount of "Good" signals that would feed the application. This is an desireable quality when

we want to provide constant feedback to the user. The following equations define those metrics:

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

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

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

Therefore, the Cohen Kappa provides an overall sense of accuracy, the F1-Score suggest the usability level of the model and the Precision indicates the security of the predictor.

Regarding the benchmarking metrics, I measured the memory usage of the model in bytes and the inference time (added to the 1d-to-2d projection time for the projection-based models) in seconds. The memory measurement is important since practical applications of heart rate estimations often imposes hardware constraints that limits the allowed memory occupation. Additionally, the inference time, even though less important than the antecedent metric, is desireable for an almost-instant evaluation of the result, making the application more responsive.

4.1.6 The overall schema

The Figure 4.2 expresses how I did the experiment for the CV models. A first observation is that the main difference from the framework applied to non-CV models is that the 1d to 2d conversion acts as a frontier between the dataset and the rest of the components. That means the non-CV models experiment can be expressed using almost the same schema by removing that conversion block. With that, the experiment began by the Hyperparameters Selecion, splitting the BUTPPG dataset using simple division for the purpose of, subsequentially, selecting the best hyperparameters of the CV. With the best hyperparameters chosen, all models, including the non CV models, will be evaluated using the LOSO strategy. For each fold, I subjected the model to a training procedure that repeats epocs of training-and-validating until the early stopper interferes. Then, the model is tested, producing the metrics for that fold.

4.1.7 The implementation details

Presented the general concepts, in this section I discuss some details, following the practical sequence of events. I begin describing the process until the training. I done it by using

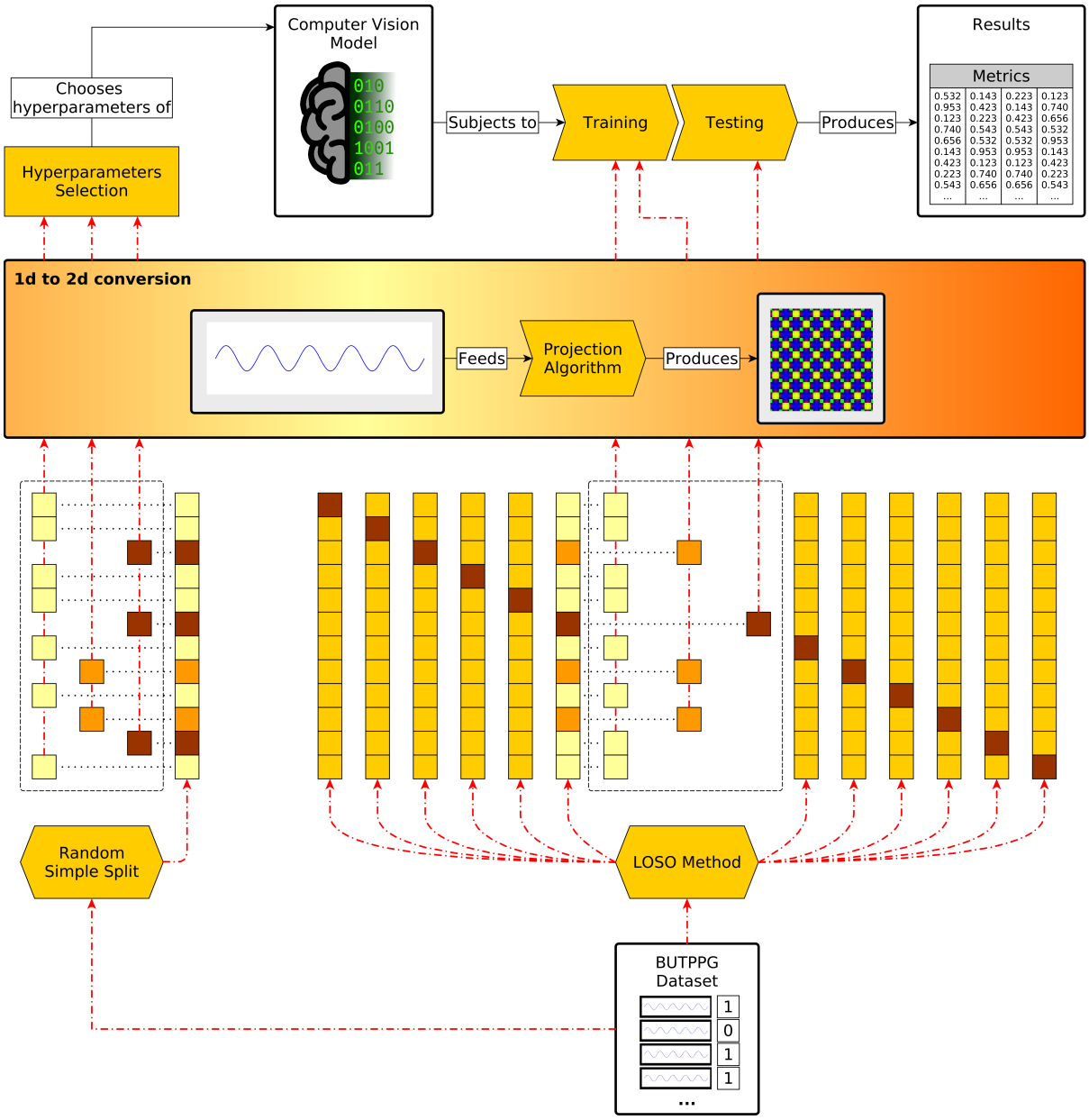

Figura 4.2: The framework of the experiment. Red dotted arrows indicates data flow that sources from the BUTPPG Dataset. Notice that the figure presents the training-testing cycle for only one of the twelve folds. The above schema is replicated for each combination of Computer Vision Model and Projection Algorithm.

the Python multithreading data feeding solution, named Data Loader[1]. I configured it to load in batches of size 32. But, before loading the batches, I did the training dataset balancing using the Imbalanced Learn [85] implementation of the random oversampling[2]. Then, the batches are loaded transforming the signal into its images using the projection algorithm implementations of the PyTS Python library [86]. Even though the signals are now 2d, the width, height and number of channels dimensions are incompatible with the networks inputs. To solve this, I applyed the Pytorch resize transform[3] to ajust the width and the height; and, to ajust the model to the number of channels, I added an additional layer with 4 channels that I use in a pointwise convolution that outputs to the original model input layer, which contained 3 channels. For some models, I replaced the input layer with one containing the desired number of channels.

In sequence, I describe the training and testing procedures. For comporting the CV models in both procedures, I used the following GPUs ????. For training, I used the Pytorch implementation of the Adam optmization algorithm[4], which uses the gradients evaluated by the Pytorch autograd engine [87]. The loss class (which, in our case, is the Cross Entropy Loss[5]) backpropagates the gradients based of the model foward pass errors. For the models testing, I used the Sklearn [88] implementation of the metrics[6]. For model memory measurement, I counted the summation of the size of each parameter and buffer tensors in the CV models, while for the non-CV models, I used the asizeof function[7] of the Pympler library [89]. Finally, I descibe the inference time measurement, for which I extracted 500 measurement samples. For the non-CV models, I used the Python time method[8], of the time library, to measure instants of time. For the CV models, I marked the time instants by using CUDA events interface provided by Pytorch[9], while, before measuring, performing 500 iterations to warm-up the GPU.

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

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

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

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

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

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

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

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

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

4.2 Experimental results

In this section, I analyze the results will by comparing the metrics in the following order: firstly, I verify if the reescaled Cohen Kappa Score value is superior o equal to 90%. Secondly, the F1-Score; and, lastly, the Precision score. Moreover, I considered the trade-off between models with respect to the memory consumption and the inference time, in that order. Since I considered a large set of models, I did the comparisions in sections. In each section, I consider one of the six before-presented computer vision model classifications: Transformers, Residual Nets, Mobile Oriented, Extreme Nets, Efficiency Oriented, and Diverse. In each of those sections, I selected the best combinatons of model variant and projection method, if at least one of them is sufficiently good. Subsequently, I will choose the best non-CV models present in the Aeon toolkit. Lastly, I will determine the final best choices, also discussing the differences between the projection methods.

4.2.1 Categories analysis

Transformers

I tested four transformers the Vision Transformer, the Multi-axis Vision Transformer, the Shifted Windows Transformer and its second version. The Vision Transformer is the base model of the others. It transforms the visual input into into a word where the letters are linear embedings of subimages obtained by particioning the original image in a grid-like shape. Then, the other modules increments the original by using multiple layers and by changing the attention mechanisms. For instance, the MaxViT employs achitectural blocks where each alternates between two self-atention modes: grid attention, a mode of high granularity, and block attention, a mode of low granularity. The Swin Transformer, on the other hand, changes its attention in layer-level, by merging patches of the antecedent layer into a new one, and in block-level, by shifting the self-attention windows into different positions. Finally, the Swin Transformer V2 propose several specific improvements to the older version.

About the results, we can see on the tables 4.3, 4.4, 4.5, and 4.6 that only the Swin Transformer V2 and the Vision Transformer variants obtained a Cohen Kappa score superior than 90%. Specifically, for the Swin Transformer V2, only the standard variant combined with the Mix method achieved such a score. For the Vision Transformer, only the large and base variants suprassed the limit. The variants with patch size 32×32 did so by using the Mix, while the variants with patch size 16×16 not only reached the score when combined with Mix but also with RP. Since both of the models reached a similar score, we observe the benchmarking metrics. We see in the Table 4.7 that the Swin Transformer V2 S uses considerably less memory than the Vision Transformer.

On the other hand, we observe in Figure 4.3 that the Swin Transformer V2 S have a slower inference speed if compared to the Vision Transformer variants. However, because I consider the memory consumption as a factor of major importance, I selected for this section the Swin Transformer V2 standard variant with Mix.

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

Tabela 4.3: Averages and standard deviations of the folds evaluation for the VisionTransformer variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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
	Mix	**0.875 ± 0.169**	0.921 ± **0.098**	**0.944 ± 0.130**

Tabela 4.4: Averages and standard deviations of the folds evaluation for the Maxvit variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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

Tabela 4.5: Averages and standard deviations of the folds evaluation for the SwinTransformer variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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

Tabela 4.6: Averages and standard deviations of the folds evaluation for the SwinTransformer V2 variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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

Tabela 4.7: Memory size in Mega Bytes of each of the Transformers models variants.

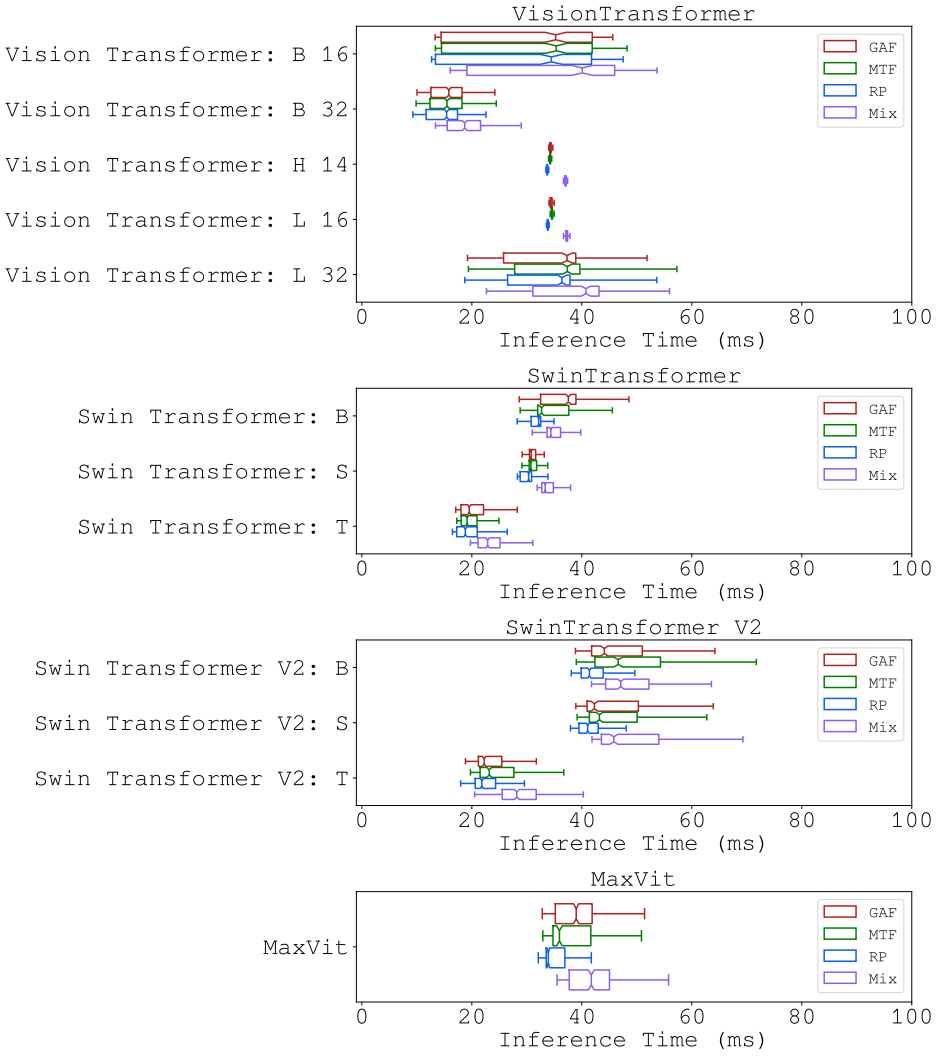

Figura 4.3: Inference time in miliseconds of each Transformers models variants.

Residual Nets

This section analyzes the ResNet itself and its variations. The ResNet is a model that introduced the residual conections, links between non-adjacent layers that bypasses intermediate layers. As its variations, I analyzed two: the Wide ResNet and the ResNeXt. The first one widens the original net by increasing the number of channels per block, with the intention of providing an alternative to increasing the layer depth. The second one employs a multipath philosophy, agreggating the paths by an additive operation. As oposed to the antecedent, it avoids increasing the width and the depth of the network by providing an additional dimension to increase.

Of those models, only the original ResNet and the Wide ResNet achieved a suficient Cohen Kappa score, as we can see in the tables 4.8, 4.9 and 4.10. From the ResNet variants, the ones with 50 and 101 layers reached that score. Solely the Wide ResNet with 101 layers and 2 convolutions per block sucessfully score above the limit. Those models variants got those scores by employing the Mix projection method. Comparing these last variants, we conclude that the Wide ResNet 101-2 variant with the Mix method obtained the best score, both for Cohen Kappa and F1 scores. However, that Wide ResNet posesses the largest memory occupation, according to the table 4.11. It also spends the fourth higher inference time, as seen in the figure 4.4. Despite those disavantages, the Wide ResNet 101-2 variant with the Mix was the choice of this section, because I prioritize the classification metrics over the benchmarking ones.

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
	Mix	**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
	Mix	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
	Mix	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
	Mix	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
	Mix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± 0.130

Tabela 4.8: Averages and standard deviations of the folds evaluation for the ResNet variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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
	Mix	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
	Mix	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
	Mix	0.792 ± 0.234	0.919 ± 0.087	0.882 ± 0.148

Tabela 4.9: Averages and standard deviations of the folds evaluation for the ResNeXt variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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

Tabela 4.10: Averages and standard deviations of the folds evaluation for the Wide ResNet variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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

Tabela 4.11: Memory size in Mega Bytes of each of the Residual Nets models variants.

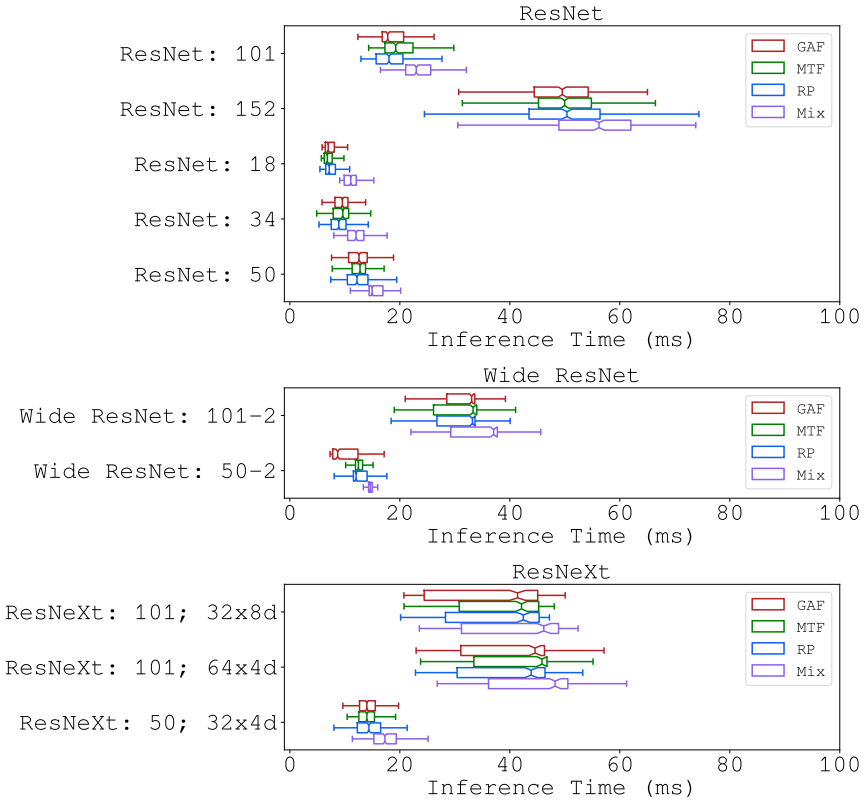

Figura 4.4: Inference time in miliseconds of each Residual Nets models variants.

Mobile Oriented

This section explores the mobile oriented networks, that is, networks designed specifically for mobile hardware constraints. I tested three models: the MNASNet, the Mobile Net V2 and the Mobile Net V3. In cronologial order, the Mobile Net V2 comes first. This newtork proposes several achitectural changes to use less memory while mantaining accuracy, with inverted residual blocks being one of them. This proposed alteration swaps places of the high-channeled and low-channeled layers, which causes the connection between layers with a lower ammount of channels, reducing the number of parameters in that block. After the antecedent, researchers introduced the MNASNet. Its name stands for Mobile Neural Architecture Search. It select blocks to fit in a predefined achitectural skeleton, optimizing the model performance on real-world mobile hardware. Finally, the Mobile Net V3 combined both approaches, while also made changes such as adding the NetAdapt [90] algorithm in the achitectural search.

In those set of models, only the MNASNet obtained a sufficient Cohen Kappa score, as visible in the tables 4.12, 4.13 e 4.14. Specifically, the MNASNet with depth multiplier of 1.0, being that multiplier related to the number of channels dimension. It achieved that score combined with the Mix projection method method. According to the table 4.15 and the figure 4.5, all of those networks have very small memory size, with values inferior to 30 MB. They also have very fast inference time, with medians bellow to 20 ms. For those reasons and since it was the only net to achieve the minimum accuracy requirements, I chose the MNASNet 1.0 with Mix.

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
	Mix	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
	Mix	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
	Mix	**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**
	Mix	0.604 ± 0.249	0.884 ± 0.107	0.750 ± 0.312

Tabela 4.12: Averages and standard deviations of the folds evaluation for the MNASNet variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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**
	Mix	0.854 ± 0.249	**0.951 ± 0.086**	0.889 ± 0.296

Tabela 4.13: Averages and standard deviations of the folds evaluation for the MobileNet V2 variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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
	Mix	**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
	Mix	0.667 ± 0.246	0.822 ± 0.174	0.833 ± 0.207

Tabela 4.14: Averages and standard deviations of the folds evaluation for the MobileNet V3 variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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

Tabela 4.15: Memory size in Mega Bytes of each of the Mobile Oriented models variants.

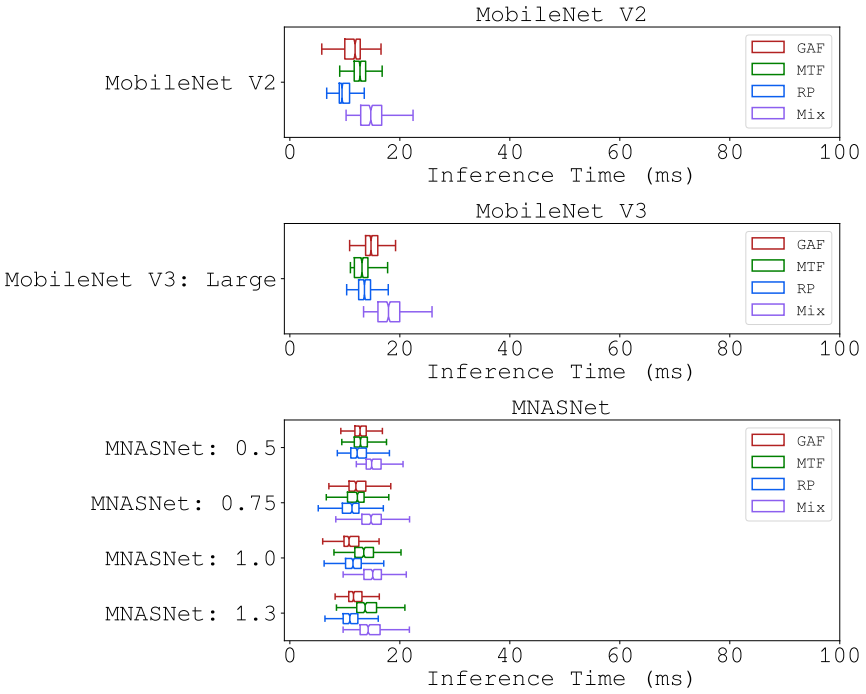

Figura 4.5: Inference time in miliseconds of each Mobile Oriented models variants.

Extreme Nets

In this section, I discuss the results of neural models which focus on one concept and push it to the limit, such as the VGG, the DenseNet and the SqueezeNet. The VGG is a model that uses filters of size 3×3 to allow adding more layers, increasing the depth model. On the other hand, the DenseNet applies skipping connections of the residual nets to all pairs of achitectural blocks present in the network, to achieve the benefits of having each layer closer to the input and the output of the model. Finally, the SqueezeNet tries to reach minimum memory usage not only by applying model compression techiniques but by introducing a new achitectural module which reduces the number of channels of a layer before another one with large convolutions filters, such as 3×3 filters. This reduces the number of parameters considerably.

Accordingly to the tables 4.16, 4.18 and 4.25, only the SqueezeNet and the VGG achieved the minimum Cohen Kappa score. For the Squeezenet, only the version 1.1, a more economic version if compared to the 1.0, reached the score, when combined to the RP and the Mix, with the Mix giving the best score. The second was the VGG with 16 weight layers without batch normalization, when combined to the RP projection method. Of those combinations, the SqueezeNet 1.1 with Mix performed better than the others, while having less variance. Adding to that performance, the SqueezeNet 1.1 is the smalest model in memory, according to the table 4.19. Furthermore, the SqueezeNets have low inference time if copared to the other models, as the Figure 4.6 depicts. Therefore, I selected for this section the SqueezeNet 1.1 with Mix.

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**
	Mix	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
	Mix	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
	Mix	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
	Mix	0.875 ± 0.199	**0.943** ± 0.086	0.924 ± 0.140

Tabela 4.16: Averages and standard deviations of the folds evaluation for the DenseNet variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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
	Mix	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**
	Mix	**0.917** ± 0.163	**0.955** ± 0.083	0.944 ± 0.130

Tabela 4.17: Averages and standard deviations of the folds evaluation for the SqueezeNet variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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
	Mix	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
	Mix	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
	Mix	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
	Mix	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**
	Mix	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
	Mix	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
	Mix	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
	Mix	0.708 ± 0.257	0.885 ± 0.123	0.833 ± 0.195

Tabela 4.18: Averages and standard deviations of the folds evaluation for the VGG variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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

Tabela 4.19: Memory size in Mega Bytes of each of the Extreme Nets models variants.

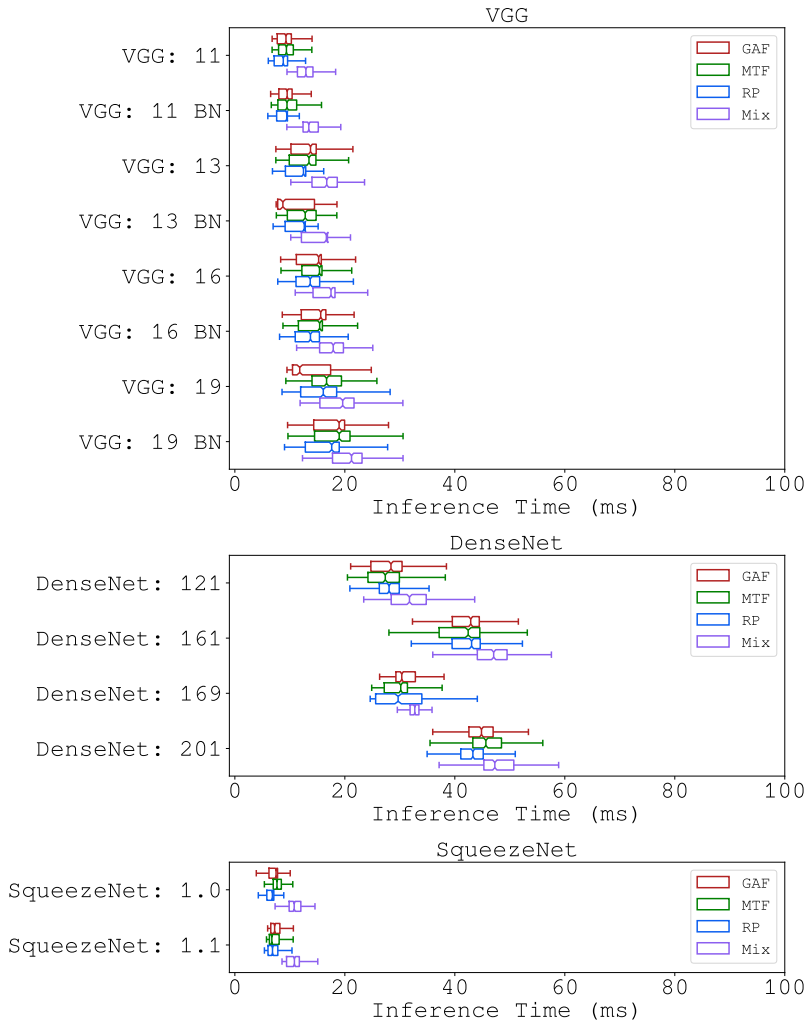

Figura 4.6: Inference time in miliseconds of each Extreme Nets models variants.

Efficiency Oriented

In this setion I discuss the models that researchers designed to efficiently utilize resources. For this work, I elaborate on the ShuffleNet V2, EfficientNet and EfficientNet V2. The first one is the ShuffleNet V2, sucessor of the ShuffleNet, which introduced the channel shuffle operator to allow information exchange among channels. It changes its predecessor by proposing, for each block, a channel split operation, to avoid costly operators such as grouped convolutions. Beside the shuffle nets, we have the efficient nets. The EfficientNet researchers studied the model scaling and created a compound resizing method that proportionally increase multiple dimensions, such as depth, number of channels and resolution. This allowed to create a very efficient base model that the researchers scaled up to bigger variants that mantain the advantages of the original. Finnaly, the EfficientNet V2 reformulated the antecessor by proposing a non-proportional scaling and by employing network architectural search. Additionally, it employed progressive learning by gradually increasing the dataset regularization.

We observe in the tables 4.22, 4.20, and 4.21 that none of the EfficientNet variants accomplished the minimum Cohen Kappa Score. Only a few variants of the other models remained. For the ShuffleNet V2, we see that the variants with $\times 1$ and $\times 0.5$ channels were the remainders, when combined with the Mix method. For the EfficientNet V2, the unique variant that I tested, we observe that only the Mix method allowed the minimum score. The scores that the ShuffleNet V2 variants reached are similar to the ones of the EfficientNet V2. However, the ShuffleNet V2 variants are smaller than the EfficientNet V2, as we can visualize in the Table 4.23. Furthermore, the Figure 4.7 shows that the ShuffleNet V2 variants are generally faster than the EfficientNet V2. For those reasons, I chose the smallest variant, ShuffleNet V2 $\times 0.5$, combined with the Mix projection method.

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
	Mix	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
	Mix	**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
	Mix	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
	Mix	0.604 ± 0.225	0.788 ± 0.181	0.792 ± 0.209

Tabela 4.20: Averages and standard deviations of the folds evaluation for the EfficientNet variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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
	Mix	**0.917** ± 0.163	**0.955** ± 0.083	**0.944** ± **0.130**

Tabela 4.21: Averages and standard deviations of the folds evaluation for the EfficientNetV2 variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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
	Mix	**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
	Mix	**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
	Mix	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
	Mix	0.729 ± 0.249	0.896 ± 0.106	0.847 ± 0.173

Tabela 4.22: Averages and standard deviations of the folds evaluation for the ShuffleNet V2 variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

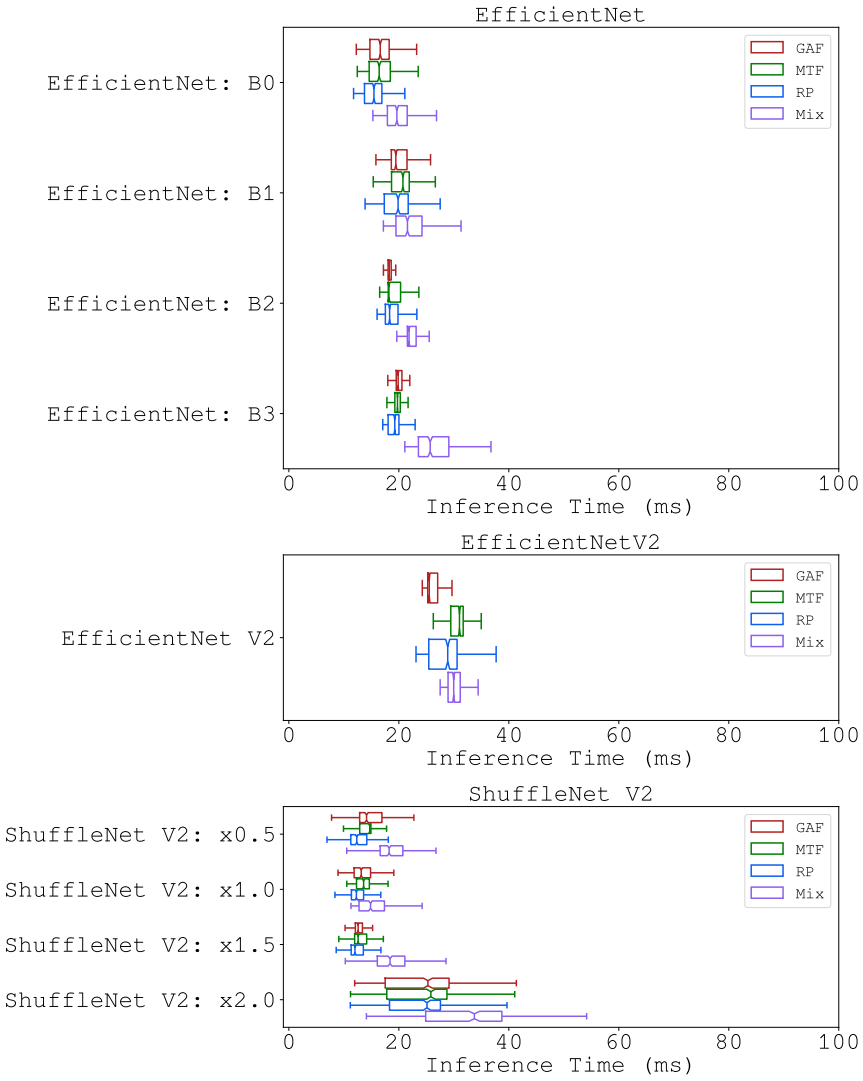

Figura 4.7: Inference time in miliseconds of each Efficiency Oriented models variants.

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		

Tabela 4.23: Memory size in Mega Bytes of each of the Efficiency Oriented models variants.

Diverse

In this section, I analyze the remaining models. They are: AlexNet, ConvNeXt and RegNet. Researchers introduced the first one in the old year of 2012. That net tested the idea of deep learning trained on multiple GPUs, allowing a faster training. Furthermore, it applyied dropout to reduce overfitting. In contrast, the ConvNeXt is a modern model of 2022, which reunites several convolutional techiniques of the past years, such as patchfied convolutions, inverted bottlenecks and grouped convolutions. The researchers proposed that model to push the state of traditional convolutional networks to the limit. Differently of the antecedents, the RegNet employs the ideia of designing network spaces rather than individual nets. Those populations are caracterized by the restriction of the parameter space to a linear function. This created populations of networks that are more apropriate for random achitectural search. In this manner, I am testing networks with big differences among them.

Amid those models, only the RegNet and the AlexNet passed the minimum Cohen Kappa test, as we can infer from the tables 4.25, 4.26, and 4.24. The AlexNet did so by applying the Mix method for its only variant. The RegNet, on the other hand, has a plenty of variants, that originate from two network design spaces: X, the base space, and Y, which adds a squeeze-and-excitation operation after the blocks of X. From those variants, only the 3.2 GF (Giga FLOPs) and 800 MF (Mega FLOPs) variants of the X network space and only the 400 MF and 800 MF variants of the Y network space reached the score. Most of them did so by applying the Mix, with only one exception: The RegNet Y 400 MF did by using the RP. Since the variants of both models scored similarly, we observe the Figure 4.8 and the Table 4.27. The first observation is that the the AlexNet occupies, at least, two times more memory than most of the RegNet variants, including the ones with good score. The second one is that the AlexNet has faster inference speed than every model in this section. But, since the memory occupation is priority, I chose the RegNet over the AlexNet. Specifically, the RegNet Y 400 MF variant (with RP), because it is the smallest.

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
	Mix	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
	Mix	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
	Mix	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
	Mix	**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
	Mix	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
	Mix	**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**
	Mix	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
	Mix	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
	Mix	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
	Mix	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
	Mix	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
	Mix	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
	Mix	**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
	Mix	0.771 ± 0.249	0.908 ± 0.109	0.868 ± 0.176

Tabela 4.24: Averages and standard deviations of the folds evaluation for the RegNet variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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
	Mix	**0.917** ± 0.163	**0.955** ± **0.083**	**0.944** ± **0.130**

Tabela 4.25: Averages and standard deviations of the folds evaluation for the AlexNet variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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**
	Mix	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**
	Mix	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
	Mix	**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
	Mix	0.875 ± 0.199	**0.951** ± **0.086**	0.939 ± 0.135

Tabela 4.26: Averages and standard deviations of the folds evaluation for the ConvNeXt variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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

Tabela 4.27: Memory size in Mega Bytes of each of the Diverse models variants.

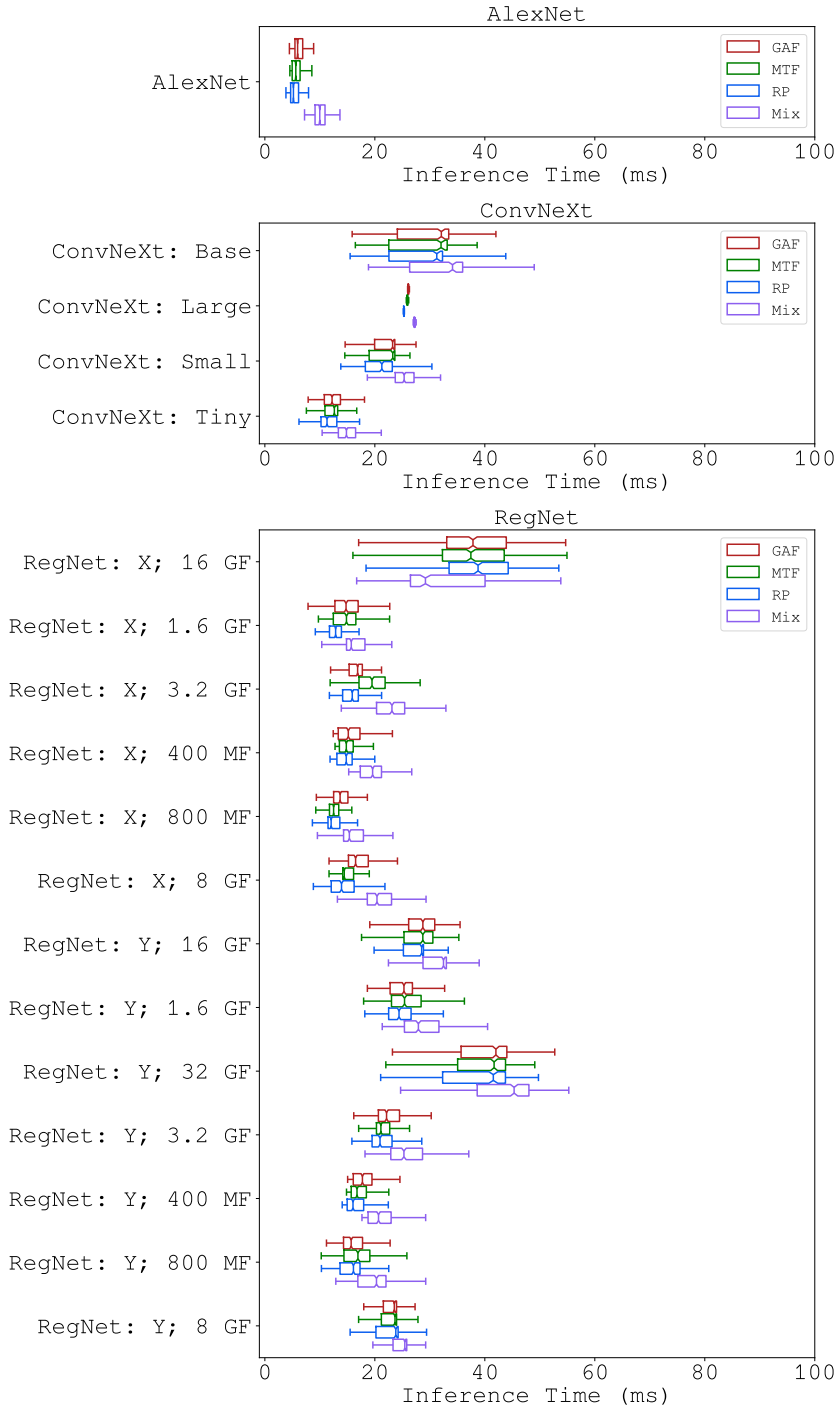

Figura 4.8: Inference time in miliseconds of each Diverse models variants.

4.2.2 Non-CV models comparison

This section evaluate non-computationally-visual models, which I presented in the chapter 2. Firstly, we observe in the Table 4.28 that seven models achieved the minimum Cohen Kappa score. Among them only two achieved the greatest scores: the Random Interval Spectral Ensemble Classifier and the Temporal Dictionary Ensemble. But they got similar scores, so I refer to the Table 4.29 and Figure 4.9 to decide which to choose. It is visible that the Random Interval Spectral Ensemble Classifier is faster in inference time than the Temporal Dictionary Ensemble, while the later is smaller in memory size. Due to the memory being a factor of major priority, I chose the Temporal Dictionary Ensemble.

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
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
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
MLP Classifier	0.485 ± **0.050**	0.821 ± 0.102	0.722 ± 0.130
MUSE	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
Ordinal TDE	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
RDST Classifier	0.633 ± 0.196	0.842 ± 0.125	0.819 ± 0.137
REDCOMETS	0.508 ± 0.095	0.817 ± 0.101	0.743 ± 0.153
Random Interval Classifier	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
Random Interval Spectral Ensemble Classifier	**0.938** ± 0.155	**0.971** ± 0.068	**0.972 ± 0.096**
Rocket Classifier	0.729 ± 0.225	0.859 ± 0.122	0.868 ± 0.176
Rotation Forest Classifier	0.854 ± 0.225	0.932 ± 0.111	0.910 ± 0.172
Shape DTW	0.729 ± 0.225	0.875 ± 0.106	0.868 ± 0.176
Shapelet Transform Classifier	0.625 ± 0.199	0.873 ± **0.067**	0.803 ± 0.131
Summary Classifier	0.883 ± 0.184	0.913 ± 0.154	0.944 ± 0.130
Supervised Time Series Forest	0.862 ± 0.184	0.896 ± 0.154	**0.972 ± 0.096**
TS Fresh Classifier	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
Temporal Dictionary Ensemble	**0.938** ± 0.155	**0.971** ± 0.068	**0.972 ± 0.096**
Time Series Forest Classifier	0.896 ± 0.167	0.938 ± 0.093	0.944 ± 0.130
WEASEL	0.875 ± 0.199	0.943 ± 0.086	0.924 ± 0.140
WEASEL V2	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130

Tabela 4.28: Averages and standard deviations of the folds evaluation for the Non CV variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

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

Tabela 4.29: Memory size in Mega Bytes of each of the Non CV models variants.

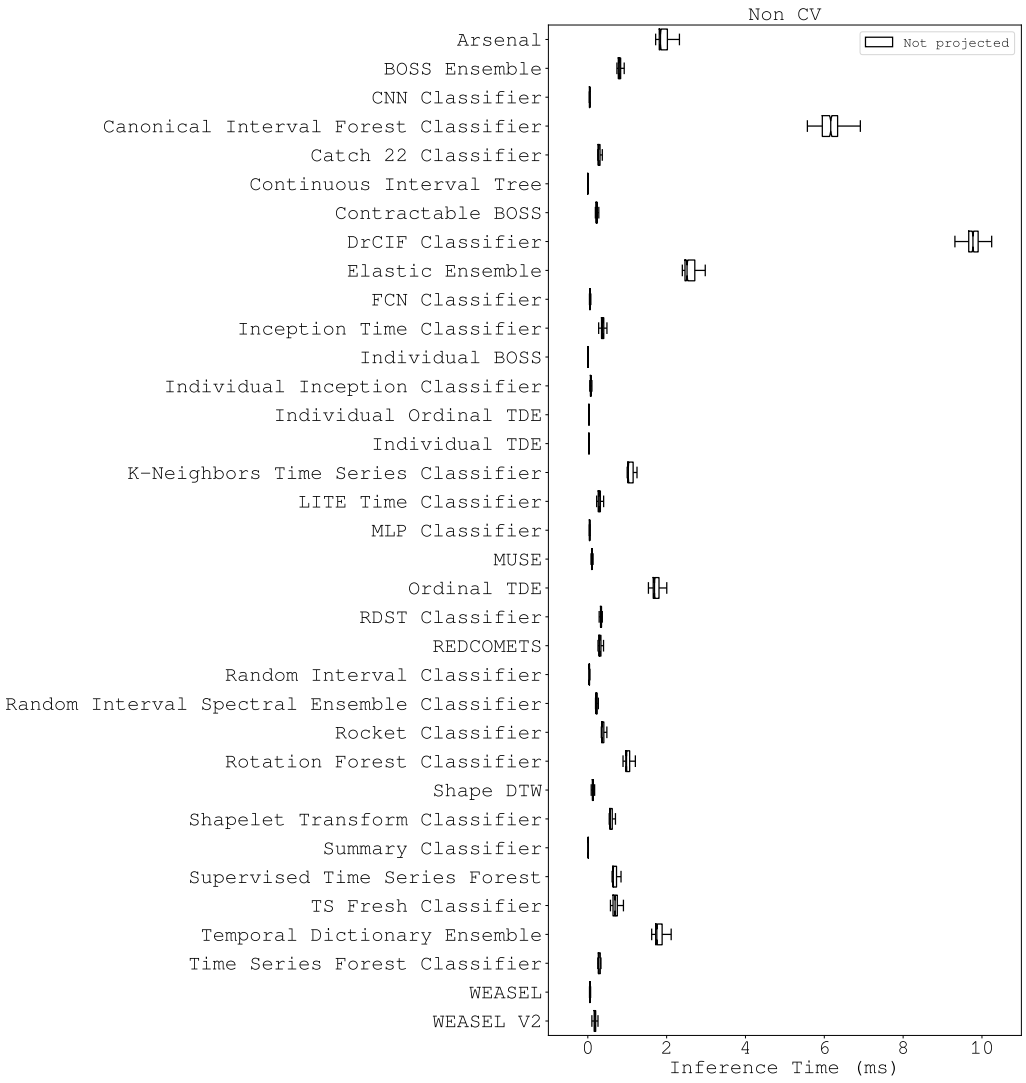

Figura 4.9: Inference time in miliseconds of each Non CV models variants.

4.2.3 Best models comparison

This section aggregates all choices that I made through all past sections. Observing their Table 4.30, we see that the two models obtained the best Cohen Kappa scores: the Temporal Dictionary Ensemble and the Wide ResNet 100-2 with Mix, even though the other models obtained comparable score. That Wide ResNet obtained the highest Cohen Kappa score, but there are some considerations. Firstly, that model is the largest of this set, as we can read on table 4.31. Secondly, it did not obtained the best F1 score and Precision. Lastly, the Wide ResNet is the second slowest model as we can refer to the Figure 4.10. On the other hand, the Temporal Dictionary Ensemble was the best in the usabiliy and security scores, F1-Score and Precision. It also is the fastest in inference and the smallest in memory. Therefore, I conclude that the CV approach lead to a better accuracy, represented by the Wide ResNet 100-2 with Mix, but the non-CV approach can be more useful, since it uses less resources and is less inconvenient to the user.

Model	Projection	Cohen Kappa	F1 Score	Precision
MNASNet: 1.0	Mix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
RegNet: Y; 400 MF	RP	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
ShuffleNet V2: x0.5	Mix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
SqueezeNet: 1.1	Mix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
Swin Transformer V2: S	Mix	0.917 ± 0.163	0.955 ± 0.083	0.944 ± 0.130
TemporalDictionaryEnsemble	Not projected	0.938 ± 0.155	**0.971 ± 0.068**	**0.972 ± 0.096**
Wide ResNet: 101-2	Mix	**0.955 ± 0.101**	0.967 ± 0.078	0.944 ± 0.130

Tabela 4.30: Averages and standard deviations of the folds evaluation for the Best Models variants. The colors represent the comparison of a cell to the others in the same column. The underline indicates that a value is greater than 90%.

Neural Network	Memory Size (MB)	Neural Network	Memory Size (MB)
TemporalDictionaryEnsemble	0.586264	RegNet: Y; 400 MF	15.617376
ShuffleNet V2: x0.5	1.376760	Swin Transformer V2: S	195.880352
SqueezeNet: 1.1	2.894136	Wide ResNet: 101-2	499.369720
MNASNet: 1.0	12.420792		

Tabela 4.31: Memory size in Mega Bytes of each of the Best Models models variants.

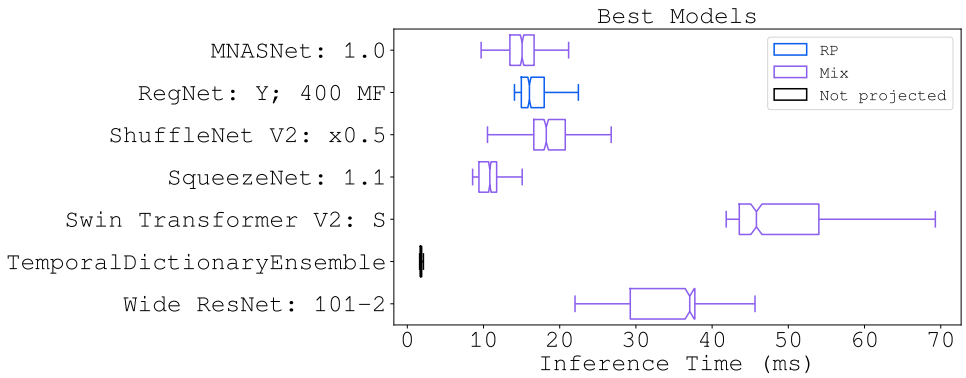

Figura 4.10: Inference time in miliseconds of each Best Models models variants.

4.3 Limitations

Even though the results are promising, we need to consider a series of limitations present in my experiment. One main observation is that I used 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, and signal lengths could lead to alterations on the obtained performance. Furthermore, the small size of the dataset resulted in a low ammount of testing, which makes the obtained results less trustable. A small dataset size also implies that the deep learning models had less resources to effectively learn. This contasts with the usual treatment for deep learning, where usually large ammounts of data feed the training of the model, allowing the proper adjustment of the large set of parameters. Therefore, my experiment would be more complete if I tested on different and larger datasets.

Other limit is that I did not explored all the options avaliable, in terms of models and hyperparameters. For instance, I did left out some of the models of the Pytorch and the Aeon libraries. Examples of them are the GoogLeNet [91], from the Pytorch, and the Hydra Classifier [92], from the Aeon. Additionally, I did not tested models external to those libraries, such as the Xception [93], avaliable at the Keras library [94]. Also, for some of the tested models, I did not evaluated all the variants, such as the Efficient Net B7. Furthermore, I did not search over the hypeparameters of the projection methods, as the number of dimensions of the RP. Thus, various options could be explored, as the different libraries that implement machine learning models, while the currenly used ones could be better availed. Moreover, I could do Optuna hyperparameter search over the projection hyperparameters.

Finally, some other limitations were evident at the implementation level. Firstly, I only

did data random oversampling for the purpose of balacing the dataset. However, there are other options designed specifically for time series, such as presented in [95]. With that, I could not only balance the dataset but enlarge it as well. Secondly, I used a resize transform to adapt the projection image to the model input. This could cause a considerable loss of information for matrix images that encode relationships in each pixel. 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 I proposed. There is a chance that this method prematurely stopped the training for models that required more epochs. Lastly, I did the measurements of the benchmarking metrics by using the Python standard API, which is not a direct method of measuring them, but is limited to the interpreter. Additionally, I 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, I could have done much improvements at the implementation level, such as applying time series augmentation techiniques, resizing without distorting the image by using an integer multiplier and padding the remainig space, researching early stopping methods, and measuring the benchmarking metrics in more controled environment and using low-level interfaces.

Capítulo 5

Conclusion

Referências

[1] Rice, S. O.: *Mathematical analysis of random noise.* The Bell System Technical Journal, 23(3):282–332, 1944. 2

[2] Shannon, C. E.: *A mathematical theory of communication.* The Bell System Technical Journal, 27(3):379–423, 1948. 2

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

[4] Wang, J.Y.: *A new method for evaluating ecg signal quality for multi-lead arrhythmia analysis.* Em *Computers in Cardiology*, páginas 85–88, 2002. 2

[5] Li, Q, R G Mark e 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`. 2, 3

[6] Deshmane, Anagha Vishwas: *False arrhythmia alarm suppression using ecg, abp, and photoplethysmogram.* Tese de Mestrado, Massachusetts Institute of Technology, 2009. 3

[7] Zhang, Pandeng, Jia Liu, Xinyu Wu, Xiaochang Liu e Qingchun Gao: *A novel feature extraction method for signal quality assessment of arterial blood pressure for monitoring cerebral autoregulation.* Em *2010 4th International Conference on Bioinformatics and Biomedical Engineering*, páginas 1–4, 2010. 3

[8] Kaplan Berkaya, Selcan, Alper Kursat Uysal, Efnan Sora Gunal, Semih Ergin, Serkan Gunal e M. Bilginer Gulmezoglu: *A survey on ecg analysis.* Biomedical Signal Processing and Control, 43:216–235, 2018, ISSN 1746-8094. `https://www.sciencedirect.com/science/article/pii/S1746809418300636`. 3

[9] Naseri, H. e 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. 3

[10] Orphanidou, Christina e Ivana Drobnjak: *Quality assessment of ambulatory ecg using wavelet entropy of the hrv signal.* IEEE Journal of Biomedical and Health Informatics, 21(5):1216–1223, 2017. 3

[11] Shahriari, Yalda, Richard Fidler, Michele M. Pelter, Yong Bai, Andrea Villaroman e Xiao Hu: *Electrocardiogram signal quality assessment based on structural image similarity metric*. IEEE Transactions on Biomedical Engineering, 65(4):745–753, 2018. 3

[12] Moeyersons, Jonathan, Elena Smets, John Morales, Amalia Villa, Walter De Raedt, Dries Testelmans, Bertien Buyse, Chris Van Hoof, Rik Willems, Sabine Van Huffel e Carolina Varon: *Artefact detection and quality assessment of ambulatory ecg signals*. Computer Methods and Programs in Biomedicine, 182:105050, 2019, ISSN 0169-2607. https://www.sciencedirect.com/science/article/pii/S0169260719312817. 3

[13] Huerta Álvaro, Arturo Martinez-Rodrigo, Vicente Bertomeu-González, Óscar Ayo-Martin, José J. Rieta e Raúl Alcaraz: *Single-lead electrocardiogram quality assessment in the context of paroxysmal atrial fibrillation through phase space plots*. Biomedical Signal Processing and Control, 91:105920, 2024, ISSN 1746-8094. https://www.sciencedirect.com/science/article/pii/S1746809423013538. 3

[14] Scardulla, Francesco, Gloria Cosoli, Susanna Spinsante, Angelica Poli, Grazia Iadarola, Riccardo Pernice, Alessandro Busacca, Salvatore Pasta, Lorenzo Scalise e Leonardo D'Acquisto: *Photoplethysmograhic sensors, potential and limitations: Is it time for regulation? a comprehensive review*. Measurement, 218:113150, 2023, ISSN 0263-2241. https://www.sciencedirect.com/science/article/pii/S0263224123007145. 3

[15] Li, Q e 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. 3, 4

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

[17] Pereira, Tania, Kais Gadhoumi, Mitchell Ma, Xiuyun Liu, Ran Xiao, Rene A. Colorado, Kevin J. Keenan, Karl Meisel e Xiao Hu: *A supervised approach to robust photoplethysmography quality assessment*. IEEE Journal of Biomedical and Health Informatics, 24(3):649–657, 2020. 4

[18] Pereira, Tania, Cheng Ding, Kais Gadhoumi, Nate Tran, Rene A Colorado, Karl Meisel e Xiao Hu: *Deep learning approaches for plethysmography signal quality assessment in the presence of atrial fibrillation*. Physiological Measurement, 40(12):125002, dec 2019. https://dx.doi.org/10.1088/1361-6579/ab5b84. 4

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

The 2nd International Conference on Emerging Data and Industry 4.0 (EDI40 2019) / Affiliated Workshops. 5

[20] Zanelli, Serena, Mounim A. El Yacoubi, Magid Hallab e Mehdi Ammi: *Transfer learning of cnn-based signal quality assessment from clinical to non-clinical ppg signals*. Em *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, páginas 902–905, 2021. 5

[21] Lucafó, Giovani Decico, Pedro Freitas, Rafael Lima, Gustavo da Luz, Ruan Bispo, Paula Rodrigues, Frank Cabello e Otavio Penatti: *Signal quality assessment of photoplethysmogram signals using hybrid rule-and learning-based models*. Journal of Health Informatics, 15(Especial), 2023. 5

[22] Gao, Haoyuan, Chao Zhang, Shengbing Pei e Xiaopei Wu: *Lstm-based real-time signal quality assessment for blood volume pulse analysis*. Biomed. Opt. Express, 14(3):1119–1136, Mar 2023. `https://opg.optica.org/boe/abstract.cfm?URI=boe-14-3-1119`. 5

[23] Chen, Jianzhong, Ke Sun, Yi Sun e Xinxin Li: *Signal quality assessment of ppg signals using stft time-frequency spectra and deep learning approaches*. Em *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, páginas 1153–1156, 2021. 5

[24] Chatterjee, Tamaghno, Aayushman Ghosh e Sayan Sarkar: *Signal quality assessment of photoplethysmogram signals using quantum pattern recognition technique and lightweight cnn module*. Em *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, páginas 3382–3386, 2022. 8

[25] Roh, Donggeun e Hangsik Shin: *Recurrence plot and machine learning for signal quality assessment of photoplethysmogram in mobile environment*. Sensors, 21(6), 2021, ISSN 1424-8220. `https://www.mdpi.com/1424-8220/21/6/2188`. 8

[26] Freitas, Pedro Garcia, Rafael G. De Lima, Giovani D. Lucafo e Otávio A.B. Penatti: *Photoplethysmogram signal quality assessment via 1d-to-2d projections and vision transformers*. Em *2023 15th International Conference on Quality of Multimedia Experience (QoMEX)*, páginas 165–170, 2023. 8

[27] Freitas, Pedro Garcia, Rafael G. De Lima, Giovani D. Lucafo e Otávio A. B. Penatti: *Assessing the quality of photoplethysmograms via gramian angular fields and vision transformer*. Em *2023 31st European Signal Processing Conference (EUSIPCO)*, páginas 1035–1039, 2023. 8

[28] Liu, Jian, Shuaicong Hu, Ya'nan Wang, Qihan Hu, Daomiao Wang e Cuiwei Yang: *A lightweight hybrid model using multiscale markov transition field for real-time quality assessment of photoplethysmography signals*. IEEE Journal of Biomedical and Health Informatics, 28(2):1078–1088, 2024. 8

[29] Nemcova, Andrea, Enikö Vargova, Radovan Smisek, Lucie Marsanova, Lukas Smital e Martin Vitek: *Brno university of technology smartphone ppg database (but ppg):*

Annotated dataset for ppg quality assessment and heart rate estimation. BioMed Research International, 2021. https://doi.org/10.1155/2021/3453007. 11

[30] Team, Aeon: *Aeon.* https://www.aeon-toolkit.org/en/stable/. Accessed: May 18, 2024. 13

[31] Foundation, PyTorch: *Pytorch documentation.* https://pytorch.org/docs/stable/index.html, 2023. Accessed: May 18, 2024. 13

[32] Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit e Neil Houlsby: *An image is worth 16x16 words: Transformers for image recognition at scale.* Em *International Conference on Learning Representations*, 2021. https://openreview.net/forum?id=YicbFdNTTy. 14

[33] Tu, Zhengzhong, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik e Yinxiao Li: *Maxvit: Multi-axis vision transformer.* Em Avidan, Shai, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella e Tal Hassner (editores): *Computer Vision – ECCV 2022*, páginas 459–479, Cham, 2022. Springer Nature Switzerland, ISBN 978-3-031-20053-3. 14

[34] Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin e Baining Guo: *Swin transformer: Hierarchical vision transformer using shifted windows.* Em *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, páginas 9992–10002, 2021. 14

[35] Liu, Ze, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei e Baining Guo: *Swin transformer v2: Scaling up capacity and resolution.* Em *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 11999–12009, 2022. 14

[36] He, Kaiming, Xiangyu Zhang, Shaoqing Ren e Jian Sun: *Deep residual learning for image recognition.* Em *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 770–778, 2016. 14

[37] Xie, Saining, Ross Girshick, Piotr Dollár, Zhuowen Tu e Kaiming He: *Aggregated residual transformations for deep neural networks.* Em *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 5987–5995, 2017. 14

[38] Zagoruyko, Sergey e Nikos Komodakis: *Wide residual networks.* Em Richard C. Wilson, Edwin R. Hancock e William A. P. Smith (editores): *Proceedings of the British Machine Vision Conference (BMVC)*, páginas 87.1–87.12. BMVA Press, September 2016, ISBN 1-901725-59-6. https://dx.doi.org/10.5244/C.30.87. 14

[39] Huang, Gao, Zhuang Liu, Laurens Van Der Maaten e Kilian Q. Weinberger: *Densely connected convolutional networks.* Em *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 2261–2269, 2017. 14

[40] Simonyan, Karen e Andrew Zisserman: *Very deep convolutional networks for large-scale image recognition*. Em Bengio, Yoshua e Yann LeCun (editores): *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. 14

[41] Iandola, Forrest N., Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally e Kurt Keutzer: *Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5MB model size*, 2017. https://openreview.net/forum?id=S1xh5sYgx. 14

[42] Tan, M., B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard e Q. V. Le: *Mnasnet: Platform-aware neural architecture search for mobile*. Em *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 2815–2823, Los Alamitos, CA, USA, jun 2019. IEEE Computer Society. https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00293. 14

[43] Sandler, M., A. Howard, M. Zhu, A. Zhmoginov e L. Chen: *Mobilenetv2: Inverted residuals and linear bottlenecks*. Em *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 4510–4520, Los Alamitos, CA, USA, jun 2018. IEEE Computer Society. https://doi.ieeecomputersociety.org/10.1109/CVPR.2018.00474. 14

[44] Howard, Andrew, Mark Sandler, Bo Chen, Weijun Wang, Liang Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, Yukun Zhu, Ruoming Pang, Hartwig Adam e Quoc Le: *Searching for mobilenetv3*. Em *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, páginas 1314–1324, 2019. 14

[45] Tan, Mingxing e Quoc V. Le: *Efficientnet: Rethinking model scaling for convolutional neural networks*. Em Chaudhuri, Kamalika e Ruslan Salakhutdinov (editores): *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 de *Proceedings of Machine Learning Research*, páginas 6105–6114. PMLR, 2019. http://proceedings.mlr.press/v97/tan19a.html. 14

[46] Tan, Mingxing e Quoc V. Le: *Efficientnetv2: Smaller models and faster training*. Em Meila, Marina e Tong Zhang (editores): *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 de *Proceedings of Machine Learning Research*, páginas 10096–10106. PMLR, 2021. http://proceedings.mlr.press/v139/tan21a.html. 14

[47] Ma, Ningning, Xiangyu Zhang, Hai Tao Zheng e Jian Sun: *Shufflenet v2: Practical guidelines for efficient cnn architecture design*. Em Ferrari, Vittorio, Martial Hebert, Cristian Sminchisescu e Yair Weiss (editores): *Computer Vision – ECCV 2018*, páginas 122–138, Cham, 2018. Springer International Publishing, ISBN 978-3-030-01264-9. 14

[48] Krizhevsky, Alex, Ilya Sutskever e Geoffrey E. Hinton: *Imagenet classification with deep convolutional neural networks*. Commun. ACM, 60(6):84–90, may 2017, ISSN 0001-0782. https://doi.org/10.1145/3065386. 14

[49] Liu, Zhuang, Hanzi Mao, Chao Yuan Wu, Christoph Feichtenhofer, Trevor Darrell e Saining Xie: *A convnet for the 2020s*. Em *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 11966–11976, 2022. 14

[50] Radosavovic, Ilija, Raj Prateek Kosaraju, Ross B. Girshick, Kaiming He e Piotr Dollár: *Designing network design spaces*. Em *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, páginas 10425–10433. Computer Vision Foundation / IEEE, 2020. https://openaccess.thecvf.com/content_CVPR_2020/html/Radosavovic_Designing_Network_Design_Spaces_CVPR_2020_paper.html. 14

[51] Middlehurst, Matthew, James Large, Michael Flynn, Jason Lines, Aaron Bostrom e Anthony Bagnall: *Hive-cote 2.0: a new meta ensemble for time series classification*. Mach. Learn., 110(11–12):3211–3243, dec 2021, ISSN 0885-6125. https://doi.org/10.1007/s10994-021-06057-9. 15

[52] Dempster, Angus, François Petitjean e Geoffrey I. Webb: *ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels*. Data Min. Knowl. Discov., 34(5):1454–1495, 2020. https://doi.org/10.1007/s10618-020-00701-z. 15

[53] Zhao, Bendong, Huanzhang Lu, Shangfeng Chen, Junliang Liu e Dongya Wu: *Convolutional neural networks for time series classification*. Journal of Systems Engineering and Electronics, 28(1):162–169, 2017. 15

[54] Wang, Zhiguang, Weizhong Yan e Tim Oates: *Time series classification from scratch with deep neural networks: A strong baseline*. Em *2017 International Joint Conference on Neural Networks, IJCNN 2017, Anchorage, AK, USA, May 14-19, 2017*, páginas 1578–1585. IEEE, 2017. https://doi.org/10.1109/IJCNN.2017.7966039. 15

[55] Ismail Fawaz, Hassan, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F. Schmidt, Jonathan Weber, Geoffrey I. Webb, Lhassane Idoumghar, Pierre Alain Muller e François Petitjean: *Inceptiontime: Finding alexnet for time series classification*. Data Min. Knowl. Discov., 34(6):1936–1962, nov 2020, ISSN 1384-5810. https://doi.org/10.1007/s10618-020-00710-y. 15

[56] Ismail-Fawaz, Ali, Maxime Devanne, Jonathan Weber e Germain Forestier: *Deep learning for time series classification using new hand-crafted convolution filters*. Em *2022 IEEE International Conference on Big Data (Big Data)*, páginas 972–981, 2022. 15

[57] Ismail-Fawaz, Ali, Maxime Devanne, Stefano Berretti, Jonathan Weber e Germain Forestier: *Lite: Light inception with boosting techniques for time series classification*. Em *2023 IEEE 10th International Conference on Data Science and Advanced Analytics (DSAA)*, páginas 1–10, 2023. 15

[58] Schäfer, Patrick: *The BOSS is concerned with time series classification in the presence of noise*. Data Min. Knowl. Discov., 29(6):1505–1530, 2015. https://doi.org/10.1007/s10618-014-0377-7. 15

[59] Middlehurst, Matthew, William Vickers e Anthony Bagnall: *Scalable dictionary classifiers for time series classification*. Em Yin, Hujun, David Camacho, Peter Tino, Antonio J. Tallón-Ballesteros, Ronaldo Menezes e Richard Allmendinger (editores): *Intelligent Data Engineering and Automated Learning – IDEAL 2019*, páginas 11–19, Cham, 2019. Springer International Publishing, ISBN 978-3-030-33607-3. 15

[60] Middlehurst, Matthew, James Large, Gavin Cawley e Anthony Bagnall: *The temporal dictionary ensemble (tde) classifier for time series classification*. Em *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part I*, página 660–676, Berlin, Heidelberg, 2020. Springer-Verlag, ISBN 978-3-030-67657-5. https://doi.org/10.1007/978-3-030-67658-2_38. 15

[61] Schäfer, Patrick e Ulf Leser: *Multivariate time series classification with WEASEL+MUSE*. CoRR, abs/1711.11343, 2017. http://arxiv.org/abs/1711.11343. 15

[62] Schäfer, Patrick e Ulf Leser: *Fast and accurate time series classification with weasel*. Em *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, página 637–646, New York, NY, USA, 2017. Association for Computing Machinery, ISBN 9781450349185. https://doi.org/10.1145/3132847.3132980. 15

[63] Schäfer, Patrick e Ulf Leser: *WEASEL 2.0: a random dilated dictionary transform for fast, accurate and memory constrained time series classification*. Mach. Learn., 112(12):4763–4788, 2023. https://doi.org/10.1007/s10994-023-06395-w. 15

[64] Bennett, Luca A. e Zahraa S. Abdallah: *RED comets: An ensemble classifier for symbolically represented multivariate time series*. Em Ifrim, Georgiana, Romain Tavenard, Anthony J. Bagnall, Patrick Schäfer, Simon Malinowski, Thomas Guyet e Vincent Lemaire (editores): *Advanced Analytics and Learning on Temporal Data - 8th ECML PKDD Workshop, AALTD 2023, Turin, Italy, September 18-22, 2023, Revised Selected Papers*, volume 14343 de *Lecture Notes in Computer Science*, páginas 76–91. Springer, 2023. https://doi.org/10.1007/978-3-031-49896-1_6. 15

[65] Abdallah, Zahraa S. e Mohamed Medhat Gaber: *Co-eye: a multi-resolution ensemble classifier for symbolically approximated time series*. Mach. Learn., 109(11):2029–2061, 2020. https://doi.org/10.1007/s10994-020-05887-3. 15

[66] Lines, Jason e Anthony J. Bagnall: *Time series classification with ensembles of elastic distance measures*. Data Min. Knowl. Discov., 29(3):565–592, 2015. https://doi.org/10.1007/s10618-014-0361-2. 15

[67] Zhao, Jiaping e Laurent Itti: *shapedtw: Shape dynamic time warping*. Pattern Recognition, 74:171–184, 2018, ISSN 0031-3203. https://www.sciencedirect.com/science/article/pii/S0031320317303710. 15

[68] Lubba, Carl Henning, Sarab S. Sethi, Philip Knaute, Simon R. Schultz, Ben D. Fulcher e Nick S. Jones: *catch22: Canonical time-series characteristics - selected through highly comparative time-series analysis*. Data Min. Knowl. Discov., 33(6):1821–1852, 2019. https://doi.org/10.1007/s10618-019-00647-x. 15

[69] Christ, Maximilian, Nils Braun, Julius Neuffer e Andreas W. Kempa-Liehr: *Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package)*. Neurocomputing, 307:72–77, 2018, ISSN 0925-2312. https://www.sciencedirect.com/science/article/pii/S0925231218304843. 15

[70] Bagnall, Anthony, Michael Flynn, James Large, Jason Lines e Matthew Middlehurst: *On the usage and performance of the hierarchical vote collective of transformation-based ensembles version 1.0 (hive-cote v1.0)*. Em *Advanced Analytics and Learning on Temporal Data: 5th ECML PKDD Workshop, AALTD 2020, Ghent, Belgium, September 18, 2020, Revised Selected Papers*, página 3–18, Berlin, Heidelberg, 2020. Springer-Verlag, ISBN 978-3-030-65741-3. https://doi.org/10.1007/978-3-030-65742-0_1. 15

[71] Middlehurst, Matthew, James Large e Anthony Bagnall: *The canonical interval forest (cif) classifier for time series classification*. Em *2020 IEEE International Conference on Big Data (Big Data)*, páginas 188–195, 2020. 15

[72] Lines, Jason, Sarah Taylor e Anthony Bagnall: *Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles*. ACM Trans. Knowl. Discov. Data, 12(5), jul 2018, ISSN 1556-4681. https://doi.org/10.1145/3182382. 15

[73] Cabello, Nestor, Elham Naghizade, Jianzhong Qi e Lars Kulik: *Fast and accurate time series classification through supervised interval search*. Em *2020 IEEE International Conference on Data Mining (ICDM)*, páginas 948–953, 2020. 15

[74] Deng, Houtao, George Runger, Eugene Tuv e Martyanov Vladimir: *A time series forest for classification and feature extraction*. Information Sciences, 239:142–153, 2013, ISSN 0020-0255. https://www.sciencedirect.com/science/article/pii/S0020025513001473. 15

[75] Hills, Jon, Jason Lines, Edgaras Baranauskas, James Mapp e Anthony J. Bagnall: *Classification of time series by shapelet transformation*. Data Min. Knowl. Discov., 28(4):851–881, 2014. https://doi.org/10.1007/s10618-013-0322-1. 15

[76] Bostrom, Aaron e Anthony J. Bagnall: *Binary shapelet transform for multiclass time series classification*. Trans. Large Scale Data Knowl. Centered Syst., 32:24–46, 2017. https://doi.org/10.1007/978-3-662-55608-5_2. 15

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

in Computer Science, páginas 653–664. Springer, 2022. https://doi.org/10.1007/978-3-031-09037-0_53. 15

[78] Guillaume, Antoine, Christel Vrain e Wael Elloumi: *Time series classification with Shapelets: Application to predictive maintenance on event logs. (Classification de séries temporelles avec les Shapelets : application à la maintenance prédictive via journaux d'événements)*. Tese de Doutoramento, University of Orléans, France, 2023. https://tel.archives-ouvertes.fr/tel-04368849. 15

[79] Ayllón-Gavilán, Rafael, David Guijo-Rubio, Pedro Antonio Gutiérrez e César Hervás-Martínez: *A dictionary-based approach to time series ordinal classification*. Em *Advances in Computational Intelligence: 17th International Work-Conference on Artificial Neural Networks, IWANN 2023, Ponta Delgada, Portugal, June 19–21, 2023, Proceedings, Part II*, página 541–552, Berlin, Heidelberg, 2023. Springer-Verlag, ISBN 978-3-031-43077-0. https://doi.org/10.1007/978-3-031-43078-7_44. 15

[80] Deng, Houtao, George Runger, Eugene Tuv e Martyanov Vladimir: *A time series forest for classification and feature extraction*. Information Sciences, 239:142–153, 2013, ISSN 0020-0255. https://www.sciencedirect.com/science/article/pii/S0020025513001473. 15

[81] Rodríguez, Juan José, Ludmila I. Kuncheva e Carlos J. Alonso: *Rotation forest: A new classifier ensemble method*. IEEE Trans. Pattern Anal. Mach. Intell., 28(10):1619–1630, 2006. https://doi.org/10.1109/TPAMI.2006.211. 15

[82] Contributors, Optuna: *Optuna: A hyperparameter optimization framework*. https://optuna.readthedocs.io/en/stable/, 2018. Accessed: May 18, 2024. 13

[83] Stanford Vision Lab, Stanford University, Princeton University: *Imagenet*. https://www.image-net.org/, 2020. Accessed: May 18, 2024. 14

[84] Kingma, Diederik P. e Jimmy Ba: *Adam: A method for stochastic optimization*. Em Bengio, Yoshua e Yann LeCun (editores): *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. 14

[85] developers, The imbalanced-learn: *imbalanced-learn documentation*. https://imbalanced-learn.org/stable/, 2024. Accessed: May 18, 2024. 19

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

[87] Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai e Soumith Chintala: *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates Inc., Red Hook, NY, USA, 2019. 19

[88] developers scikit-learn: *scikit-learn*. https://scikit-learn.org/, 2024. Accessed: May 18, 2024. 19

[89] Jean Brouwers, Ludwig Haehne, Robert Schuppenies: *Pympler*. https://pympler.readthedocs.io/en/latest/, 2020. Accessed: May 18, 2024. 19

[90] Yang, Tien-Ju, Andrew G. Howard, Bo Chen, Xiao Zhang, Alec Go, Mark Sandler, Vivienne Sze e Hartwig Adam: *Netadapt: Platform-aware neural network adaptation for mobile applications*. Em Ferrari, Vittorio, Martial Hebert, Cristian Sminchisescu e Yair Weiss (editores): *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part X*, volume 11214 de *Lecture Notes in Computer Science*, páginas 289–304. Springer, 2018. https://doi.org/10.1007/978-3-030-01249-6_18. 27

[91] Szegedy, C., Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke e A. Rabinovich: *Going deeper with convolutions*. Em *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 1–9, Los Alamitos, CA, USA, jun 2015. IEEE Computer Society. https://doi.ieeecomputersociety.org/10.1109/CVPR.2015.7298594. 43

[92] Dempster, Angus, Daniel F. Schmidt e Geoffrey I. Webb: *Hydra: competing convolutional kernels for fast and accurate time series classification*. Data Min. Knowl. Discov., 37(5):1779–1805, may 2023, ISSN 1384-5810. https://doi.org/10.1007/s10618-023-00939-3. 43

[93] Chollet, F.: *Xception: Deep learning with depthwise separable convolutions*. Em *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 1800–1807, Los Alamitos, CA, USA, jul 2017. IEEE Computer Society. https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.195. 43

[94] Inc, Google: *Keras*. https://keras.io/. Accessed: May 18, 2024. 43

[95] Wen, Qingsong, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang e Huan Xu: *Time series data augmentation for deep learning: A survey*. Em Zhou, Zhi-Hua (editor): *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, páginas 4653–4660. ijcai.org, 2021. https://doi.org/10.24963/ijcai.2021/631. 44