Computers in Biology and Medicine 199 (2025) 111322

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/compbiomed

# Explainable prototype-based image classification using adaptive feature extractors in medical images

Nicolas Vasconcellos [a,b], Luis M.N. Tavora [a,b], Rolando Miragaia [b,c], Carlos Grilo [b,c], Lucas A. Thomaz [a,b,*]

[a] Instituto de Telecomunicações, Leiria, 2411-901, Portugal
[b] ESTG, Polytechnic of Leiria, Leiria, 2411-901, Portugal
[c] Computer Science and Communications Research Centre, Polytechnic of Leiria, Leiria, 2411-901, Portugal

ARTICLE INFO

*MSC:*
68T07
92C50
92C55

*Keywords:*
Explainable artificial intelligence
Explainable classification
Prototype-based classifiers
Clustering
Adaptive feature extractors
Medical imaging
Medical imaging classification

ABSTRACT

Prototype-based classifiers are a category of Explainable Artificial Intelligence methods that use representative samples from the data, called prototypes, to classify new inputs based on a similarity criterion. However, these methods often rely on pre-trained Convolutional Neural Networks as feature extractors, which may not be adapted for the specific type of data being used, thus not suited for identifying the most representative prototypes. In this paper, we propose a method named Explainable Prototype-based Image Classification, a cluster-oriented training strategy that enhances the performance and explainability of prototype-based classifiers. Our method uses a novel loss function, called Cluster Density Error, to fine-tune the feature extractor and preserve the most representative feature vectors in the latent space. We also use Principal Component Analysis-based approach to reduce the dimensionality and complexity of the feature vectors. We conduct experiments on four medical image datasets and compare the results with those from different prototype-based classifiers and state-of-the-art non-explainable learning methods. The proposed method demonstrated superior explainable capabilities and comparable classification performance to the compared methods. Specifically, the proposed method achieved up to 95.01% accuracy and 0.992 AUC using only 43 prototypes. This translated to an improvement in accuracy and AUC score of 21.54% and 9.06%, respectively, and a substantial reduction in the number of prototypes by 98,38%.

## 1. Introduction

Deep neural networks are Machine Learning (ML) models that have shown great effectiveness in performing tasks where pattern recognition plays an important role [1]. That is the case in medicine, for example in areas like haematology [2,3], dermatology [4], oncology [5], ophthalmology [6], radiography [7], and neurology [8]. In some specific cases, it was reported that their performance is rising towards the humans level [9]. These algorithms have shown high performance particularly in tasks such as brain tumour segmentation [10], body organ recognition in medical images [11], magnetic resonance image reconstruction [12,13], interstitial lung disease classification [14], lung nodule detection [15], and cancer detection in different body regions, such as the lungs, breasts, and kidneys [16,17], among others.

However, these models are often seen as black-boxes due to the difficulty in interpreting and tracking the outputs produced with each set of input data. The opacity of such ML models can result in the unintentional learning of patterns within the training data, which may lead to undesirable behaviour. In the domain of medical diagnosis, for instance, ML algorithms have exhibited unintended biases based on factors such as gender [18], ethnicity [19], and socioeconomic status [20]. These biases can have far-reaching consequences and pose ethical and practical challenges.

Machine learning algorithms are adept at leveraging any available signals to optimise their performance within the specific dataset they are trained on. In doing so, they might even exploit unknown confounding factors that, while effective within the training dataset, may not be reliable. This can ultimately hinder the algorithm's ability to generalise effectively to new and unseen datasets [21]. Several studies have shed light on the existence of systematic differences between images belonging to different classes. Intriguingly, these differences may perfectly correlate with the presence of a specific disease, even though they are unrelated to whether an individual has the disease or not. This phenomenon has been observed in various studies [22–26].

* Corresponding author at: ESTG, Polytechnic of Leiria, Leiria, 2411-901, Portugal.
*E-mail addresses:* nicolas.vasconcellos@ieee.org (N. Vasconcellos), luis.tavora@co.it.pt (L.M.N. Tavora), rolando.miragaia@ipleiria.pt (R. Miragaia), carlos.grilo@ipleiria.pt (C. Grilo), lucas.thomaz@co.it.pt (L.A. Thomaz).

https://doi.org/10.1016/j.compbiomed.2025.111322
Received 14 May 2024; Received in revised form 10 September 2025; Accepted 14 November 2025
Available online 21 November 2025

In a notable study published in DeGrave et al. [27], the issue of systematic inter-class disparities in datasets is addressed, particularly in the context of detecting COVID-19 infection from chest X-ray images. The researchers shed light on how deep learning classifiers tend to exploit these systematic differences as "shortcuts" to make classification decisions. This problem can be attributed to the composition process of training datasets, which often draws images exclusively from positive and negative sources, inadvertently creating artificial disparities between the classes.

In this context, several questions may arise: "Can we trust the results of the algorithm?", "To what extent can we make decisions based on these results?" [28]. This is the background to the recent emergence of Explainable Artificial Intelligence (XAI), whose aim is to develop a framework of models and tools that make it possible to analyse the relevance of each of the factors algorithmically involved in the definition of a given output.

Prototype-based classifiers are a well-known category of XAI algorithms, falling within the subset of Intrinsically Interpretable algorithms. The hallmark of these algorithms is that their internal processes are inherently interpretable by humans without the need for any adaptation. Prototype-based classifiers belong to this category because of the way they make decisions, which involves using representative samples from the data, referred to as prototypes, to assign labels to new inputs based on a similarity criterion. The underlying idea is that if the prototypes are interpretable, the decisions based on them will also be interpretable [29].

Prototype-based classifiers are often constructed using a pre-trained *CNN as a feature extractor*. *This CNN* project the inputs into an embedding space on which the clustering process is based. However, a primary challenge with this approach is that the used networks were initially trained for a classification task, rather than identifying the most representative samples within the dataset, as required by prototype-based classifiers. Classification-oriented training aims to separate inputs of one class within the embedding space from inputs of other classes and concentrate them around a single point, known as a global maximum, to facilitate the classification process. This is not well-suited for prototype-based classifiers, which need various local maxima to obtain a set of representative samples for explaining their results. When all samples are concentrated around a single global maximum, the identified prototypes may fail to be representative of the full diversity of the dataset. Consequently, the explanations generated by the classifier may not be sufficiently understandable from a human perspective.

In order to address this issue, we propose the ExPIC method, a cluster-based training strategy designed to enhance the performance of *CNNs* as feature extractors. This approach enables prototype-based classifiers to identify more representative samples (whole images from the dataset), enhancing their explainability while maintaining competitive accuracy levels. The proposed method also offers the flexibility to define the allowed similarity levels between prototypes of different classes and provides optimisation possibilities for its implementation in scenarios where time and memory complexities are critical. The code for the proposed method is available on GitHub at the following link: https://github.com/nicoVascon/ExPIC.

The remainder of the document is structured as follows: Section 2 delves deeper into the challenges associated with XAI and explores the main approaches used to explain ML algorithms. Section 2.3 details the primary prototype-based classifiers studied and employed in this work: the *xDNN* methods. In Section 3, the different components of the proposed method including the optimisation possibilities mentioned above are presented. The results of the conducted experiments that assess the efficiency of the proposed method are presented in Section 4. Additionally, Section 4.4 presents an ablation study to show the individual contribution of each component of the proposed method in the final results. Finally, in Section 5 the conclusions of this work are presented.

## 2. Related work

ML algorithms can be divided into interpretable and non-interpretable. The main difference between the two categories is the ability to inspect the internal decision-making process of each algorithm. Their intrinsic characteristics may or may not coincide with those of the processes that humans use to take their decisions [30]. This often makes it impossible to understand and trust the predictions of these models.

Even with the dramatic advances in machine learning techniques, straightforward models such as linear regressions and decision trees continue to be favoured in fields like marketing, healthcare analytics, and scientific research—domains where interpreting the underlying data patterns matters as much or more to users than just prediction accuracy alone [31]. However, as discussed in Nazir et al. [32], higher explainability levels are often obtained at the cost of model Accuracy, i.e., overall performance. This behaviour is due to the fact that solving more complex problems requires more complex systems [33], whose process of arriving at conclusions is not easily understandable by human beings. These models possess greater flexibility than their simple counterparts, enabling them to capture more complex relationships. These models possess greater flexibility than their simple counterparts, enabling them to capture more complex relationships [34].

To achieve the goal of explaining the processes by which ML models carry out their decision-making, there are two main approaches. One, to develop methods that allow the identification of features learned by models already trained and deployed. These methods are coined *Post-hoc* and can be applied to a wide variety of models depending on their nature. The other is to develop algorithms whose processes are self-explanatory to humans, called *Intrinsically Interpretable* methods [35]; examples are Classification Rules, K-Nearest Neighbours, and Decision Trees. Such models, while being easier to interpret, can be somewhat limited in their performance. When working with sophisticated models such as Support Vector Machines and CNN, researchers can employ post-hoc XAI techniques to generate simpler interpretive models that approximate the behaviour of the more complex systems [36].

### 2.1. Post-hoc methods

*Post-hoc* methods can be grouped into two main categories based on their relationship with the model to be explained: *Model-Specific* or *Model-Agnostic*. *Model-Specific* methods properly explore the intrinsic characteristics of the model to create an explanation that reflects the meaning of the internal processes of the *black-box* algorithm to be explained. On the other hand, *Model-Agnostic* methods, although less "specific", have the advantage of being applicable to any kind of model, regardless of its architecture or complexity.

From the above, it can be understood that adopting one approach or the other involves some sort of compromise: *Model-Agnostic* methods are more versatile and suitable for various problems, but that comes with the limitation that they may not capture the model's behaviour in detail. *Model-Specific* approaches are tailored to a certain model and are therefore suited to exploit its features; hence, they have the advantage of being more accurate and reliable for the model they are designed for, but they have the limitation of being incompatible with other models and may need to be adjusted if the model changes [37,38].

The application of *Model-Specific* methods is restricted to only a *specific* family of algorithms [39], and an example are *CNNs*. In these, their own gradients can be used to determine the importance of each image pixel to create an *Attention Map* that reflects where the neural network focused. Such algorithms are generally coined *Gradient-Based*, and examples are: *Vanilla Gradients*[1] that use raw gradients to create attention maps [40,41]; *Integrated Gradients*, which accumulate the

[1] In the context of algorithms, AI/ML in particular, the term "vanilla" is used for models in their simplest, unmodified or standard form ("raw").

vanilla gradients computed at all the points along the straightline path from a baseline input to the input of interest [42]; and the *SmoothGrad* methods which improve the attention maps' coherency by integrating the sensitivity maps computed from noised samples of the original input [43].

Concerning the *Model-Agnostic* family of methods, one of the most popular is the *Local Interpretable Model-Agnostic Explanations (LIME)* [28]. This method treats the algorithm as a black box and aims to explain its behaviour by identifying the most relevant features of the input that lead the model to return that particular output. This is done by analysing variations in the model's output when exposed to perturbed versions of the same input.

### 2.2. Intrinsically intepretable methods

*Intrinsically Interpretable* methods are the opposite of *black-box*-like approaches, and are, therefore, usually referred by as *Transparent* Artificial Intelligence (AI) algorithms [44]. The main characteristic of these methods is that they are interpretable by design, i.e. they do not require additional processes to explain their results because their own processes are already interpretable by humans.

Examples of *Intrinsically Intepretable* methods are: *Linear Regression* models, which use a set of independent input variables to predict a dependent one by fitting a straight line to the observed data [45–48]; *Decision Trees*, which involve partitioning the data along the predictor axes into subsets with uniform values of the dependent variable and then using this process to make predictions from new observations [49,50]; *Classification Rules*, that make use of *IF-THEN* rules for class prediction [51–53].

#### 2.2.1. Prototype-based classifiers

Prototype-based classifiers constitute a category of machine learning methods that offer an inherently interpretable approach to classification. These methods operate on the principle of comparing new, unseen data with a set of learned prototypical examples that represent characteristic patterns in the training data [54]. The fundamental premise underlying this approach is that these prototypes serve as exemplars or typical instances of different classes within the dataset. Classification is performed by quantifying the similarity between the input data and these prototypes, typically employing a nearest neighbour rule where the input is assigned to the class of the most similar prototype [55].

The inherent interpretability of prototype-based classifiers makes them particularly valuable in high-stakes domains such as medical diagnosis [56,57]. In these applications, understanding the rationale behind a diagnostic prediction is crucial for building trust among healthcare professionals and ensuring the adoption of these technologies in clinical practice [58]. Prototype-based classifiers enhance transparency by providing explanations based on similarity to learned medical examples, an advantage often missing in more complex "black box" models [57].

Such explanations have been applied in various medical tasks, including classifying mass lesions in mammogram images [59–61], diagnosing pneumonia from chest X-ray images [62], whole-slide classification of breast cancer [63], and identifying COVID-19 cases [64]. This transparency helps detect potential biases [65], identify failure cases [66], and ultimately allows medical experts to validate the model's decision-making process [67].

Within the domain of prototype-based classification, two primary categories emerge: whole-image-based and patch-based classifiers.

***Whole-image-based prototype classifiers:***. These algorithms are conceived to learn characteristic features of each class in the whole-images. The core principle involves identifying both local and global features that are common among the members of each class and selecting typical images that exhibit these features [68,69].

Prominent examples include Prototypical Networks [70] designed for few-shot classification problems. These networks learn a metric space in which classification can be performed by computing distances to prototype representations of each class. Another notable example is presented by Li et al. [71], who employ an autoencoder to project images into a latent space where comparisons between input images and prototypes occur. This model autonomously learns an optimal set of prototypes to achieve accurate predictions, with the decoder subsequently used to visualise these prototypes.

***Patch-based prototype classifiers:***. In contrast, patch-based prototype classifiers utilise prototypes that represent small, localised regions or "patches" extracted from images. The underlying premise is that different image classes may be characterised by the presence or absence of specific prototypical parts [72]. These classifiers learn a set of such prototypical parts, with each prototype ideally capturing a distinct visual feature that is relevant for discriminating between different classes [54]. The classification of a new image is determined by identifying which class contains the largest number of prototypical patches within the image [72].

A seminal example of patch-based prototype classifiers is ProtoPNet proposed by Chen et al. [72]. This network emulates human reasoning when confronted with challenging image classification tasks by dissecting the image and identifying prototypical aspects of different classes before making a decision. The architecture is designed to identify representative parts of an image for a particular class and use them to classify new images. Due to its patch-based reasoning, the network can highlight the most relevant parts of the input image that contributed to the classification result. This method has inspired several derivatives, including: XProtoNet, proposed by Kim et al. [73], which provides local and global explanations by modifying ProtoPNet to allow dynamic-sized prototypes. This approach better aligns with the nature of disease indicators in medical images; BRAIxProtoPNet++, proposed by Wang et al. [74], which applies knowledge distillation from a non-interpretable model into ProtoPNet with modifications to increase prototype diversity; and PIP-Net, proposed by Nauta et al. [75], which builds upon ProtoPNet's principles but introduces a novel regularisation for learning prototype similarity that better correlates with human visual perception. Unlike other methods requiring a fixed number of prototypes, PIP-Net only requires a maximum number and selectively employs as few prototypes as possible to achieve strong classification accuracy with compact explanations.

### 2.3. Explainable deep neural networks (xDNN) model

An example of *Intrinsically Intepretable* methods is described in Angelov and Soares [76], where the authors propose the *xDNN* model. It is a prototype-based classifier that, as demonstrated in Li et al. [77], exploits the ability of *CNNs* to extract features from the input images and create a latent hyperspace over which the network represents, or projects, the input images. This latent space enables a clustering process where the algorithm identifies samples that are most representative of the training set for each class of interest. In this context, a sample is represented by a whole-image of the dataset. These representative samples serve as the basis for creating IF-THEN rules. The decision-making process of the *xDNN* algorithm relies on these rules, which are "designed" to be inherently understandable to humans.

The architecture of the *xDNN* classification method is illustrated in Fig. 1, where a CNN is employed to extract a set of features $\Phi$. These features are then passed through a Fully Connected network to obtain the projection of the image $X$, denoted as $\varphi$, in the latent space. The projection $\varphi$ serves as input for the prototype-based classifier *Evolving ADP* described in Angelov and Gu [78], which provides, for each image, the predicted class along with the corresponding prototype

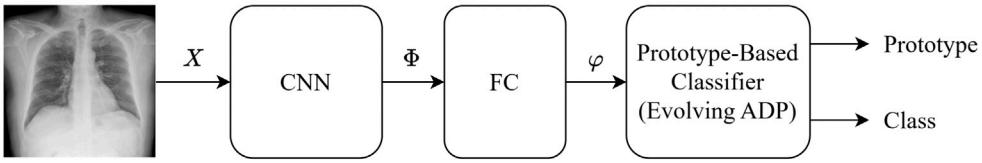



**Fig. 1.** Generic *xDNN* classification method architecture. *CNN*: Convolutional Neural Network used as a feature extractor; *FC*: Fully Connected Network used to combine the extracted feature; Prototype-based classifier: *Evolving ADP*.

as a visual explanation of the classification. On top of this model, a set of enhancements were developed in this work, which led to a new processing framework, described in Section 3.

The same authors of Angelov and Soares [76] introduced an alternative approach for the *Evolving ADP* as outlined in Angelov and Gu [78]. This alternative method is referred to as the *Offline ADP*. In the next sections, an overview of both methods is provided. For the sake of simplicity, in the rest of this document the *xDNN* method is referred to as the "*xDNN Evolving*" and its alternative version as the "*xDNN Offline*".

### 2.3.1. xDNN evolving method

The architectural layout of the *xDNN Evolving* classifier is presented in this section. This classifier, as detailed in the research conducted by Angelov and Soares [76], comprises four main blocks: the *Feature Extraction* block, the *Density* block, the *Prototypes* block, and the *MegaClouds* block.

The *Feature Extraction* block plays a central role as it encompasses convolutional layers responsible for extracting essential features and projecting the input data onto a latent space. The *xDNN Evolving* classifier groups the images according to their similarity level. In order to determine it, the *Data Density* is calculated (*Density* block), which depends on the relative distance, in latent space, between the images in the training set. The *Data Density* is obtained through Eq. (1), where $x_i$ represents the feature vector of an image in latent space, $\mu_N$ is the average feature vector of all images in the training set, and $\sigma$ is the standard deviation of the *Feature Vectors*' components of the images with respect to their mean.

$$D(x_i) = \frac{1}{1 + \frac{\|x_i - \mu_N\|^2}{\sigma_N^2}} \tag{1}$$

The *Prototype* block is the core of the algorithm, being in charge of determining which input samples (images) from the training dataset, will be considered as prototypes. The images with peak values of *Data Density* are considered as new prototypes around which a new cluster, also referred to in Angelov and Soares [76] as a *Data Cloud*, will be created. A cluster (or *Data Cloud*) corresponds to the set of inputs associated to the same prototype. Thus, each class has its own set of prototypes, which can be used to create *IF-THEN* rules to classify the images within a given class.

In the training process, the training images are analysed to identify the initial prototypes and create the clusters. Each cluster has an area of influence within which the new points that appear (new images to be analysed) will be absorbed and, instead of creating a new prototype, will serve to move its centre and modify its area of influence.

The area of influence of a cluster represents a region of space within which all the images falling in will be considered as having been derived from the same prototype. In Angelov and Soares [76], it is considered that two vectors whose angles are less than 30 degrees apart are pointing in the same direction, so they can be considered to be derived from the same prototype. Assuming that the vectors are unitary, the maximum Euclidean distance between them is $r_{max} = \sqrt{2 - 2cos(30°)}$, corresponding to the distance between two unit vectors with a 30° angle between them. This distance is used by *xDNN Evolving* as an auxiliary classification criterion.

The training process concludes with the definition of a prototypes set for each class, where each prototype is represented as a point in the latent space, and has an associated cluster and the set of samples within its area of influence. The inference process consists in using the *CNN* to represent the new images in the latent space and calculating the similarity degree according to the proximity between each image and all the prototypes learned. The label associated to the prototype with the highest similarity degree for each image will be the category associated to it. The explainable capacity of this algorithm lies in the possibility to visually compare the input image with the prototype with the highest degree of similarity.

### 2.3.2. xDNN offline method

Similar to the *xDNN Evolving* method, the *xDNN Offline* approach also employs a CNN as a feature extractor. This CNN is responsible for projecting the input images into a latent space, enabling a subsequent clustering process to identify the set of prototypes for each class. The classification process in this method relies on measuring the similarity between these prototypes and the input images, as elaborated in the previous method.

The primary distinction between these two methods lies in how they identify the set of prototypes. In the *xDNN Offline* method, the final set of prototypes per class is obtained in three stages; i) Data rank ordering and prototypes identification, ii) Voronoi Tessellation creation, and (iii) Local Modes filtering.

In the first stage, all the data samples in the same class, contained into set $\{u\}_L$, where $L$ is the number of samples in the class, are ranked in an indexing list denoted by $\{z\}_L$. The samples in $\{u\}_L$ are ranked based on their relative distances to the closest neighbours in the latent space. Then, a set of samples that represent the local maxima in terms of *Typicality*, denoted by $\{p\}_k$, where $k$ is the number of local maxima, is identified. In order to do this, the value of *Typicality* is calculated for all the samples in the class, which is obtained through Eq. (2). Notably, the *Typicality* value corresponds to the *Data Density* of a specific input, as defined in Eq. (1), normalised by all the *Data Density* values of the inputs in the same class. The first element of $\{z\}_L$ is the sample with the higher value of *Typicality*.

$$\tau(x_i) = \frac{D(x_i)}{\Sigma_{j=1}^{L} D(x_j)}, (i = 1, 2, \ldots, L, u_i \in \{u\}_L) \tag{2}$$

The remaining elements of $z_L$ are determined through the application of Eq. (3), in which $d(x, y)$ represents the Euclidean distance between points $x$ and $y$. At the time that a new element $u_m$ is added to $\{z\}_L$, the same element is removed from $\{u\}_L$ with the aim of exploring all samples, without having repeated elements in $\{z\}_L$. The ordered list $\{z\}_L$ is systematically explored to pinpoint local maxima. If $\tau(z_i) > \tau(z_i - 1)$ and $\tau(z_i) > \tau(z_i + 1)$, then $z_i$ is identified as one of the local maxima and is subsequently included in $\{p\}_K$, which consists in the initial set of prototypes with $K$ elements.

$$z_{j+i} \leftarrow u_m;\ m = \underset{u_i \in \{u\}_L}{\text{argmin}}(d(u_i, z_j)) \tag{3}$$

In the second stage, each local maxima generates a cluster surrounding itself, comprising all data samples closer to it than to any other prototypes, including the local maxima itself. This procedure is equivalent to the creation of Voronoi tessellations. Each cluster is associated

with a support value representing the number of samples contained within it. The centre of the cluster, labelled as $C_i$, corresponds to the central point among its constituent members.

The last stage of the prototypes identification consists in removing some of the less representative clusters that were identified in the previous stage, in order to obtain larger and more descriptive clusters. To achieve this objective, in Angelov and Gu [78] a method to estimate the average distance between the strongly connected clusters, referenced by $\varpi$, is presented. The cluster whose centre has the highest value of *Typicality* among those inside the area defined by distance $\varpi$, is considered as the most representative cluster within that area and is maintained and all the other clusters in the area are eliminated. The images belonging to the eliminated clusters are then absorbed by the closest remaining clusters.

This criterion ensures that only small and less significant clusters that exhibit substantial overlap with larger and more important ones will be eliminated during the filtering process. This is accomplished by applying multiplicative weights based on the respective supports of the clusters to the *Typicality* of its centres.

Following the filtering operation, the chosen local maxima serve as the new prototypes for generating Voronoi tessellations, much like the procedure detailed in stage 2. Subsequently, these prototypes are subjected to another round of filtering in stage 3. This process of creating Voronoi tessellations and subsequent filtering is iterated until the distances between all the clusters' centres surpass the threshold of $\varpi$. The final set of filtered prototypes resulting from this process is then employed to execute the classification procedure.

## 3. Proposed method

Prototype-based classifiers possess a distinctive characteristic: they can use the identified prototypes as the basis for their classification process and explanations of their decisions. As a result, the performance of these classifiers, both in the classification task and in the explanation of results, depends on the representativeness of the identified prototypes. Current deep clustering methods primarily focus on separating the representations of inputs belonging to different classes, while bringing those belonging to the same class closer together. The main objective of these methods is to create a global maximum for each class, around which all samples of the same class are concentrated, far away from other classes, facilitating the classification process. However, this approach may not be ideal for prototype-based classifiers, as they rely on various local maxima that need to be distinct from each other. This distinction is essential to avoid redundancy and represent the diversity within the class, facilitating the generation of more interpretable explanations [79]. Moreover, prototype-based classifiers do not require all prototypes of the same class to be tightly clustered for optimal performance. They simply need the clusters to be sufficiently far apart to distinguish them from clusters of different classes [80].

In this context, inspired by Weinberger and Saul [80],Khosla et al. [81],Joe et al. [82],Chechik et al. [83], we propose the ExPIC, a cluster-oriented training strategy designed to enhance the performance of whole-image-based prototype classifiers while preserving the most representative local maxima to improve the explainable capacity of the classifier. This section presents the different components of the proposed method. In Section 3.2, an enhanced feature extraction approach is presented, as well as a specific Fine Tuning technique, which, when used iteratively with a new cluster-oriented loss function (presented in Section 3.3), enables the classifier to find more representative prototypes. In Section 3.4, we introduce an optimisation technique for the proposed method that reduces time and memory complexities while improving the classifier's performance, both in terms of classification Accuracy and explainable capabilities. Finally, at the end of this section, an overview of the proposed method with all its components is provided.

### 3.1. Overall pipeline

In this section, we describe the overall pipeline of the proposed explainable classifier method. Fig. 2 illustrates this method, which involves using only the convolutional layers of a *CNN* as a Feature Extractor to project images into a latent space. Unlike the approach described in Angelov and Soares [76], the Fully Connected layers are not used. The high-dimensional vectors, denoted as $\Omega$, obtained from the Feature Extractor, are then passed through a *DR* block to obtain lower-dimensional vectors, denoted as $\omega$. The optimal number of components for the reduction of *Feature Vectors* $\Omega$ may vary according to the dataset under consideration. These vectors serve as input for the *ADP* methods. The *ADP* methods have two different processes; training and inference.

In the *Training mode*, the *ADP* outputs the identified prototypes, which are subsequently used during the inference process. With this initial set of prototypes (identified using the *Feature Vectors* obtained from the CNN without re-training), a Cluster-Oriented Dataset is created. Each feature vector $\mathbf{x}$ is assigned two prototypes: the nearest one from the correct class, denoted as $\mathbf{y}$, and the nearest one from the wrong class, denoted as $\mathbf{z}$. The proposed cluster-based loss function, Cluster Density Error (CDE), use these prototypes and the *Feature Vector* $\mathbf{x}$ for re-training the CNN, aiming to enhance its performance as a feature extractor for the ADP methods. This re-training process enhances Accuracy levels and reduces the required number of prototypes for accurate image classification. Fine-tuning techniques can be applied by progressively unfreezing each *CNN*'s layer. The selection of the optimal layer where the training process should stop varies depending on the dataset. In the inference process, the identified prototypes are used to classify each Feature Vector $\omega$ by assigning them to the class with the highest confidence score. This score is computed using the softmax function applied to the negative Euclidean distances between the input image and the closest prototype of each class in the latent space. This approach ensures that the highest confidence score is assigned to the class with the closest prototype.

### 3.2. Enhanced feature extraction and fine tuning

In Angelov and Soares [76] and Angelov and Soares [84], it is recommended to use the output values of the last Fully Connected layer of the networks to create the latent space. However, some information may be "lost" in the transformations performed on these layers. With the aim of preserving as more information as possible, we suggest that the features are extracted from the last convolutional layer of the network.

Since the *VGG-16* network was trained to classify generic images into 1000 classes, the *Fine Tuning* technique was used to re-train it. This is done by gradually "unfreezing" its layers and optimising it for the current *dataset*. Initially, the last layer is unfrozen and retrained for a few epochs. Subsequently, the other layers are unfrozen one by one, while keeping the previously unfrozen layers still trainable, until the entire network is unfrozen. In order to assess the performance of this approach, a comparison study between the original method and these two improvements was carried. The results of this study are presented in Section 4.4.

### 3.3. Cluster density error

The inference process of the *xDNN* methods assigns to each new image the label that is associated with the nearest prototype in the latent space. A classification error occurs when, for a given new image, a prototype (also known as cluster *centroid*) that belongs to the "wrong" class is closer than one from the correct class.

An initial approach to address this issue is to adjust the Feature Extractor to minimise the distance between the extracted *Feature Vector* of an image and its "right cluster", which refers to the nearest cluster

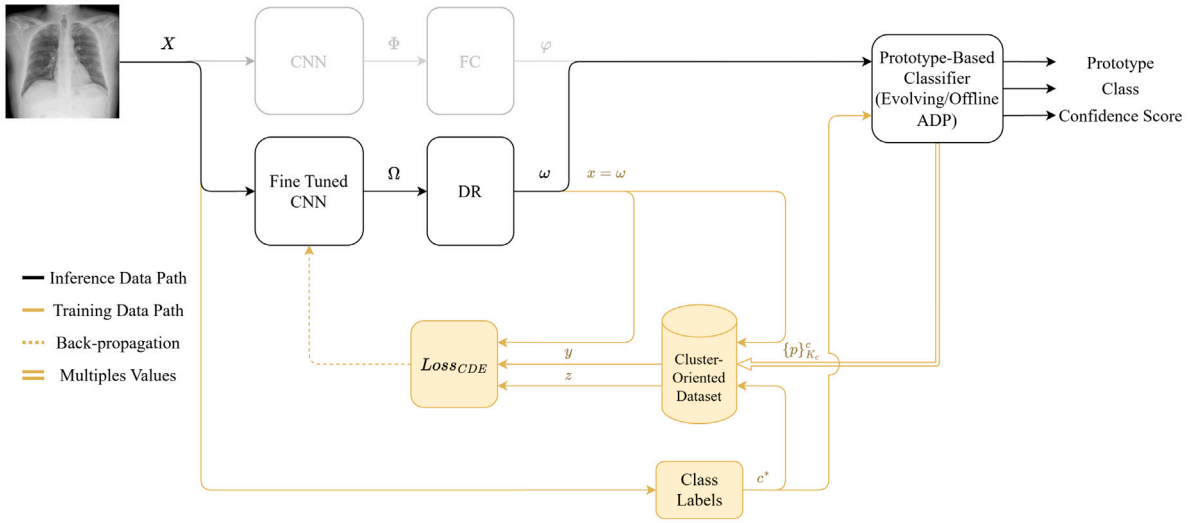



**Fig. 2.** Proposed method's pipeline. Grey path: Generic *xDNN* classification method architecture; Black path: Proposed classification method architecture on inference time; Fine Tuned *CNN*: Re-trained *CNN*; *DR*: Dimensionality Reduction; Yellow path: Proposed classification method architecture in training mode; Class Labels: Index of the class associated to each image; Cluster-Oriented Dataset: Dataset conformed by the nearest cluster centroids (prototypes) to each image; $Loss_{CDE}$: Loss function used during the re-training process of the *CNN*. Black line: Inference data path; Yellow line: Training data path; Dashed line: Back-propagation error; Double line: Multiple Feature Vectors.

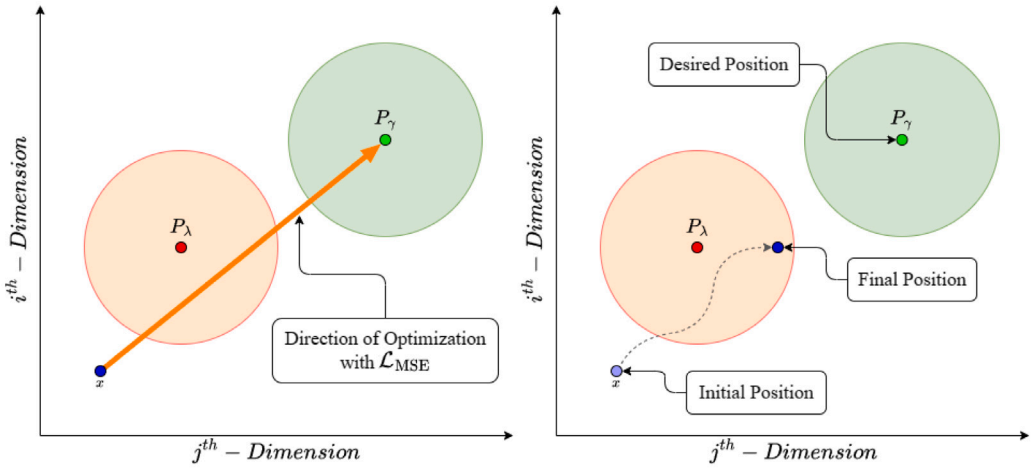



**Fig. 3.** Clustering optimisation limitation: attempting to reach the "right" cluster ($P_\gamma$), the algorithm might lead to the "wrong" one ($P_\lambda$). The figure on the left illustrates a straight path for a single image, while the figure on the right shows a possible path when multiple inputs (images) are considered.

belonging to the correct class. This can be achieved during the Fine Tuning process of the *CNN* by penalising greater distances between cluster members and their associated *centroids*. By implementing this approach, the dispersion of cluster members can be reduced, resulting in clearer separation between clusters and mitigating the problem of overlapping.

To apply this solution, it is necessary to have an initial set of prototypes (with a cluster associated to each one) in order to establish the direction in which the *Feature Vectors* of each image (from the training set) should be directed. Following each Fine Tuning stage, where one additional layer is unfrozen and retrained alongside the other unfrozen layers, either *xDNN Evolving* or *xDNN Offline* can be applied to generate a new set of prototypes that are adapted to the updated network configuration.

The initial approach of this method focuses on minimising the distance between each image and its respective correct cluster centroid. For this purpose, the *Mean Squared Error (MSE)* $\mathcal{L}_{\text{MSE}}(\mathbf{x}, \mathbf{y})$, as described by Eq. (4), is used. The choice of the *MSE* metric, and its variants, for clustering optimisation was based in its widespread use in this kind of applications [85–88]. In our particular case, in Eq. (4), $\mathbf{x} \in \mathbb{R}^N$ is a *Feature Vector* of the current image, $\mathbf{y} \in \mathbb{R}^N$ is the nearest cluster centroid to $\mathbf{x}$, and $x_n$ and $y_n$ are the $n$th components of the vectors $\mathbf{x}$ and $\mathbf{y}$ respectively.

$$\mathcal{L}_{\text{MSE}}(\mathbf{x}, \mathbf{y}) = \frac{1}{N} \sum_{n=1}^{N} (x_n - y_n)^2 \tag{4}$$

The neural network training process aims to minimise $\mathcal{L}_{\text{MSE}}(\mathbf{x}, \mathbf{y})$, that meaning to reduce the Euclidean distance of each *Feature Vector* to its nearest prototype/cluster. However, during the (re-)training phase, the attempt to place the features within the area of influence of the prototypes belonging to the correct class may end up with the Feature Vector being placed within the area of influence of the prototype belonging to the wrong class, as shown in Fig. 3. This behaviour is due to the fact that the $\mathcal{L}_{\text{MSE}}(\mathbf{x}, \mathbf{y})$ function only considers a single target position. In our case, this position corresponds to $P_\gamma$, the nearest prototype belonging to the correct class.

To improve the model's performance, the loss function may also consider the position of the prototypes belonging to any of the other

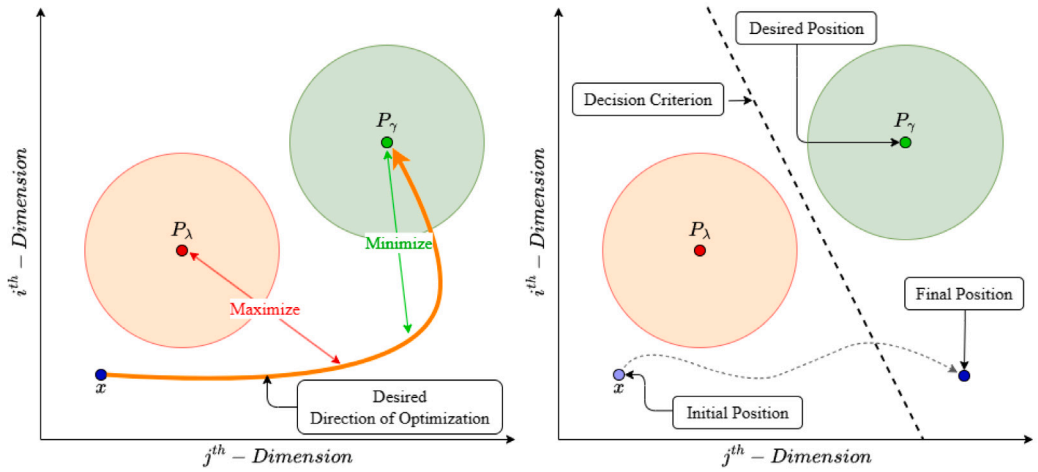



**Fig. 4.** Improved optimisation process with new *CDE loss function: The search path to reach the "right" cluster ($P_\gamma$)* now also tries to maximise the distance to the wrong one ($P_\lambda$). Despite the final distance to the "right" cluster centre, it is closer to the "right" than to the "wrong" one, thus enabling a correct classification. The left figure depicts an ideal path for a single *Feature Vector* (image), while the right one shows a potential path when multiple images are taken into account.

non-correct classes and try to maximise it in order avoid placing the *Feature Vector* $x$ within the area of influence of an incorrect prototype. This concept, illustrated in Fig. 4, serves as the fundamental principle behind the proposed cluster-oriented loss metric, that we name as *CDE*. This metric, provided in Eq. (5), has two terms where the first calculates the *MSE*, as defined in Eq. (4), between the *Feature Vector* $\mathbf{x}$ and its respective correct cluster centroid $\mathbf{y}$ and the second one is the inverse of the *MSE* between $\mathbf{x}$ and an incorrect cluster centroid $\mathbf{z}$ in addition with a value $\epsilon$ corresponding to a very low value to avoid divisions per 0 when the position of $\mathbf{x}$ is the same as $\mathbf{z}$. Furthermore, a novel user-adjustable parameter called $\alpha$ is introduced in this equation. This parameter determines the significance assigned to moving the Feature Vector away from the incorrect prototypes.

$$\mathcal{L}_{\text{CDE}}(\mathbf{x},\mathbf{y},\mathbf{z}) = (1-\alpha)\cdot\mathcal{L}_{\text{MSE}}(\mathbf{x},\mathbf{y}) + \alpha\cdot\frac{1}{\mathcal{L}_{\text{MSE}}(\mathbf{x},\mathbf{z})+\epsilon} \tag{5}$$

The $\mathcal{L}_{\text{CDE}}$, as demonstrated by the experimental results presented in Section 4, not only enhances the precision of the model, but also has the additional benefit of reducing the number of prototypes used. This improvement in efficiency contributes to the model's explainability capabilities.

In order to calculate the $\mathcal{L}_{\text{CDE}}$, one must determine both the correct cluster centroid $\mathbf{y}$, and the incorrect cluster centroid $\mathbf{z}$. As such, it is necessary to create a "Cluster-Oriented Dataset". This dataset is built during the training process and must be updated each time the network weights are changed. This dataset associates each *Feature Vector* $x$ with one prototype from the correct class and one from the remaining classes. Considering the *Feature Vector* $\mathbf{x}\in\mathbb{R}^N$ belonging to the class $c^*\in C$, where $C$ is the set of all the classes, and considering $\{p\}^c_{K_c}$ being the set of prototypes from the class $c$ with $K_c$ elements, it is possible to associate a target cluster $\mathbf{y}$ belonging to class $c^*$ and an avoidance cluster $\mathbf{z}$ belonging to class $a\in C\setminus\{c^*\}$ to $\mathbf{x}$ through Eqs. (6) and (7), where $d(\mathbf{a},\mathbf{b})$ is the Euclidean distance between vectors $\mathbf{a}$ and $\mathbf{b}$.

$$\mathbf{y} = \underset{p_k\in\{p\}^{c^*}_{K_{c^*}}}{\text{argmin}}\{d(p_k,\mathbf{x})\} \tag{6}$$

$$\mathbf{z} = \underset{p_k\in\{p\}^{a}_{K_a}}{\text{argmin}}\{d(p_k,\mathbf{x})\} \tag{7}$$

The value of the $\alpha$ parameter is dependent on the specific problem and should be optimised based on the requirements at hand. One approach to obtain a reference value is to calculate a power of 10 that balances the loss terms. Therefore, the initial value of $\alpha$ can be determined using Eqs. (8) and (9). In Eq. (9), the "floor" value of $\eta$ is used. This choice is made with the intention of equalising the loss terms

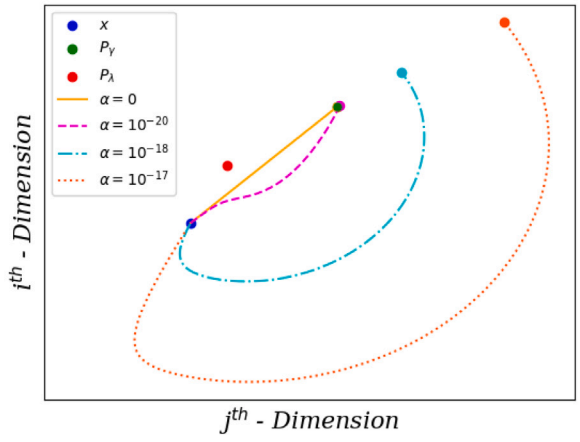



**Fig. 5.** *CDE* optimisation Paths varying the value of $\alpha$.

to the same order of magnitude while preserving their differences. By doing so, a reference value for $\alpha$ is established from which other values can be explored and evaluated.

$$\eta = \log_{10}\left(\mathcal{L}_{\text{MSE}}(x,y)\cdot\mathcal{L}_{\text{MSE}}(x,z)\right) \tag{8}$$

$$\alpha_{ref} = 10^{\lfloor\eta\rfloor} \tag{9}$$

The value of $\alpha$ directly influences the distance between the wrong prototype and the optimisation path. A higher value of $\alpha$ results in a greater separation, pushing the optimisation path further away from the wrong prototype. Conversely, a smaller value of $\alpha$ reduces the impact of the wrong prototype, causing the optimisation path to closely resemble a straight line connecting the initial position of $x$ to the correct prototype. This behaviour is illustrated in Fig. 5.

As expected, the optimisation process using the $\mathcal{L}_{\text{CDE}}$ function with a given Feature Vector, increasing the value of $\alpha$ causes the optimisation paths to diverge further away from the wrong prototype. On the other hand, selecting smaller values of $\alpha$ brings the optimisation paths closer to the wrong prototype, at the point that it is completely ignored and a direct path towards the correct prototype is taken when $\alpha$ is set to 0. The experimental results presented in Section 4.4 illustrate how the iterative process of the Fine Tuning technique using the $\mathcal{L}_{\text{CDE}}$ loss function leads to an improvement in the Accuracy values as more

layers are gradually unfrozen. Additionally, the results demonstrate a reduction in the number of prototypes required by the *xDNN* methods.

### 3.4. Dimensionality reduction

Using the $\mathcal{L}_{\text{CDE}}$ loss function requires identifying the closest prototype/cluster from the correct class as well as from other classes. However, this process may become computationally intensive, especially when dealing with a large number of prototypes, due to the increment of the number of prototypes which extends the search time for the nearest clusters. Therefore, the training process becomes infeasible. To address this issue, the number of dimensions in the *Feature Vectors* produced by the *CNN* is reduced using the *Principal Component Analysis (PCA)* technique. This reduction in dimensionality helps mitigating the computational burden, enabling a more manageable and efficient training process.

In the literature, Sufficient Dimensionality Reduction (SDR) algorithms [89–94] are commonly employed to handle large datasets effectively. These algorithms contribute to enhance interpretability while reducing computational complexity, thereby enabling the application of various classification and clustering algorithms that would otherwise be infeasible with the original high-dimensional data [95].

Consequently, numerous studies have demonstrated the effectiveness of SDR algorithms, specifically highlighting the case of *PCA* and its variants [96], as a preprocessing step in tasks such as clustering and classification. *PCA* offers two fundamental applications, namely dimensionality reduction and noise reduction [97–99], making it highly advantageous for tasks involving high-dimensional datasets.

The core concept behind *PCA* is to reduce the data dimensionality while retaining the maximum "variability". This is achieved by determining new variables ("dimensions") that are linear combinations of the original ones. These new variables, known as principal components, are selected to maximise variance and remain uncorrelated with each other. As described in Wang and Liu [100], the process of applying the *PCA* technique can be summarised in three main steps. First, given a training dataset $V = \{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_M}\} \in \mathbb{R}^{N \times M}$, where $M$ represents the number of training samples, all *Feature Vectors* $\mathbf{v_i}$ are centralised, and the covariance matrix $Q$ of the *Feature Vectors* is computed. Each training sample $\mathbf{v_i} \in V$ satisfies the following condition:

$$\mathbf{v_i} = \mathbf{v_i} - \frac{1}{M}\sum_{j=1}^{M} \mathbf{v_j}, \tag{10}$$

$$Q = \frac{1}{M} V V^T. \tag{11}$$

Next, the eigenvalues of the covariance matrix $Q$ and their corresponding eigenvectors $H = \{\mathbf{h_1}, \mathbf{h_2}, \ldots, \mathbf{h_N}\} \in \mathbb{R}^{N \times N}$ are calculated and arranged in descending order according to their eigenvalues. Finally, the first $d$ eigenvalues and their corresponding eigenvectors, denoted as $H' = \{\mathbf{h_1}, \mathbf{h_2}, \ldots, \mathbf{h_d}\} \in \mathbb{R}^{N \times d}$, are selected to form the projection matrix $W$ for dimensionality reduction. The result of applying *PCA* is then given by:

$$Y = W^T V, \tag{12}$$

where $Y$ represents the $d$-dimensional output dataset.

Dimensionality reduction is performed by selecting the $d$ components with the highest variance. This value varies for each problem and dataset. The use of different values for $d$ naturally results in different Accuracy values and computational/time complexity.

## 4. Experimental assessment

In this section we present the results of the conducted experiments to validate the proposed method. The datasets used in the experiments are introduced in Section 4.1. An overview of the results obtained is provided in Section 4.3. Additionally, to gain a better understanding of the individual contributions of each component of the proposed method, an ablation study is presented in Section 4.4.

**Table 1**
Number of images assigned to the training, validation and test sets for each dataset used.

| Dataset | Split | | | Total |
|---|---|---|---|---|
| | Training | Validation | Test | |
| *COVID-CT* [101] | 428 | 109 | 209 | 746 |
| *COVID-QU-Ex* [102] | 21 715 | 5417 | 6788 | 33 920 |
| CBIS-DDSM [103] | 1054 | 264 | 378 | 1696 |
| INbreast [104] | 216 | 54 | 117 | 387 |

### 4.1. Datasets

In this study, four different medical image datasets were used. The first dataset, *COVID-CT* Dataset, is presented in Yang et al. [101]. It consists of 349 *CT scan* chest images obtained from 249 patients diagnosed with *COVID-19*, and 463 images collected from 55 healthy patients sourced from various datasets. This dataset, was used by the authors of Angelov and Soares [76] to validate the *xDNN* method. In Yang et al. [101], the dataset authors performed experimental studies which demonstrated that this dataset is useful for developing AI-based diagnosis models of *COVID-19* based on *CT Scans*.

Using this dataset, they developed diagnosis methods based on multi-task learning and self-supervised learning. According to a prominent radiologist who has been diagnosing and treating *COVID-19* patients at Tongji Hospital in Wuhan, China, ever since the disease first appeared there, models with an accuracy and F1-Score greater than 89% are good enough for clinical usage [101].

The second dataset used is the so called *COVID-QU-Ex* Dataset [102], which includes over 33,000 Chest X-ray (CXR) images, from three different classes: 11,956 *COVID-19* cases, 11,263 *Non-COVID* infections (viral or bacterial pneumonia) cases, and 10,701 Normal (healthy) cases. This dataset was created based on different publicly available repositories, all of which are scattered and with varying formats. In Tahir et al. [102], an extensive set of experiments was performed using state-of-the-art segmentation networks, and the developed network reached superior performance for lung region segmentation and *COVID-19* infections localisation.

The quality of this dataset was ensured through a rigorous quality control process where duplicates, extremely low-quality, and over-exposed images were identified and removed [102]. The resulting dataset thus comprises images of high interclass dissimilarity with few varying resolutions, quality, and SNR levels. The ground-truth lung segmentation masks for all CXR images were gathered using a collaborative human–machine segmentation approach that significantly reduces human labour to annotate the images.

In order further assess the effectiveness of the proposed method in other than only pulmonary conditions, we also considered commonly used mammographic datasets in mass/nodules classification tasks, such as the *CBIS-DDSM* [103] and *INbreast* [104] datasets. The *CBIS-DDSM* dataset consists of 1,696 mammography images and is widely used for training and testing in mammographic mass classification. To ensure comparability with other studies, we followed the recommended training and test split for mass classification. In the INbreast dataset, images were labelled according to the BI-RADS assessment categories. Following the approach in Zhang et al. [105], we classified images with a BI-RADS score of 1 or 2 as benign masses, and those with a score of 4, 5, or 6 as malignant masses. Label 3 was excluded because the INbreast dataset does not assign a definitive category for this label, resulting in a total of 387 images. The division of the dataset into training, validation, and test sets is shown in Table 1.

### 4.2. Experimental conditions

In this work, the *VGG-16*[2] neural network, pre-trained on the *ImageNet dataset* [106], was used as the Feature Extractor for all experiments. During training, as a pre-processing step, the images were converted from grey-scale to BGR, and each colour channel was zero-centred based on the *ImageNet dataset* [106].

The experiments were conducted on an NVIDIA GeForce RTX 3090 GPU and a CPU Xeon Gold 6336Y CPU @ 2.40 GHz. In each Fine Tuning stage, the model was trained for up to 100 epochs using the Adam optimiser [107] with a batch size of 64 and a learning rate of $5 \times 10^{-5}$. A reduce-on-plateau mechanism was applied to adjust the learning rate, using a factor of 0.5 with a relative threshold of 0.0001 and a patience of 10 epochs. The minimum learning rate was set to $10^{-10}$. Additionally, early stopping was implemented with a relative threshold of 0 and a patience of 52 epochs.

### 4.3. Results

The first studies carried out with the *xDNN Evolving* method, involve reproducing the experiments conducted by the authors and comparison of results. In this studies, the *COVID-CT* Dataset, referenced in [76] (and detailed in Section 4.1), was used in order to compare the results obtained locally with those published by the authors.

In Table 2, the results obtained replicating the experiments conducted by the authors in Angelov and Soares [76] are presented and denoted as Vanilla *xDNN Evolving*, alongside the results they show in the paper. Although very close, it is possible to note that the reproduced results are slightly below than those published in the article. This is due to the sensitivity of the algorithm to the set of images designated for training and testing: in our reproduction, we tried to replicate the experimental conditions used in Angelov and Soares [76], however the number of images in each set and the partition between training and testing was not disclosed by authors in the original paper. Thus, we could not guarantee that the sets had exactly the same images in the two implementations. Only using exactly the same dataset partition could lead to the same results since the datasets composition is determinant to build the clusters of the prototypes.

Our reproduction of the *xDNN Evolving* method was applied to the *Feature Vectors* provided by the authors for their own dataset published in Soares et al. [108]. In this experiment, the obtained outcomes perfectly match the reported results for all the performance metrics outlined in the publication. This validation reaffirms that the prototype-based classifier described in Angelov and Soares [76] has been effectively reproduced and that any deviations in the results can be attributed to variances in dataset splitting procedures.

While the accuracy values achieved in the reproduced method closely align with those reported by the authors, there is a significant disparity in the number of prototypes required to construct the classification model. Specifically, the replicated method necessitates the use of a considerably larger number of prototypes, totalling 387. Notably, this number of prototypes is close to the total number of images within the training set, which stands at 428. Consequently, the classification model has nearly one image assigned to each cluster, including its own centre, thereby compromising the model's interpretability. To provide a fair comparison between our proposals and the methods proposed by Angelov et al. for the remainder of the experiments, our reproduction of their methods will be used as reference and the training (428), testing (209), and validation (109) sets will be kept the same.

To address this issue, the alternative version of the *xDNN Evolving* method, namely the *xDNN Offline*, was employed on the same dataset. As indicated in Table 2, the Vanilla *xDNN Offline* method yields a reduced number of prototypes, at the cost of a slight reduction in accuracy. Given the behaviour of the model and the reduced number of prototypes offered by the *xDNN Offline* method, it has been selected as the baseline for the proposed method.

In Table 2, it is evident how the implementation of the proposed method leads to a significant improvement in performance for both classification methods, namely *xDNN Evolving* and *xDNN Offline*. In the table, the results of our proposals are denoted as ExPIC Evolving and ExPIC Offline. This improvement is reflected in increased Accuracy values and a reduction in the number of clusters/prototypes required to construct the model, thereby enhancing the system's explainability. The degree of interpretability of a prototype-based model depends on the representativeness of its prototypes. Indeed, a model with a large number of prototypes can be seen as less interpretable: a large number of different prototypes in a given class means that the model was not so well succeeded in grouping the training samples according their common features; or, from another perspective, if each prototype resemble only a few training samples – sometimes just two or three – it may indicate that the model failed to capture common features within a class. It is worth noting that in a direct comparison between both Evolving Methods our proposal yields an increase of 8.13% in accuracy and a reduction of about 13% in the number of prototypes. When comparing the Offline methods, our proposal yield an increase of 14.88% in accuracy and a reduction of about 83% in the number of prototypes. Finally, we can notice that our methods follow the trend of the two versions of xDNN, where there is a trade-off between accuracy and number of prototypes between the Evolving and Offline versions. However, it was noticed that the use of the *DR* improves the models accuracy and yields a reduced number of prototypes, enhancing the explainable capacities of the system.

As previously stated, to further evaluate the effectiveness of the proposed method, other datasets with different image modalities were also considered. The results of these experiments are presented in Tables 3–5, for the datasets *COVID-QU-Ex*, *CBIS-DDSM*, and *INbreast*. It becomes evident how the proposed method enhances the performance of prototype-based classifiers, specifically the *xDNN* methods, enabling them to identify more representative samples. Remarkably, for the *COVID-QU-Ex* dataset, this leads to a substantial 98.38% reduction in the number of identified prototypes while simultaneously achieving a 21.54% increase in Accuracy levels when using the ExPIC Offline method. It is worth noting that the proposed method presents a slightly lower accuracy than the baseline [102]. However, this was expected, since the baseline model is not explainable. Thus, once again, we can observe the trade-off between accuracy and explainability of the observed results.

In Table 4, the experimental results using the *CBIS-DDSM* dataset are presented and benchmarked against two other methods. Again, our proposed approach demonstrates highly competitive performance. When compared to the best performing benchmark, it shows a slight reduction in accuracy (0.56%) and AUC score (1.5%). However, this performance decrease is acceptable, as it enables the model to provide explanations for its predictions.

Similarly, in Table 5, the results obtained when using the *INbreast* dataset are presented and compared with two other methods. When compared with [105], and as before, it can be observed that the explainable capabilities of our model come with a marginal cost in terms of performance (Accuracy and AUC). To ensure a fair comparison with [110], in addition to the original training and testing splits, we used a split of 80% for training and 20% for testing, replicating their experimental conditions and reported the corresponding AUC results. Under these conditions, the proposed method achieved an AUC score of 86.47%, outperforming that model by 0.47%, while also adding an explainable component to the system. It is also worth noticing the substantial reduction in the number of prototypes achieved with our proposed approach when compared with other prototype-based methods. These results highlight the effectiveness of the proposed method when applied to various datasets and image modalities.

[2] The *VGG16* network used in this context is the pre-trained version available within the Keras framework.

**Table 2**
Comparison of the results between the reference *xDNN* methods and the proposed method ExPIC method applied on the *COVID-CT* Dataset.

| Model | Metric | | | | | |
|---|---|---|---|---|---|---|
| | Accuracy | AUC | Precision | Recall | F1 Score | Num. of Prototypes |
| *xDNN Evolving* [76] | 88.60% | N/A | 89.70% | 88.60% | 89.20% | 63 |
| Vanilla *xDNN Evolving* | 85.17% | 0.8972 | 88.51% | 78.57% | 83.24% | 387 |
| Vanilla *xDNN Offline* | 77.99% | 0.8784 | 76.53% | 76.53% | 76.53% | 109 |
| ExPIC Evolving (Proposed) | **93.30%** | **0.9614** | **94.68%** | 90.82% | **92.71%** | 337 |
| ExPIC Offline (Proposed) | 92.82% | 0.9599 | 91.09% | **93.88%** | 92.46% | **18** |

**Table 3**
Comparison of the results obtained on the *COVID-QU-Ex* Dataset between its baseline (a black-box model), the reference *xDNN* methods and the proposed method ExPIC method.

| Model | Metric | | | | | |
|---|---|---|---|---|---|---|
| | Accuracy | AUC | Precision | Recall | F1 Score | Num. of Prototypes |
| Baseline [102] | **99.23%** | N/A | **99.14%** | **99.31%** | **98.20%** | N/A |
| Vanilla *xDNN Evolving* | 78.65% | 0.9012 | 79.06% | 78.67% | 78.67% | 17 912 |
| Vanilla *xDNN Offline* | 73.47% | 0.9016 | 76.06% | 72.72% | 71.84% | 2659 |
| ExPIC Evolving (Proposed) | 96.27% | 0.9862 | 96.20% | 96.18% | 96.19% | 5210 |
| ExPIC Offline (Proposed) | 95.01% | **0.9922** | 94.88% | 94.89% | 94.89% | **43** |

**Table 4**
Comparison of the results obtained on the *CBIS-DDSM* Dataset between benchmark non-interpretable methods, the reference *xDNN* methods and the proposed ExPIC method.

| Model | Metric | | | | | |
|---|---|---|---|---|---|---|
| | Accuracy | AUC | Precision | Recall | F1-Score | Num. Prototypes |
| [109] | **74.90%** | **0.8040** | N/A | N/A | N/A | N/A |
| [110] | N/A | 0.7900 | N/A | N/A | N/A | N/A |
| Vanilla *xDNN Evolving* | 63.23% | 0.6360 | 52.74% | 52.38% | 52.56% | 1002 |
| Vanilla *xDNN Offline* | 63.49% | 0.6600 | 53.15% | 51.70% | 52.41% | 236 |
| ExPIC Evolving (Proposed) | 72.75% | 0.7390 | 66.92% | 59.18% | 62.82% | 737 |
| ExPIC Offline (Proposed) | 74.34% | 0.7890 | **68.94%** | **61.90%** | **65.23%** | **216** |

**Table 5**
Comparison of the results obtained on the *INbreast* Dataset between benchmark non-interpretable methods, the reference *xDNN* methods and the proposed ExPIC method.

| Model | Metric | | | | | | |
|---|---|---|---|---|---|---|---|
| | Accuracy | AUC | AUC[a] | Precision | Recall | F1-Score | Num. Prototypes |
| [105] | **87.93%** | **0.9054** | N/A | **93.77%** | **53.63%** | **68.23%** | N/A |
| [110] | N/A | N/A | 0.8600 | N/A | N/A | N/A | N/A |
| Vanilla *xDNN Evolving* | 64.10% | 0.5368 | 0.6754 | 34.28% | 38.71% | 36.36% | 193 |
| Vanilla *xDNN Offline* | 70.09% | 0.5289 | 0.7029 | 42.86% | 38.71% | 40.68% | 48 |
| ExPIC Evolving (Proposed) | 80.34% | 0.7029 | 0.7686 | 66.67% | 51.61% | 58.18% | 62 |
| ExPIC Offline (Proposed) | 81.20% | 0.7464 | **0.8647** | 73.68% | 45.16% | 56.00% | **13** |

[a] The AUC score values of this column were calculated using 20% of the dataset for testing to allow the fair comparison with the benchmark method [110]. The images used for this calculus were taken from the test set used to calculated the other metrics.

### 4.4. Ablation study

To analyse the individual contribution of each component of the proposed method, the results of an ablation study for both classification methods, ExPIC Evolving and ExPIC Offline, are presented in Table 6. These experiments were conducted over the test set of images of the *COVID-CT* Dataset. Highlighted in BOLD are the overall best results in terms of Accuracy and the number of prototypes, while the best outcome for each classification method is underlined for clarity. In general, the ExPIC Evolving method yields higher Accuracy values when using a classification-oriented trained network, such as the original *VGG16* network trained on the *ImageNet* dataset [106] and combined with traditional Fine Tuning techniques. This holds true whether using all the features extracted from the last Fully Connected layer or from the last convolutional layer. However, when using only a subset of the principal components of the *Feature Vectors*, the system's performance is superior towards the employment of the ExPIC Offline method for the identification of clusters/prototypes.

On the other hand, when employing the proposed cluster-oriented training method, the ExPIC Offline classification method achieves superior Accuracy values while reducing the number of clusters/prototypes required to build the model. In all scenarios, the ExPIC Offline method attains highly competitive performance with a remarkably low number of prototypes.

#### 4.4.1. Enhanced feature extraction

Upon individual component analysis, it becomes apparent that the system's performance is enhanced when using "rawer" features, i.e., those extracted from the last convolutional layer (25088 features) rather than those from the Fully Connected ones (4096 features). This suggests that there might be some loss of information in the latter.

However, in the case of the ExPIC Evolving method, as the number of prototypes approaches the total number of training images for each class (228 for *Non-COVID* and 200 for *COVID*), some of its explainability capacity may be compromised. On the other hand, when employing the ExPIC Offline method, the usage of "rawer" features leads to an

**Table 6**
Ablation Study: Comparing the impact of individual components in proposed method on Accuracy levels and number of prototypes on test ("unseen") images of the *COVID-CT* Dataset.

| Classification method | | Feature extraction layer | | Fine tuning | *PCA* | $\mathcal{L}_{CDE}$ | Accuracy | Num. of prototypes |
|---|---|---|---|---|---|---|---|---|
| *Evolving* | *Offline* | Last FC | Last Conv. | | | | | |
| ✔ | | ✔ | | | | | 85.17% | 387 |
| ✔ | | ✔ | | ✔ | | | 92.34% | 283 |
| ✔ | | ✔ | | | ✔ | | 80.38% | 339 |
| ✔ | | ✔ | | ✔ | ✔ | | 92.34% | 241 |
| ✔ | | ✔ | | ✔ | | ✔ | 90.43% | 112 |
| ✔ | | ✔ | | ✔ | ✔ | ✔ | 91.39% | 235 |
| ✔ | | | ✔ | | | | 87.56% | 421 |
| ✔ | | | ✔ | ✔ | | | 93.30% | 421 |
| ✔ | | | ✔ | | ✔ | | 87.56% | 420 |
| ✔ | | | ✔ | ✔ | ✔ | | 92.34% | 393 |
| ✔ | | | ✔ | ✔ | | ✔ | 90.91% | 420 |
| ✔ | | | ✔ | ✔ | ✔ | ✔ | **93.30%** | 337 |
| | ✔ | ✔ | | | | | 77.99% | 109 |
| | ✔ | ✔ | | ✔ | | | 90.43% | 108 |
| | ✔ | ✔ | | | ✔ | | 81.34% | 111 |
| | ✔ | ✔ | | ✔ | ✔ | | 91.87% | 109 |
| | ✔ | ✔ | | ✔ | | ✔ | 91.39% | 66 |
| | ✔ | ✔ | | ✔ | ✔ | ✔ | 91.39% | 58 |
| | ✔ | | ✔ | | | | 82.78% | 110 |
| | ✔ | | ✔ | ✔ | | | 89.95% | 102 |
| | ✔ | | ✔ | | ✔ | | 84.21% | 107 |
| | ✔ | | ✔ | ✔ | ✔ | | 89.00% | 102 |
| | ✔ | | ✔ | ✔ | | ✔ | 91.39% | 65 |
| | ✔ | | ✔ | ✔ | ✔ | ✔ | 92.82% | **18** |

increase in classification Accuracy while simultaneously reducing the number of identified clusters/prototypes.

#### 4.4.2. Fine tuning

Irrespective of the classification algorithm employed, the Fine Tuning process proves effective in raising the Accuracy values. To carry out a traditional Fine Tuning process without using the $\mathcal{L}_{CDE}$ loss function, the MLP classifier of the *CNN*, which originally consisted of 1000 outputs (one for each category of ImageNet), is replaced with simpler classifier containing only two outputs to perform binary classification. This modified network is then trained as a whole.

For the Fine Tuning process, a fix batch size and learning rate of 64 and $5 \times 10^{-5}$ was used. These hyper-parameters were used for all the experiments that implied re-training the network. Following the complete pipeline of the proposed method, the best result among all the conducted experiments is obtained using the features extracted from the last convolutional layer and applying the *Offline* classification method.

Given that the models obtained from different Fine Tuning stages achieve varying Accuracy values, Table 6 presents only the highest Accuracy value and its corresponding number of prototypes.

#### 4.4.3. Cluster density error

To apply the $\mathcal{L}_{CDE}$ loss function effectively, an optimal value for $\alpha$ needs to be determined. The process involves calculating a reference value of $\alpha$ as described in Section 3.3. Following this, adjacent values are explored to find the optimal one. The *Optuna* software tool [111] was used for this purpose. The reference value varies depending on the classification method used and whether dimensionality reduction was applied. As can be seen in Table 6, the use of the $\mathcal{L}_{CDE}$ loss function during the Fine Tuning process leads to an improvement in the Accuracy values while reducing the number of identified prototypes, improving the explainable capabilities of the classifier, specially for the Offline classifier. The optimal $\alpha$ values for the ExPIC Evolving and ExPIC Offline methods are approximately $540.87 \times 10^{-6}$ and $101.09 \times 10^{-6}$, respectively. These values achieve the best performance for each method.

#### 4.4.4. Dimensionality reduction

In order to apply *DR*, an initial estimation of the optimal projection of the *Feature Vectors* is calculated using the *PCA* technique. This step is crucial for enhancing accuracy while minimising the number of prototypes. The projection is calculated once per Fine Tuning stage as a pre-processing step for the Prototype-based classifier during training. It is then used statically during inference and throughout the training of the *CNN* in the subsequent Fine Tuning stage. When applying *PCA* to the problem at hand, it is essential to determine the appropriate number of dimensions, denoted as $d$, to which the original *Feature Vectors* should be projected. To identify the optimal value for $d$, an optimisation study was conducted using the Optuna tool [111].

The relationship between the number of components and the associated Accuracy values, for the validation set with 109 images, is shown in Fig. 6. It can be seen that an increase in the number of components leads to an improvement in the Accuracy up to a certain point. Once the number of principal components exceeds approximately 120, the Accuracy values of the ExPIC Offline method reach a plateau, with an average value of 85%. This implies that including additional components beyond this threshold does not yield significant improvements in Accuracy (in the figure the maximum number of components is limited by the number of training samples in the dataset). It is worth noting that using all the original dimensions of the *Feature Vectors*, comprising 25088 components, results in an Accuracy value of 85.65% without any form of re-training of the *CNN VGG-16*, while using the *PCA* technique this value can be increased up to 89.95%.

Similarly, Fig. 7 allows us to analyse the impact of the number of components on the number of prototypes generated by the ExPIC Offline method. The number of prototypes increases gradually as the number of components increases, but at a faster pace compared to the Accuracy values. This trend continues until it reaches a stable point at around 135 prototypes, which occurs after approximately 50 components. Beyond roughly 300 components, there is a slight decrement in the number of prototypes, but it remains relatively stable (in the figure the maximum number of components is limited by the number of training samples in the dataset).

Fig. 8 reveals a significant observation regarding execution time for the model in the COVID-CT dataset: as expected, a higher number of components requires longer to run. The execution displays a rapid

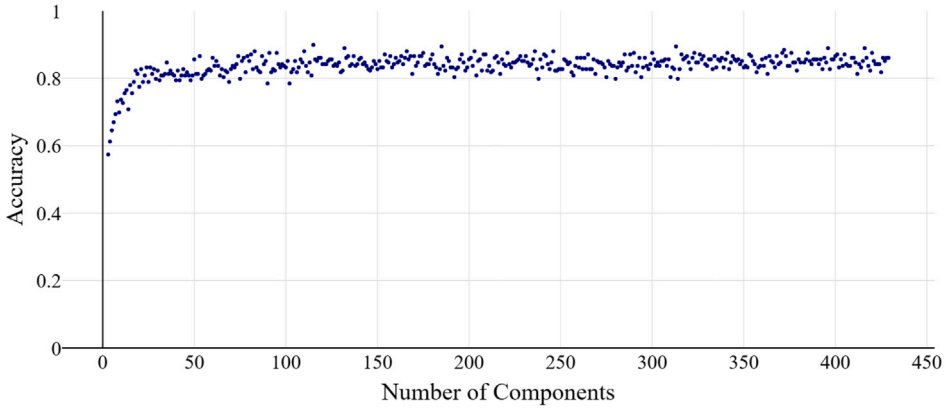



**Fig. 6.** Accuracy values of the Offline classifier, on the validation set of the *COVID-CT* Dataset, performing *PCA* dimensionality reduction for different number of components.

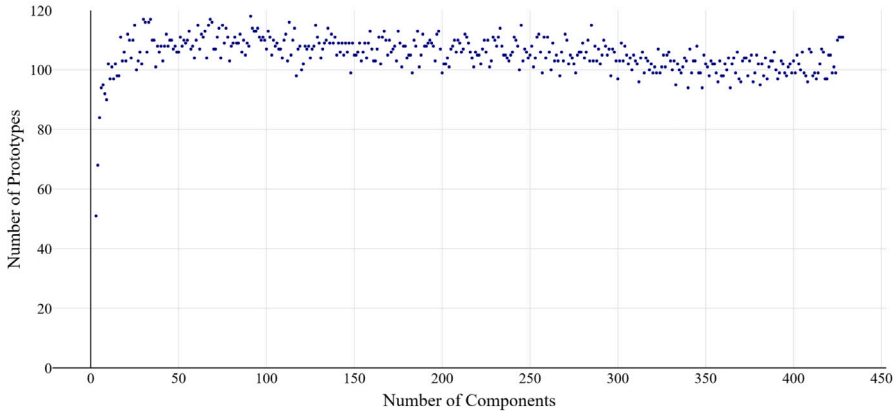



**Fig. 7.** Number of prototypes generated by the Offline classifier, on the validation set of the *COVID-CT* Dataset, performing *PCA* dimensionality reduction for different number of components.

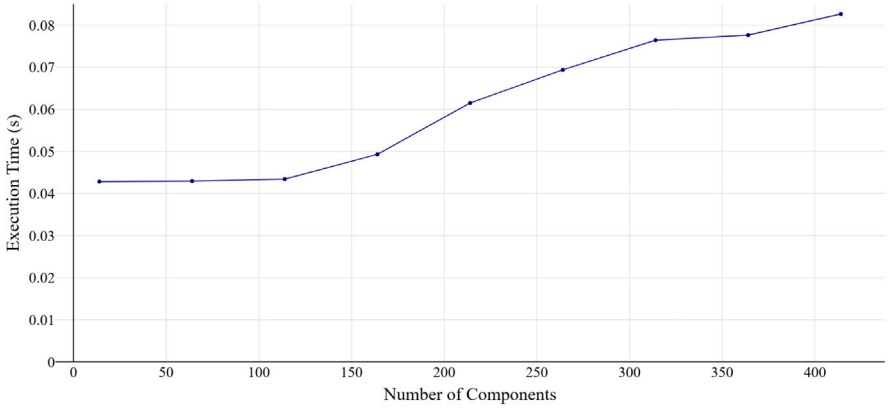



**Fig. 8.** Sampling of different Execution Times of the Offline classifier, on the validation set of the *COVID-CT* Dataset, performing *PCA* dimensionality reduction for different number of components, with a step of 50 components.

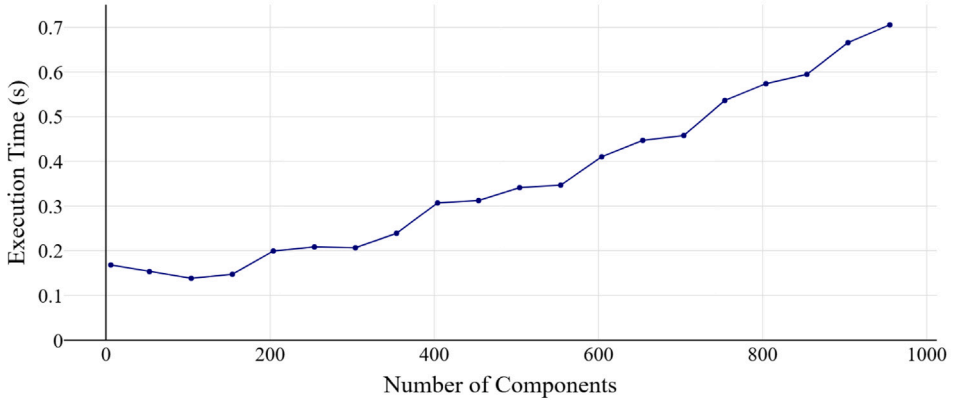



**Fig. 9.** Sampling of different Execution Times of the Offline classifier, on the validation set of the *CBIS-DDSM* Dataset, performing *PCA* dimensionality reduction for different number of components, with a step of 50 components.

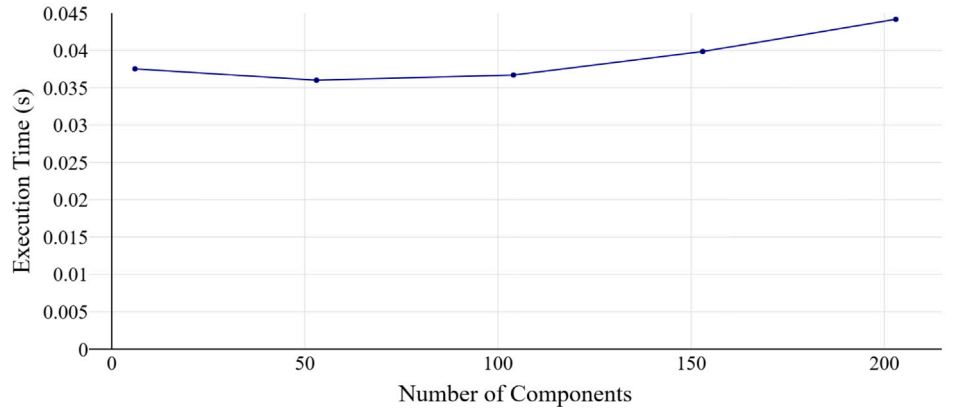



**Fig. 10.** Sampling of different Execution Times of the Offline classifier, on the validation set of the *INbreast* Dataset, performing *PCA* dimensionality reduction for different number of components, with a step of 50 components.

increase with the number of components, peaking at approximately 82.66 ms required for 414 components. Without the use of PCA to reduce the number of components (thus using the original number of components, 25088) the algorithm takes around 18 s to run. When compared with these figures, even the 83 ms required by the 414 components are orders of magnitude smaller. This behaviour is also confirmed in Figs. 9 and 10 for *CBIS-DDSM* and *INbreast* datasets, respectively.

The execution time might be seen as practically insignificant for the application of the proposed method on smaller datasets such as the *COVID-CT* Dataset. However, when dealing with larger datasets, as the *COVID-QU-Ex* Dataset, the experiments carried out with the original dimensionality of 25088 components take around 6 h to identify the prototypes using the ExPIC Offline method and 3 h to complete the inference process. In this scenario, dimensionality reduction using the *PCA* technique becomes critical to reduce the time complexity when applying the proposed method, which involves an iterative application of the classification methods.

Fig. 11 illustrates how the use of the *PCA* technique can significantly reduce the time required to identify the prototypes in the *COVID-QU-Ex* Dataset. The execution time using *PCA* in this dataset presents a rapid linear increase, with an execution time of approximately 998.75 s (around 16.65 min) required for 1062 components. It is worth noting that, similar to the *COVID-CT* Dataset, Accuracy values stabilise around 83% after around 300 components in the *COVID-QU-Ex* Dataset.

To allow the Fine Tuning process described in Section 3.2, using the $\mathcal{L}_{\text{CDE}}$ loss function, to run in an acceptable time, the Execution Time of the classifiers should be reduced to less than 5 min. This implies a constraint on the number of components that can be used to represent the *Feature Vectors*. This restriction requires the use of the *PCA* dimensionality reduction for *COVID-QU-Ex* Dataset.

Returning to the primary case study of this work, the *COVID-CT* Dataset, and considering the trade-off between Accuracy, the number of prototypes, and Execution Time, the number of components selected was 199. This choice resulted in an Accuracy of 84.21%, 107 prototypes/clusters for the two classes, and an Execution Time of 56.43 ms. This serves as the baseline from which the optimisation process with Fine Tuning, using the $\mathcal{L}_{\text{CDE}}$ loss function, starts.

Reducing the dimensionality of the *Feature Vectors* enables the possibility of performing the Fine Tuning process using the $\mathcal{L}_{\text{CDE}}$ loss function, even in cases where memory capacity our processing power is limited. Fig. 12(a) illustrates that using the $\mathcal{L}_{\text{CDE}}$ loss function for re-training the *CNN* leads to an increase in Accuracy values as the earlier layers are progressively unfrozen, peaking in the $13^{th}$ stage. This results in a notable 8.61% improvement, achieving an Accuracy level of 92.82%. This improvement is substantial compared to the original Accuracy value of 77.99% obtained when using all dimensions of the *Feature Vectors* from the last Fully Connected layer without re-training the CNN.

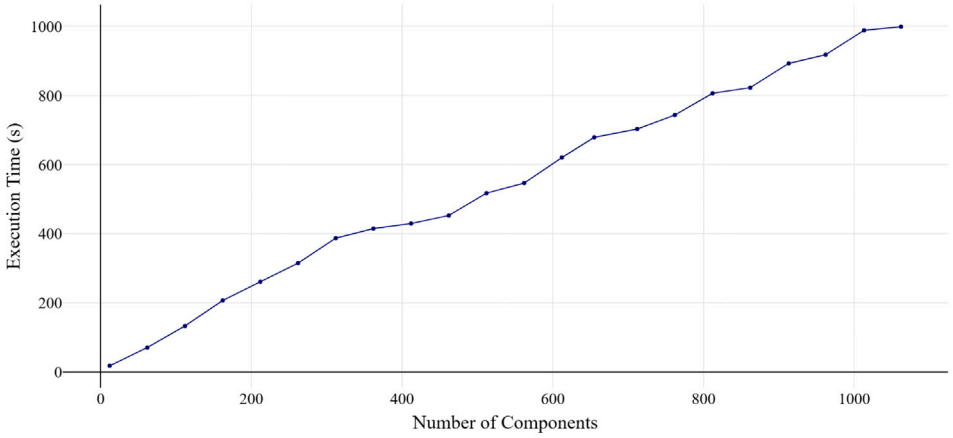
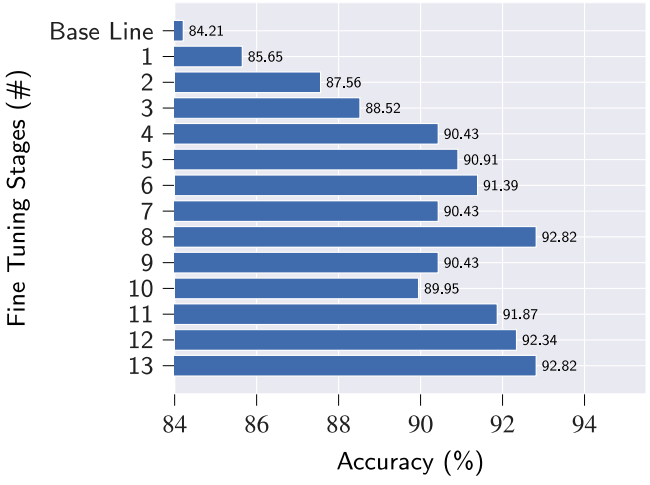
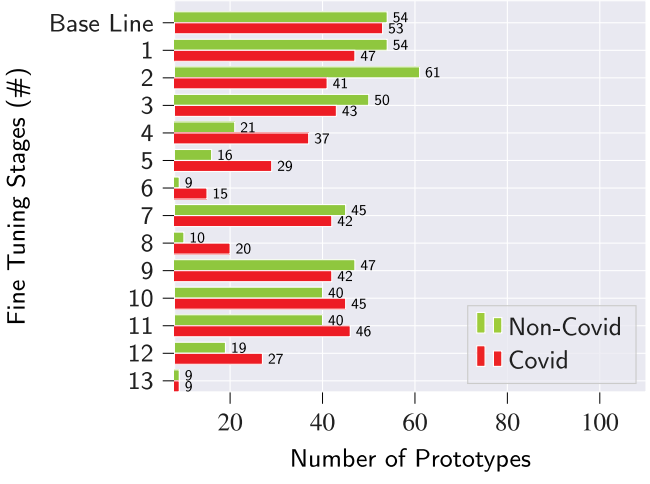



**Fig. 11.** Sampling of different Execution Times of the Offline classifier, on the validation set of the *COVID-QU-Ex* Dataset, performing *PCA* dimensionality reduction for different number of components, with a step of 50 components.




(a) Accuracy values




(b) Number of identified prototypes

**Fig. 12.** Results obtained using the ExPIC Offline method, on the test set of the *COVID-CT* Dataset, fine-tuning the all the convolutional layers (one per stage) with the *CDE* loss function and performing a *PCA* dimensionality reduction from 25088 to 119 components.

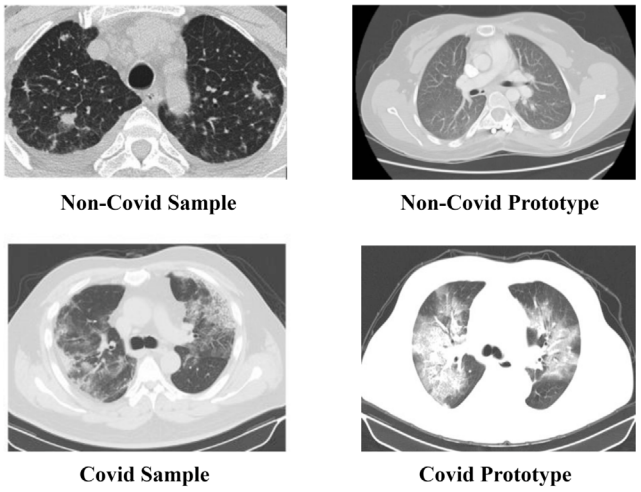



**Fig. 13.** Explanation examples for COVID-CT dataset. Left: Randomly selected input image from the test set; Right: Corresponding prototype associated by the model.

However, the most remarkable observation pertains to the number of prototypes shown in Fig. 12(b). As the number of prototypes decreases, the *Accuracy* values for the respective stages increase, reaching their highest values in the stages with a lower number of prototypes. Conversely, the stages with the highest number of prototypes achieve the lowest Accuracy levels. This behaviour underscores the efficiency of the proposed method in enhancing the model's performance in both classification and explainability. The total number of prototypes decreased from 109 to 18 for all classes, showcasing a remarkable 95.35% reduction in the number of prototypes compared with the method proposed in Angelov and Soares [76] (first row in Table 6).

Figs. 13–16 exemplify the explainable nature of the algorithm: for an image under analysis, a classification label is determined and the associated prototype displayed. The left column presents randomly selected samples from the datasets in use (test subsets) and the algorithm's predicted label; in the right column the prototype leading to the classification of the input sample is presented. This prototype is determined as the nearest image in the training set to the centroid of the cluster to which the sample was assigned in the classification process.

In Figs. 17 and 18, additional examples are presented to demonstrate the explainability of the proposed method. Each example includes

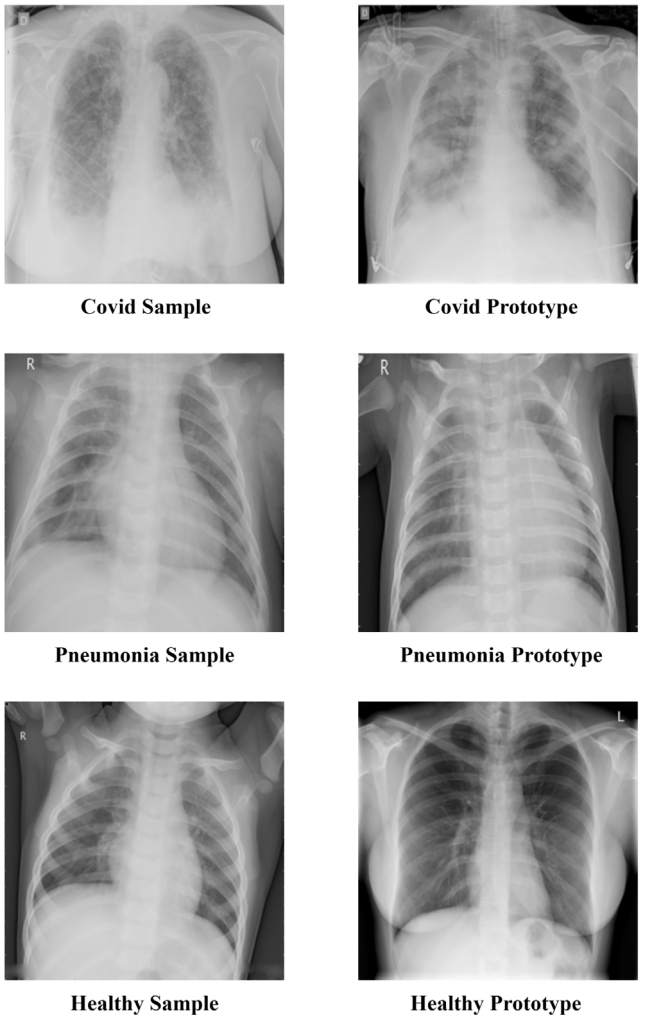



**Fig. 14.** Explanation examples for COVID-QU-Ex dataset. Left: Randomly selected input image from the test set; Right: Corresponding prototype associated by the model.

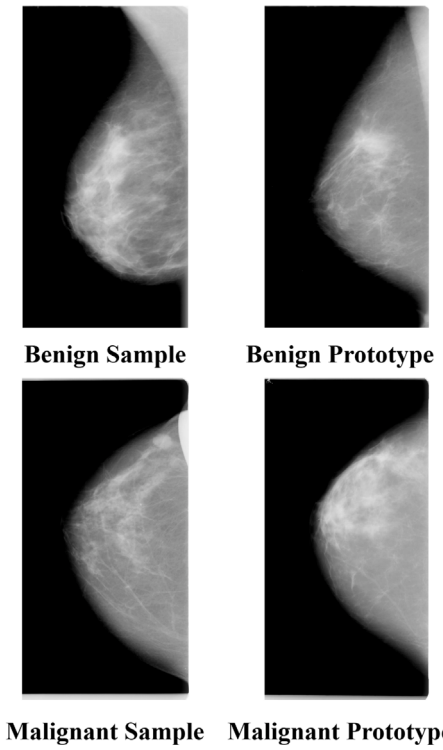



**Fig. 15.** Explanation examples for CBIS-DDSM dataset. Left: Randomly selected input image from the test set; Right: Corresponding prototype associated by the model.

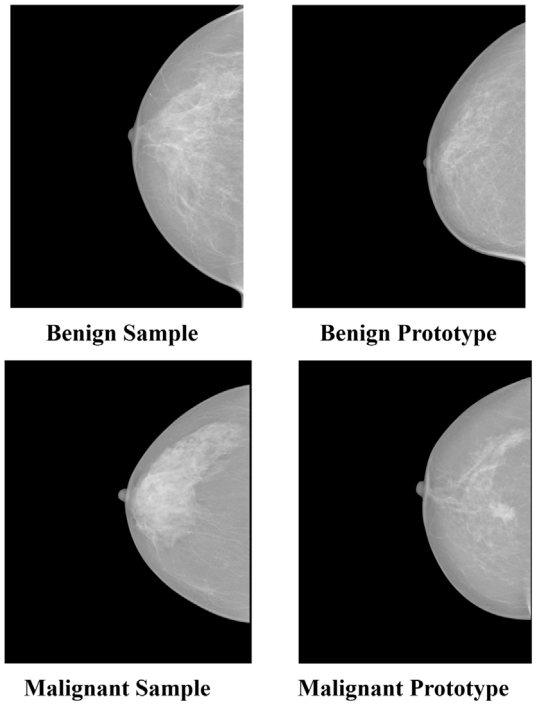



**Fig. 16.** Explanation examples for INbreast dataset. Left: Randomly selected input image from the test set; Right: Corresponding prototype associated by the model.

the input image, the predicted class along with the model's confidence score, and the closest prototype. To further emphasise the explainability aspect, a gradient-based method was applied to generate attention maps for both the input image and its corresponding prototype. These maps highlight the key regions in both images that, due to their similarity, were crucial for the classification decision. The attention maps were created using the Integrated Gradients method [42] combined with the SmoothGrad method [43].

The effectiveness of applying *PCA* for dimensionality reduction is particularly pronounced when dealing with high-dimensional feature sets, namely those obtained from the last convolutional layer. Conversely, the features derived from the last fully connected layer already undergo an intrinsic recombination, diminishing the impact of *PCA* dimensionality reduction on these features.

The results presented in this section demonstrate two interesting behaviours. Firstly, the results in Section 4.4.4 showcase the "compressibility" property of *PCA*, as observed by Mukherjee and Zhang [97], which effectively reduces the distance between data points (*Feature Vectors*) belonging to the same clusters while mildly reducing inter-cluster distances. This property explains the Accuracy improvements illustrated in Fig. 6. Secondly, the $\mathcal{L}_{\text{CDE}}$ loss function follows a similar principle but also reduces the number of prototypes by increasing the

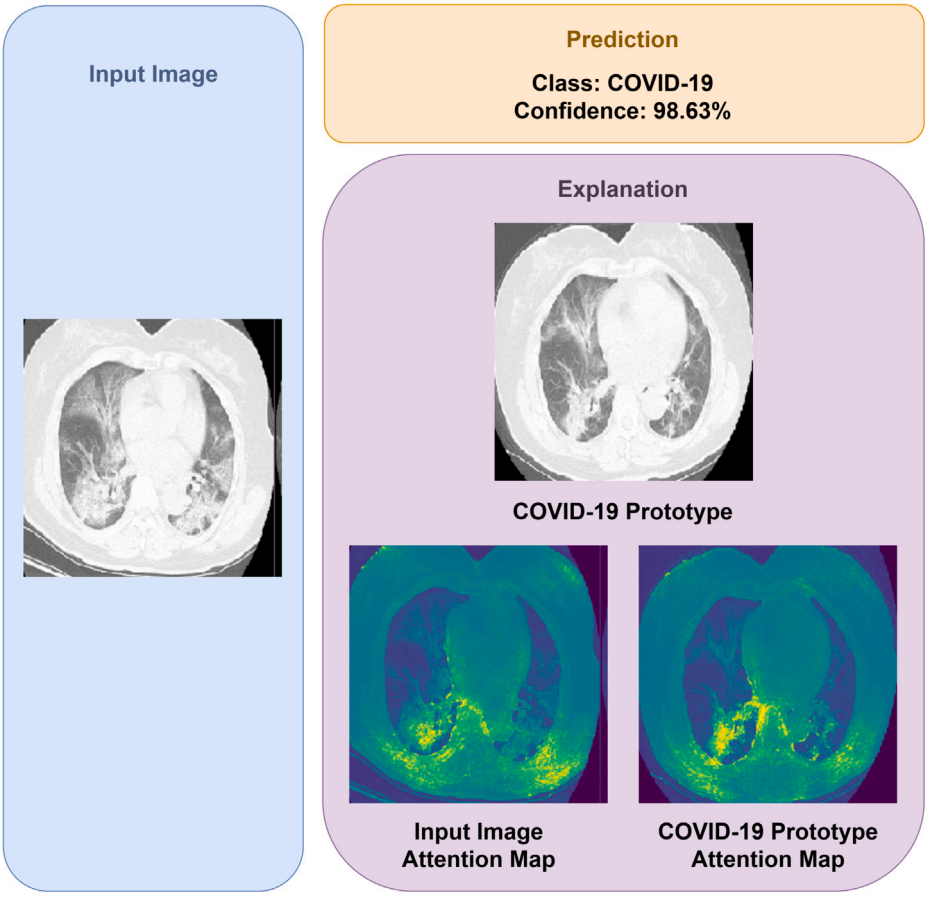



**Fig. 17.** Additional example of the method explainability for the *COVID-CT* dataset. Left: A randomly selected test image. Right-Top: Prediction result displaying the predicted label and confidence value. Right-Bottom: Prediction explanation showing the corresponding prototype and attention maps for both the input image and the prototype.

distance between clusters from different classes, facilitating the classification task. When used together, both *PCA* dimensionality reduction and Fine Tuning with the $\mathcal{L}_{\text{CDE}}$ loss function effectively enhance the Accuracy values of the model while improving its explainability.

## 5. Conclusion

In this work, we introduced the ExPIC method, a cluster-oriented training strategy designed to enhance the performance of prototype-based classifiers, which possess the advantage of being able to provide explanations for their decisions. Such classifiers are particularly valuable in fields where result interpretability is crucial, such as healthcare.

The primary contribution of the proposed method centres on the introduction of a novel cluster-based loss function, referred to as *Cluster Density Error (CDE)*. This loss function embodies the core idea behind our approach, which is to minimise the distance between each image representation in the latent space and its corresponding correct cluster while maximising the distance to the incorrect one. Furthermore, the proposed loss function introduces an adjustable parameter known as $\alpha$, which allows for tailoring the loss function to specific problem requirements.

The underlying concept of the proposed loss function aligns with that of contrastive loss functions, but it diverges in the selection of similar and dissimilar pairs, aiming to create multiple local maxima instead of a single global maximum. This approach is intended to enhance the explainable capacity of prototype-based classifiers, which use these local maxima to provide explanations for their decisions.

In this work, we demonstrated how the application of the proposed loss function, in conjunction with the Fine Tuning technique, leads to improved classification performance while enhancing the classifier's explainable capacity. The results presented showcase how, across various stages, the classifier is capable of identifying more representative prototypes, thereby reducing the number of clusters while preserving the most dissimilar ones for use in the explanation process.

Furthermore, it was demonstrated that the inclusion of a *DR* stage during the feature extraction process, instead of using a Fully Connected layer, results in improved classification outcomes. We demonstrated that incorporating dimensionality reduction estimation during training enables the implementation of more complex and powerful loss functions, which in turn facilitates the creation of more intelligent systems capable of addressing more complex problems. Moreover, the use of this technique allows for the application of the proposed loss function even in scenarios where time complexity and memory usage are critical or limited.

The results presented in this work demonstrate that the proposed method surpasses the baseline classifiers not only in classification performance, but also in explainable capacity by identifying more representative prototypes to elucidate the decisions. The capability of the proposed method to identify more representative local maxima, rather than merely separating different classes, makes it an ideal choice for prototype-based classifiers to perform explainable classification. The development of more explainable systems is critical to facilitate the widespread use of ML-based systems in healthcare.

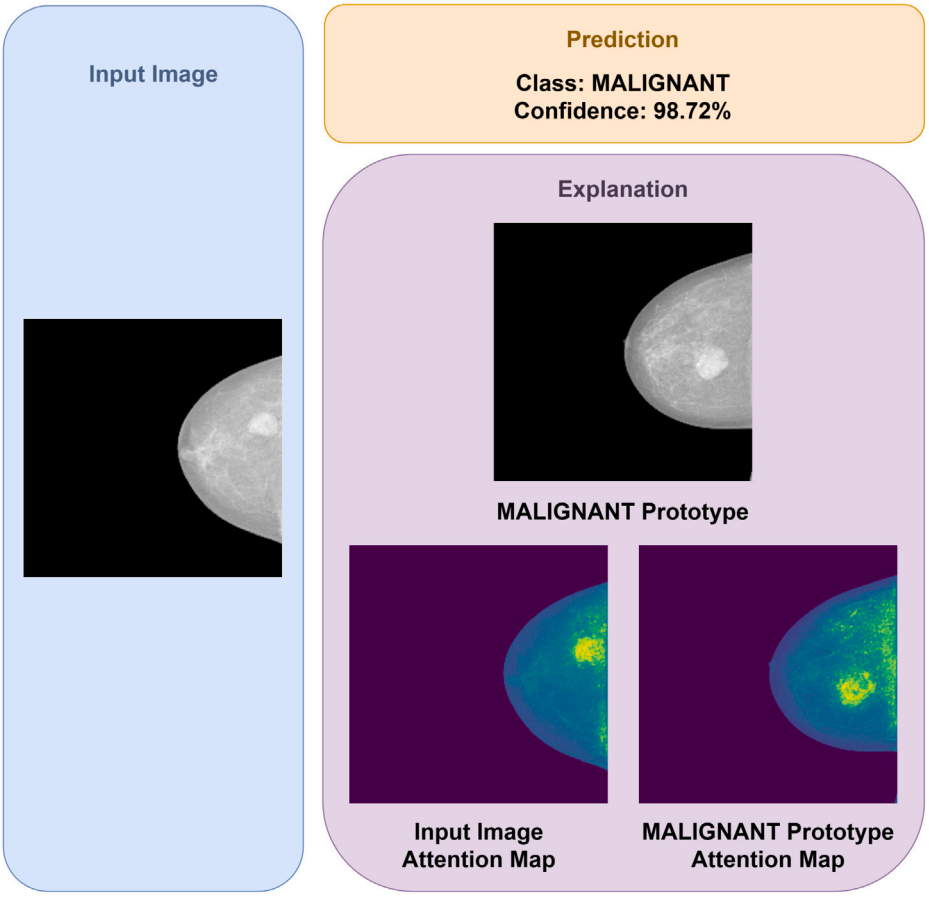



**Fig. 18.** Additional example of the method explainability for the INbreast dataset. Left: A randomly selected test image. Right-Top: Prediction result displaying the predicted label and confidence value. Right-Bottom: Prediction explanation showing the corresponding prototype and attention maps for both the input image and the prototype.

## CRediT authorship contribution statement

**Nicolas Vasconcellos:** Writing – original draft, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Luis M.N. Tavora:** Writing – review & editing, Supervision, Resources, Project administration, Formal analysis, Conceptualization. **Rolando Miragaia:** Writing – review & editing, Visualization, Validation, Formal analysis, Data curation. **Carlos Grilo:** Writing – review & editing, Visualization, Validation, Formal analysis, Data curation. **Lucas A. Thomaz:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition, Formal analysis, Conceptualization.

## Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used OpenAI-ChatGPT to improve language and readability. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgements

This work is funded by the Polytechnic of Leiria under the project "1° Prémio + Ciência 2024", national funds through FCT – Fundação para a Ciência e a Tecnologia, I.P. under projects CoMBINNe 2022.09914.PTDC (DOI: 10.54499/2022.09914.PTDC), 2023.07886.CEECIND (DOI: 10.54499/2023.07886.CEECIND/CP2862/CT0003), and UID/4524/2025 (DOI: 10.54499/UID/04524/2025), and, when eligible, co-funded by EU funds under project/support UID/50008/2025 – Instituto de Telecomunicações, with DOI identifier 10.54499/UID/50008/2025 and LA/P/0109/2020 (DOI: 10.54499/LA/P/0109/2020).

## References

[1] A. Bhardwaj, Promise and provisos of artificial intelligence and machine learning in healthcare, J. Heal. Leadersh. 14 (2022) 113–118.

[2] L. Kaestner, Artificial intelligence meets hematology, Transfus. Apher. Sci. 59 (6) (2020) 102986.

[3] Y. Arai, T. Kondo, K. Fuse, Y. Shibasaki, M. Masuko, J. Sugita, T. Teshima, N. Uchida, T. Fukuda, K. Kakihana, Y. Ozawa, T. Eto, M. Tanaka, K. Ikegame, T. Mori, K. Iwato, T. Ichinohe, Y. Kanda, Y. Atsuta, Using a machine learning algorithm to predict acute graft-versus-host disease following allogeneic transplantation, Blood Adv. 3 (22) (2019) 3626–3634.

[4] S. Chan, V. Reddy, B. Myers, Q. Thibodeaux, N. Brownstone, W. Liao, Machine learning in dermatology: Current applications, opportunities, and limitations, Dermatol. Ther. (Heidelb.) 10 (3) (2020) 365–386.

[5] A. Sharma, R. Rani, A systematic review of applications of machine learning in cancer prediction and diagnosis, Arch. Comput. Methods Eng. 28 (7) (2021) 4875–4896.

[6] V. Gulshan, L. Peng, M. Coram, M.C. Stumpe, D. Wu, A. Narayanaswamy, S. Venugopalan, K. Widner, T. Madams, J. Cuadros, R. Kim, R. Raman, P.C. Nelson, J.L. Mega, D.R. Webster, Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs, JAMA 316 (22) (2016) 2402.

[7] E. Sorantin, M.G. Grasser, A. Hemmelmayr, S. Tschauner, F. Hrzic, V. Weiss, J. Lacekova, A. Holzinger, The augmented radiologist: artificial intelligence in the practice of radiology, Pediatr. Radiol. 52 (11) (2022) 2074–2086.

[8] A.A.-A. Valliani, D. Ranti, E.K. Oermann, Deep learning and neurology: A systematic review, Neurol. Ther. 8 (2) (2019) 351–365.

[9] A. Qayyum, J. Qadir, M. Bilal, A. Al-Fuqaha, Secure and robust machine learning for healthcare: A survey, IEEE Rev. Biomed. Eng. 14 (2021) 156–180.

[10] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.M. Jodoin, H. Larochelle, Brain tumor segmentation with Deep Neural Networks, Med. Image Anal. 35 (2017) 18–31.

[11] Zhennan Yan, Yiqiang Zhan, Zhigang Peng, Shu Liao, Y. Shinagawa, Shaoting Zhang, D.N. Metaxas, Xiang Sean Zhou, Multi-instance deep learning: Discover discriminative local anatomies for bodypart recognition, IEEE Trans. Med. Imaging 35 (5) (2016) 1332–1343.

[12] J. Schlemper, J. Caballero, J.V. Hajnal, A.N. Price, D. Rueckert, A deep cascade of convolutional neural networks for dynamic MR image reconstruction, IEEE Trans. Med. Imaging 37 (2) (2018) 491–503.

[13] J. Mehta, A. Majumdar, RODEO: Robust DE-aliasing autoencoder for real-time medical image reconstruction, Pattern Recognit. 63 (2017) 499–510.

[14] M. Anthimopoulos, S. Christodoulidis, L. Ebner, A. Christe, S. Mougiakakou, Lung pattern classification for interstitial lung diseases using a deep convolutional neural network, IEEE Trans. Med. Imaging 35 (5) (2016) 1207–1216.

[15] W. Shen, M. Zhou, F. Yang, C. Yang, J. Tian, Multi-scale convolutional neural networks for lung nodule classification, Inf. Process. Med. Imaging 24 (2015) 588–599.

[16] R. Yang, P. Liu, L. Ji, ProDiv: Prototype-driven consistent pseudo-bag division for whole-slide image classification, Comput. Methods Programs Biomed. 249 (2024) 108161, http://dx.doi.org/10.1016/j.cmpb.2024.108161.

[17] P. Liu, L. Ji, X. Zhang, F. Ye, Pseudo-bag mixup augmentation for multiple instance learning-based whole slide image classification, IEEE Trans. Med. Imaging 43 (5) (2024) 1841–1852, http://dx.doi.org/10.1109/TMI.2024.3351213.

[18] I. Straw, H. Wu, Investigating for bias in healthcare algorithms: a sex-stratified analysis of supervised machine learning models in liver disease prediction, BMJ Health Care Inf. 29 (1) (2022).

[19] Z. Obermeyer, B. Powers, C. Vogeli, S. Mullainathan, Dissecting racial bias in an algorithm used to manage the health of populations, Science 366 (6464) (2019) 447–453.

[20] Y.J. Juhn, E. Ryu, C.I. Wi, K.S. King, M. Malik, S. Romero-Brufau, C. Weng, S. Sohn, R.R. Sharp, J.D. Halamka, Assessing socioeconomic bias in machine learning algorithms in health care: a case study of the HOUSES index, J. Am. Med. Inf. Assoc. 29 (7) (2022) 1142–1151.

[21] C.J. Kelly, A. Karthikesalingam, M. Suleyman, G. Corrado, D. King, Key challenges for delivering clinical impact with artificial intelligence, BMC Med. 17 (1) (2019) 195.

[22] B. Ghoshal, A. Tucker, Estimating uncertainty and interpretability in deep learning for coronavirus (COVID-19) detection, 2020, CoRR, arXiv:2003.10769.

[23] L. Brunese, F. Mercaldo, A. Reginelli, A. Santone, Explainable deep learning for pulmonary disease and coronavirus COVID-19 detection from X-rays, Comput. Methods Programs Biomed. 196 (105608) (2020) 105608.

[24] M.R. Karim, T. Döhmen, M. Cochez, O. Beyan, D. Rebholz-Schuhmann, S. Decker, DeepCOVIDExplainer: Explainable COVID-19 diagnosis from chest X-ray images, in: 2020 IEEE International Conference on Bioinformatics and Biomedicine, BIBM, 2020, pp. 1034–1037, http://dx.doi.org/10.1109/BIBM49941.2020.9313304.

[25] E.E.D. Hemdan, M.A. Shouman, M.E. Karar, COVIDX-Net: A framework of deep learning classifiers to diagnose COVID-19 in X-ray images, 2020, arXiv.

[26] L. Wang, Z.Q. Lin, A. Wong, COVID-Net: a tailored deep convolutional neural network design for detection of COVID-19 cases from chest X-ray images, Sci. Rep. 10 (1) (2020) 19549.

[27] A.J. DeGrave, J.D. Janizek, S.I. Lee, AI for radiographic COVID-19 detection selects shortcuts over signal, Nat. Mach. Intell. 3 (7) (2021) 610–619.

[28] M.T. Ribeiro, S. Singh, C. Guestrin, "Why Should I Trust You?": Explaining the predictions of any classifier, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, Association for Computing Machinery, New York, NY, USA, 2016, pp. 1135–1144, http://dx.doi.org/10.1145/2939672.2939778.

[29] O. Davoodi, S. Mohammadizadehsamakosh, M. Komeili, On the interpretability of part-prototype based classifiers: a human centric analysis, Sci. Rep. 13 (1) (2023).

[30] J.A. McDermid, Y. Jia, Z. Porter, I. Habli, Artificial intelligence explainability: the technical and ethical dimensions, Philos. Trans. A Math. Phys. Eng. Sci. 379 (2207) (2021) 20200363.

[31] J. Wang, R. Fujimaki, Y. Motohashi, Trading interpretability for accuracy, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2015, pp. 1245–1254.

[32] S. Nazir, D.M. Dickson, M.U. Akram, Survey of explainable artificial intelligence techniques for biomedical imaging with deep neural networks, Comput. Biol. Med. 156 (2023) 106668, http://dx.doi.org/10.1016/j.compbiomed.2023.106668.

[33] C. Rudin, Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead, Nat. Mach. Intell. 1 (5) (2019) 206–215.

[34] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, F. Herrera, Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI, Inf. Fusion 58 (2020) 82–115.

[35] K. Bauer, O. Hinz, W. van der Aalst, C. Weinhardt, Expl(AI)n it to me – explainable AI and information systems research, Bus. Inf. Syst. Eng. 63 (2) (2021) 79–82.

[36] G. Yang, Q. Ye, J. Xia, Unbox the black-box for the medical explainable AI via multi-modal and multi-centre data fusion: A mini-review, two showcases and beyond, Inf. Fusion 77 (2022) 29–52.

[37] T. Danesh, R. Ouaret, P. Floquet, S. Negny, Hybridization of model-specific and model-agnostic methods for interpretability of neural network predictions: Application to a power plant, Comput. Chem. Eng. (108306) (2023) 108306.

[38] A. Holzinger, A. Saranti, C. Molnar, P. Biecek, W. Samek, Explainable AI methods - a brief overview, in: xxAI - Beyond Explainable AI, in: Lecture Notes in Computer Science, Springer International Publishing, Cham, 2022, pp. 13–38.

[39] P. Linardatos, V. Papastefanopoulos, S. Kotsiantis, Explainable AI: A review of machine learning interpretability methods, Entropy (Basel) 23 (1) (2020) 18.

[40] K. Simonyan, A. Vedaldi, A. Zisserman, Deep inside convolutional networks: Visualising image classification models and saliency maps, 2013, arXiv.

[41] D. Erhan, Y. Bengio, A. Courville, P. Vincent, Visualizing higher-layer features of a deep network, Tech. Rep. Univ. Montr. (2009).

[42] M. Sundararajan, A. Taly, Q. Yan, Axiomatic attribution for deep networks, 2017, arXiv arXiv:1703.01365.

[43] D. Smilkov, N. Thorat, B. Kim, F. Viégas, M. Wattenberg, SmoothGrad: removing noise by adding noise, 2017, arXiv.

[44] H. Felzmann, E. Fosch-Villaronga, C. Lutz, A. Tamò-Larrieux, Towards transparency by design for artificial intelligence, Sci. Eng. Ethics 26 (6) (2020) 3333–3361.

[45] K. Kumari, S. Yadav, Linear regression analysis study, J. Pract. Cardiovasc. Sci. 4 (1) (2018) 33.

[46] C. Yu, W. Yao, Robust linear regression: A review and comparison, Comm. Statist. Simulation Comput. 46 (8) (2017) 6261–6282, http://dx.doi.org/10.1080/03610918.2016.1202271.

[47] N. Altman, M. Krzywinski, Simple linear regression, Nat. Methods 12 (11) (2015) 999–1000.

[48] M. Saarela, S. Jauhiainen, Comparison of feature importance measures as explanations for classification models, SN Appl. Sci. 3 (2) (2021).

[49] M. Krzywinski, N. Altman, Classification and regression trees, Nat. Methods 14 (8) (2017) 757–758.

[50] L. Rutkowski, M. Jaworski, L. Pietruczuk, P. Duda, Decision trees for mining data streams based on the Gaussian approximation, IEEE Trans. Knowl. Data Eng. 26 (1) (2014) 108–119.

[51] A.K.H. Tung, Rule-based classification, in: Encyclopedia of Database Systems, Springer US, Boston, MA, 2009, pp. 2459–2462.

[52] A.C. Lumadjeng, T. Röber, M.H. Akyüz, Ş.İ. Birbil, Rule generation for classification: Scalability, interpretability, and fairness, 2021, arXiv.

[53] H. Lakkaraju, S.H. Bach, J. Leskovec, Interpretable decision sets: A joint framework for description and prediction, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, Association for Computing Machinery, New York, NY, USA, 2016, pp. 1675–1684, http://dx.doi.org/10.1145/2939672.2939874.

[54] A. Narayanan, K.J. Bergen, Prototype-based methods in explainable AI and emerging opportunities in the geosciences, 2024, arXiv arXiv:2410.19856.

[55] L. Zhang, L. Nelson, T.L. Griffiths, Analyzing the benefits of prototypes for semi-supervised category learning, 2024, arXiv arXiv:2406.02268.

[56] M. Biehl, Biomedical applications of prototype based classifiers and relevance learning, in: Algorithms for Computational Biology, in: Lecture Notes in Computer Science, Springer International Publishing, Cham, 2017, pp. 3–23.

[57] M.A. Choukali, M.C. Amirani, M. Valizadeh, A. Abbasi, M. Komeili, Pseudo-class part prototype networks for interpretable breast cancer classification, Sci. Rep. 14 (1) (2024) 10341.

[58] Z.C. Lipton, The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery., Queue 16 (3) (2018) 31–57, http://dx.doi.org/10.1145/3236386.3241340.

[59] A.J. Barnett, F.R. Schwartz, C. Tao, C. Chen, Y. Ren, J.Y. Lo, C. Rudin, A case-based interpretable deep learning model for classification of mass lesions in digital mammography, Nat. Mach. Intell. 3 (12) (2021) 1061–1070.

[60] A.J. Barnett, F.R. Schwartz, C. Tao, C. Chen, Y. Ren, J.Y. Lo, C. Rudin, Interpretable mammographic image classification using case-based reasoning and deep learning, 2021, arXiv [Cs.LG], arXiv:2107.05605.

[61] G. Carloni, A. Berti, C. Iacconi, M.A. Pascali, S. Colantonio, On the applicability of prototypical part learning in medical images: Breast masses classification using ProtoPNet, in: J.-J. Rousseau, B. Kapralos (Eds.), Pattern Recognition, Computer Vision, and Image Processing. ICPR 2022 International Workshops and Challenges, Springer Nature Switzerland, Cham, 2023, pp. 539–557.

[62] G. Singh, K.C. Yow, These do not look like those: An interpretable deep learning model for image recognition, IEEE Access 9 (2021) 41482–41493, http://dx.doi.org/10.1109/ACCESS.2021.3064838.

[63] D. Rymarczyk, A. Pardyl, J. Kraus, A. Kaczyńska, M. Skomorowski, B. Zieliński, ProtoMIL: Multiple instance learning with prototypical parts for whole-slide image classification, in: M.R. Amini, S. Canu, A. Fischer, T. Guns, P. Kralj Novak, G. Tsoumakas (Eds.), Machine Learning and Knowledge Discovery in Databases, Springer International Publishing, Cham, 2023, pp. 421–436.

[64] A. Bontempelli, S. Teso, K. Tentori, F. Giunchiglia, A. Passerini, Concept-level debugging of part-prototype networks, 2022, arXiv [Cs.LG], arXiv:2205.15769.

[65] J.K. Winkler, C. Fink, F. Toberer, A. Enk, T. Deinlein, R. Hofmann-Wellenhof, L. Thomas, A. Lallas, A. Blum, W. Stolz, H.A. Haenssle, Association between surgical skin markings in dermoscopic images and diagnostic performance of a deep learning convolutional neural network for melanoma recognition, JAMA Dermatol. 155 (10) (2019) 1135–1141, http://dx.doi.org/10.1001/jamadermatol.2019.1735.

[66] M. Liu, Y. Ning, Y. Ke, Y. Shang, B. Chakraborty, M.E.H. Ong, R. Vaughan, N. Liu, FAIM: Fairness-aware interpretable modeling for trustworthy machine learning in healthcare, Patterns 5 (10) (2024) 101059, http://dx.doi.org/10.1016/j.patter.2024.101059.

[67] H. Hakkoum, I. Abnane, A. Idri, Interpretability in the medical field: A systematic mapping and review study, Appl. Soft Comput. 117 (2022) 108391, http://dx.doi.org/10.1016/j.asoc.2021.108391.

[68] L. Shen, X. Feng, L. Xu, W. Ding, Adaptive prototype few-shot image classification method based on feature pyramid, PeerJ Comput. Sci. 10 (2024) e2322.

[69] G. Cheng, C. Lang, J. Han, Holistic prototype activation for few-shot segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 45 (4) (2023) 4650–4666, http://dx.doi.org/10.1109/TPAMI.2022.3193587.

[70] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS '17, Curran Associates Inc., Red Hook, NY, USA, 2017, pp. 4080–4090.

[71] O. Li, H. Liu, C. Chen, C. Rudin, Deep learning for case-based reasoning through prototypes: a neural network that explains its predictions, in: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI'18/IAAI'18/EAAI'18, AAAI Press, 2018, pp. 3530–3537.

[72] C. Chen, O. Li, C. Tao, A.J. Barnett, J. Su, C. Rudin, This looks like that: Deep learning for interpretable image recognition, 2018, arXiv [Cs.LG].

[73] E. Kim, S. Kim, M. Seo, S. Yoon, XProtoNet: Diagnosis in chest radiography with global and local explanations, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2021, pp. 15714–15723, http://dx.doi.org/10.1109/CVPR46437.2021.01546.

[74] C. Wang, Y. Chen, Y. Liu, Y. Tian, F. Liu, D.J. McCarthy, M. Elliott, H. Frazer, G. Carneiro, Knowledge distillation to ensemble global and interpretable prototype-based mammogram classification models, in: L. Wang, Q. Dou, P.T. Fletcher, S. Speidel, S. Li (Eds.), Medical Image Computing and Computer Assisted Intervention, MICCAI 2022, Springer Nature Switzerland, Cham, 2022, pp. 14–24.

[75] M. Nauta, J. Schlötterer, M. van Keulen, C. Seifert, PIP-Net: Patch-based intuitive prototypes for interpretable image classification, in: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2023, pp. 2744–2753, http://dx.doi.org/10.1109/CVPR52729.2023.00269.

[76] P. Angelov, E. Soares, Towards explainable deep neural networks (xDNN), Neural Netw. 130 (2020) 185–194.

[77] O. Li, H. Liu, C. Chen, C. Rudin, Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions, Proc. Conf. AAAI Artif. Intell. 32 (1) (2018).

[78] P.P. Angelov, X. Gu, Empirical Approach to Machine Learning, first ed., in: Studies in Computational Intelligence, Springer Nature, Cham, Switzerland, 2018.

[79] D. Hong, T. Wang, S. Baek, ProtoryNet - interpretable text classification via prototype trajectories, J. Mach. Learn. Res. 24 (264) (2023) 1–39.

[80] K.Q. Weinberger, L.K. Saul, Distance metric learning for large margin nearest neighbor classification, J. Mach. Learn. Res. 10 (2009) 207–244.

[81] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, D. Krishnan, Supervised contrastive learning, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20, Curran Associates Inc., Red Hook, NY, USA, 2020, pp. 18661–18673.

[82] S. Joe, B. Kim, H. Kang, K. Park, B. Kim, J. Park, J. Lee, Y. Gwon, ContraCluster: Learning to classify without labels by contrastive self-supervision and prototype-based semi-supervision, in: 2022 26th International Conference on Pattern Recognition, ICPR, 2022, pp. 4685–4692, URL: https://api.semanticscholar.org/CorpusID:254102810.

[83] G. Chechik, V. Sharma, U. Shalit, S. Bengio, Large scale online learning of image similarity through ranking, in: Pattern Recognition and Image Analysis, in: Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 11–14.

[84] P. Angelov, E. Soares, Explainable-by-design approach for covid-19 classification via ct-scan, 2020, bioRxiv (Accessed 21 June 2023).

[85] M.I. Malinen, P. Fränti, Balanced K-means for clustering, in: Lecture Notes in Computer Science, in: Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 32–41.

[86] W. Tang, Y. Yang, L. Zeng, Y. Zhan, Optimizing MSE for clustering with balanced size constraints, Symmetry (Basel) 11 (3) (2019) 338.

[87] M. Xu, P. Fränti, Delta-MSE dissimilarity in suboptimal K-means clustering, in: 17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, August 23–26, 2004, IEEE Computer Society, 2004, pp. 577–580, http://dx.doi.org/10.1109/ICPR.2004.1333838.

[88] R. Martín-Santamaría, J. Sánchez-Oro, S. Pérez-Peló, A. Duarte, Strategic oscillation for the balanced minimum sum-of-squares clustering problem, Inf. Sci. (NY) 585 (2022) 529–542.

[89] N. Zhang, Z. Yu, Q. Wu, Overlapping sliced inverse regression for dimension reduction, Anal. Appl. 17 (05) (2019) 715–736.

[90] A.M. Samarov, Exploring regression structure using nonparametric functional estimation, J. Amer. Statist. Assoc. 88 (423) (1993) 836.

[91] K.C. Li, On principal hessian directions for data visualization and dimension reduction: Another application of stein's lemma, J. Amer. Statist. Assoc. 87 (420) (1992) 1025.

[92] R.D. Cook, X. Yin, Theory & methods: Special invited paper: Dimension reduction and visualization in discriminant analysis (with discussion), Aust. N. Z. J. Stat. 43 (2) (2001) 147–199.

[93] K. Fukumizu, F.R. Bach, M.I. Jordan, Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces, J. Mach. Learn. Res. 5 (2004) 73–99.

[94] M. Hristache, A. Juditsky, J. Polzehl, V. Spokoiny, Structure adaptive approach for dimension reduction, Ann. Stat. 29 (6) (2001) 1537–1566.

[95] L. Chen, Q. Guo, Z. Liu, S. Zhang, H. Zhang, Enhanced synchronization-inspired clustering for high-dimensional data, Complex Intell. Syst. 7 (1) (2021) 203–223.

[96] E. Barshan, A. Ghodsi, Z. Azimifar, M. Zolghadri Jahromi, Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds, Pattern Recognit. 44 (7) (2011) 1357–1371.

[97] C.S. Mukherjee, J. Zhang, Compressibility: Power of PCA in clustering problems beyond dimensionality reduction, 2022, arXiv.

[98] S.C. Chung, S.H. Wang, P.Y. Niu, S.Y. Huang, W.H. Chang, I.P. Tu, Two-stage dimension reduction for noisy high-dimensional images and application to Cryogenic Electron Microscopy, Ann. Math. Sci. Appl. Vol. 5, Number 2. (2020) 283–316.

[99] L. Zheng, R. Lukac, X. Wu, D. Zhang, PCA-based spatially adaptive denoising of CFA images for single-sensor digital cameras, IEEE Trans. Image Process. 18 (4) (2009) 797–812.

[100] X. Wang, X. Liu, Enhancing robustness of classifiers based on PCA, in: 2021 4th International Conference on Pattern Recognition and Artificial Intelligence, PRAI, 2021, pp. 336–341, http://dx.doi.org/10.1109/PRAI53619.2021.9550807.

[101] X. Yang, X. He, J. Zhao, Y. Zhang, S. Zhang, P. Xie, COVID-CT-Dataset: A CT scan dataset about COVID-19, 2020, CoRR arXiv:2003.10769.

[102] A.M. Tahir, M.E.H. Chowdhury, A. Khandakar, T. Rahman, Y. Qiblawey, U. Khurshid, S. Kiranyaz, N. Ibtehaz, M.S. Rahman, S. Al-Maadeed, S. Mahmud, M. Ezeddin, K. Hameed, T. Hamid, COVID-19 infection localization and severity grading from chest X-ray images, Comput. Biol. Med. 139 (105002) (2021) 105002.

[103] R.S. Lee, F. Gimenez, A. Hoogi, K.K. Miyake, M. Gorovoy, D.L. Rubin, A curated mammography data set for use in computer-aided detection and diagnosis research, Sci. Data 4 (1) (2017) 170177.

[104] I.C. Moreira, I. Amaral, I. Domingues, A. Cardoso, M.J. Cardoso, J.S. Cardoso, INbreast: Toward a full-field digital mammographic database, Academic Radiol. 19 (2) (2012) 236–248, http://dx.doi.org/10.1016/j.acra.2011.09.014.

[105] H. Zhang, R. Wu, T. Yuan, Z. Jiang, S. Huang, J. Wu, J. Hua, Z. Niu, D. Ji, DE-Ada*: A novel model for breast mass classification using cross-modal pathological semantic mining and organic integration of multi-feature fusions, Inform. Sci. 539 (2020) 461–486, http://dx.doi.org/10.1016/j.ins.2020.05.080.

[106] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, L.F. Fei, ImageNet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255, http://dx.doi.org/10.1109/CVPR.2009.5206848.

[107] PyTorch, torch.optim.Adam, 2023, https://pytorch.org/docs/stable/generated/torch.optim.Adam.html. (Accessed 21 August 2024).

[108] E. Soares, P. Angelov, S. Biaso, M.H. Froes, D.K. Abe, SARS-CoV-2 CT-scan dataset: A large dataset of real patients CT scans for SARS-CoV-2 identification, 2020, http://dx.doi.org/10.1101/2020.04.24.20078584, medRxiv.

[109] L. Tsochatzidis, L. Costaridou, I. Pratikakis, Deep learning for breast cancer diagnosis from mammograms—A comparative study, J. Imaging 5 (3) (2019) http://dx.doi.org/10.3390/jimaging5030037.

[110] H. Li, D. Chen, W.H. Nailon, M.E. Davies, D.I. Laurenson, Dual convolutional neural networks for breast mass segmentation and diagnosis in mammography, IEEE Trans. Med. Imaging 41 (1) (2022) 3–13, http://dx.doi.org/10.1109/TMI.2021.3102622.

[111] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 2623–2631, http://dx.doi.org/10.1145/3292500.3330701.