An Interpretable Prototype Parts-based Neural Network for Medical Tabular Data

Jacek Karolczak[1,*], Jerzy Stefanowski[1]

[1]Poznan University of Technology, Institute of Computing Science, ul. Piotrowo 2, 60-695 Poznań, Poland

Abstract

The ability to interpret machine learning model decisions is critical in such domains as healthcare, where trust in model predictions is as important as their accuracy. Inspired by the development of prototype parts-based Deep Neural Networks in computer vision, we propose a new model for tabular data, specifically tailored to medical records, that requires discretization of diagnostic result norms. Unlike the original vision models that rely on the spatial structure, our method employs trainable patching over features describing a patient, to learn meaningful prototypical parts from structured data. These parts are represented as binary or discretized feature subsets. This allows the model to express prototypes in human-readable terms, enabling alignment with clinical language and case-based reasoning. Our proposed neural network is inherently interpretable and offers interpretable concept-based predictions by comparing the patient's description to learned prototypes in the latent space of the network. In experiments, we demonstrate that the model achieves classification performance competitive to widely used baseline models on medical benchmark datasets, while also offering transparency, bridging the gap between predictive performance and interpretability in clinical decision support.

Keywords

Interpretable Machine Learning, Prototype Learning, Case-Based Reasoning, Learnable Discretization, Tabular Data

1. Introduction

Machine learning (ML) has been increasingly used in medicine for many decades, in particular to improve diagnostic accuracy, predict patient outcomes, and support clinical decision making by uncovering complex patterns in medical data [1]. Early applications of machine learning prioritized inherently interpretable models that provided symbolic knowledge representations, such as *Decision Trees* (DT) and rule-based systems [2]. Encouraged by the initial successes of these approaches, researchers began addressing more complex problems using more advanced models such as *random forests* (RF), other ensembles, or even hybrid approaches [3]. Although these ML systems offer an improvement in predictive performance [4], they do so at the expense of transparency and interpretability [5].

Nowadays, in many tasks, *Deep Neural Networks* (DNN) have become the most popular approach, particularly for analyzing modalities such as images, time series, or text data. However, a significant portion of clinical work still relies on tabular data, where the application of deep learning models, due to their black-box nature, is less widespread and less appreciated. In the healthcare domain, the reluctance to adopt DNN is partially driven by the difficulty in interpreting their decision-making processes, making it challenging for physicians to analyze, validate, and ultimately trust their results in real-world applications. As a result, there has been growing interest in using *Explainable AI* (XAI) techniques [6] to make machine learning models more transparent and understandable for clinical use.

Currently, the landscape of XAI is dominated by feature importance methods such as SHAP [7] and LIME [8] being among the most widely used. However, despite their popularity, these approaches often produce abstract and incomprehensible explanations, even for machine learning experts, and can be particularly challenging for physicians to understand [9]. As a result, there is growing interest in

EXPLIMED 2025 - Second Workshop on Explainable Artificial Intelligence for the Medical Domain - 25–30 October 2025, Bologna, Italy

*Corresponding author.

jacek.karolczak@cs.put.poznan.pl (J. Karolczak); jerzy.stefanowski@cs.put.poznan.pl (J. Stefanowski)

0000-0001-5414-960X (J. Karolczak); 0000-0002-4949-8271 (J. Stefanowski)

alternative paradigms that provide more intuitive and human-understandable insights aligned with the way physicians reason about the patient and the diagnosis.

In this context, *prototypes* – instances that represent groups of similar examples – have emerged as a particularly promising explanation technique [10]. Since they correspond directly to input data, they align more naturally with human reasoning processes and are generally easier to interpret, including for medical professionals without specialized training in machine learning [11]. Prototypes can serve as both *local explanations* by showing cases similar to the predicted instance and as *global explanations* by presenting representative examples from the data. This makes them a powerful tool for understanding both individual predictions and overall model reasoning.

Inspired by the paper [12] on the *prototypical part-based network* for image classification, where predictions are explained through interpretable patches rather than complete images, we explore how similar principles can be adapted for tabular medical data. Despite the success of [12] in other domains, prototype networks for tabular data remain underexplored, particularly in healthcare. This is notable because medical data often use discrete range language rather than raw features values. Discretized variables are easier to interpret because they correspond to clear, meaningful categories, such as age ranges, test result groups, or risk levels. These discrete features align better with clinical reasoning and allow for more transparent decision-making. Using these features, models can offer more intuitive explanations, helping physicians better understand the predictions and relate them to real-world clinical scenarios.

To address this gap, we propose a prototype-based neural network, called *Model for Explainable Diagnosis using Interpretable Concepts* (MEDIC), specifically designed for tabular medical data. Our approach introduces discrete prototypes, with the aim of improving interpretability while maintaining strong predictive performance. While traditional models such as DTs offer symbolic interpretability, their reliance on rigid rule structures may not align well with the complexity of clinical reasoning. In contrast, our method adopts a prototype-based approach that enables more flexible, example-driven explanations, allowing clinicians to interpret decisions through similarities to real, representative patient cases. The goal of this study is to develop and evaluate this model in the context of medical records of patients, with a focus on producing faithful and physician-friendly explanations.

To ensure reproducibility, the code is publicly available in a GitHub repository[1].

2. Related Work

The work [1] claims that Deep Neural Networks [13] do not provide significant performance advantages over classical approaches such as random forest [3] or gradient boosting (GB) [14] for clinical prediction tasks utilizing tabular data, which may explain their limited adoption in the healthcare domain [1].

Despite machine learning models consistently demonstrating superior performance with tabular clinical data, these ensemble methods suffer from inherent opacity in their decision processes, creating a critical need for effective explanation frameworks that can provide healthcare professionals with transparent insights into model reasoning [15].

The landscape of explainable AI approaches can be broadly categorized into two paradigms: *feature attribution methods* and *concept-based methods* [9]. The first group was briefly discussed in Section 1. The second category includes *prototype-based explanations* (also called example-based or instance-based explanations [6]), has shown particular promise in aligning with human cognitive processes, especially in domains where case-based reasoning is predominant. This is particularly true in the medical domain, where such methods have been shown to be effective in improving interpretability and trust [5].

Prototype-based explanations generally fall into two families. The post-hoc family identifies prototypes after model training, typically selecting representative instances from the training set. Notable algorithms include MMD-Critic [16], which employs maximum mean discrepancy to select prototypes and criticisms, and optimization-based approaches like A-PETE [17] and IKNN_PSLFW [18]. Although straightforward to apply to tabular medical data, these methods often struggle with high-dimensional

[1] https://github.com/jkarolczak/medic

datasets containing many irrelevant features, which is a common characteristic in healthcare, where comprehensive diagnostic panels frequently generate information records containing redundant information [19]. In this context, it is important to guide the decision maker's attention toward the specific features of the prototype that the model considers relevant [20].

The second family, ante-hoc or intrinsic prototype methods, integrates prototype reasoning directly into the model architecture. Usually, these approaches represent prototypes not as complete instances but as parts or feature conjunctions that participate in decision making through mechanisms such as weighted voting. This direction gained significant attention following [21], where the approach was originally proposed for image classification.

ProtoPNet [12] represents a breakthrough in this area, introducing a convolutional neural architecture where class predictions are based on similarities of the learned prototypical parts of images. The key innovation of ProtoPNet was enabling interpretability through visualization of prototypical image patches that the model "looks for" when making classifications. When classifying a new image, ProtoPNet identifies similar-looking patches in the input and compares them to its learned prototypes, with the similarity scores directly contributing to class predictions. This approach is particularly powerful for medical imaging, where specific visual patterns (such as tumors or lesions) are diagnostically significant. As documented in [22] and demonstrated in applications like [23], such models enhance transparency by highlighting medically relevant image regions and explicitly connecting them to learned prototypes that represent typical visual manifestations of conditions.

However, despite ProtoPNet's successful application across various image processing tasks [22], adapting this architecture for tabular data presents unique challenges. Medical tabular data lack the spatial structure of images that convolutional networks exploit, requiring fundamentally different approaches to identify meaningful "parts" or feature conjunctions. To date, a comparable architecture specifically designed for tabular medical records remains conspicuously absent from the literature.

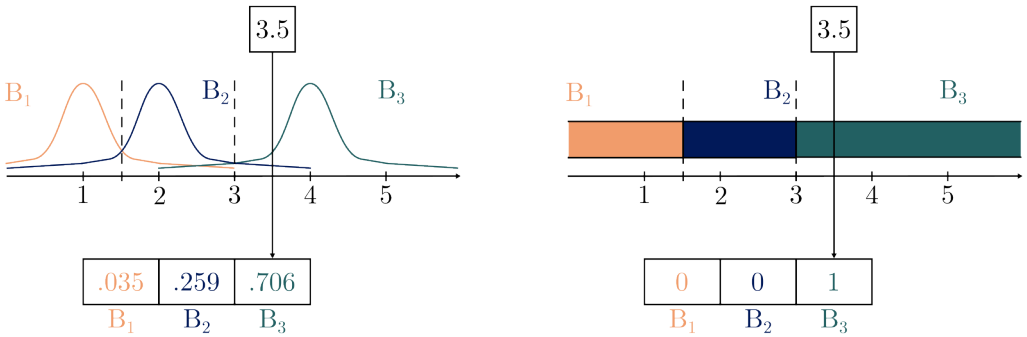
3. MEDIC: Model for Explainable Diagnosis using Interpretable Concepts

In this section, for the first time, we propose the neural network *MEDIC: Model for Explainable Diagnosis using Interpretable Concepts*. MEDIC is inspired by the prototypical parts paradigm proposed in [12] and is designed to produce accurate and inherently interpretable predictions, which makes it particularly well suited for medical decision support, which usually requires human interpretation of proposed results. The model decomposes the decision support process into a small number of meaningful, human-understandable components: discretized input features describing the patient, interpretable feature subsets (parts) and class prototypes grounded in real data. These elements enable case-based reasoning and transparent justification of the proposed classification.

At a high level, the architecture follows an interpretable processing pipeline consisting of four key stages: (1) input discretization, which transforms continuous variables in the patient's description into symbolic bins; (2) part extraction, which identifies sparse and semantically coherent subsets of input features; (3) prototype comparison, where each extracted part of the patient's description is matched to learned prototypes, stored as embeddings representing features subsets of feature-value pairs from the training data; and (4) classification of the considered instance based on its similarity to prototypes. The complete MEDIC model is trained end-to-end to jointly learn all of these components in a supervised setting.

In clinical data, where features often come from heterogeneous sources (e.g. lab tests values, vital signs, diagnoses), such structured reasoning aligns well with domain expert expectations. Discretized bins can reflect clinically relevant ranges of diagnostic tests (e.g., abnormally high glucose), sparse parts mirror combinations that physicians would consider jointly (e.g., elevated CRP and fever), and prototypes anchor predictions in real cases that can be inspected post hoc.

We now describe the architecture in detail, starting with describing the interpretable discretization of continuous input features.



(a) **Fuzzy binning**: the input value is softly assigned to each bin based on proximity to bin's center. The final encoding is a weighted combination, where the weights reflect similarity to the bin centers.

(b) **Hard binning**: the input is deterministically assigned to a single bin with the nearest center. The encoding becomes a one-hot vector.

Figure 1: Comparison between fuzzy and hard binning for a scalar input value.

3.1. Interpretable Discretization of Continuous Input Features

The decretization of continuous medical variables (e.g. age, lab tests values) into symbolic categories can aid interpretation and facilitate reasoning about patient features. In this work, our aim is to ultimately produce ranges of continuous features for symbolic interpretability. However, such a hard discretization is hardly optimizable in gradient-based neural network training.

To overcome this challenge, we introduce a fuzzy binning layer that enables a smooth, differentiable approximation of hard discretization during training. This allows gradients to flow through the discretization process and enables end-to-end optimization. After training, the soft representation can be replaced with a deterministic hard binning for better interpretability.

Fuzzy Binning To allow interpretable discretization of continuous input features, we introduce a fuzzy binning layer that softly assigns each scalar feature value $x \in \mathbb{R}$ to a set of K trainable bins. Each bin is characterized by a learnable center $\mu_k \in \mathbb{R}$ and a shared bandwidth parameter $\sigma > 0$. The soft membership of x to the bin k is defined using a Gaussian kernel:

$$d_k(x) = \frac{(x - \mu_k)^2}{2\sigma^2} \tag{1}$$

$$\tilde{b}_k(x) = \frac{\exp(-d_k(x))}{\sum_{j=1}^{K} \exp(-d_j(x)) + \varepsilon} \tag{2}$$

where $\tilde{b}_k(x)$ denotes the normalized soft assignment and ε is a small constant added for numerical stability. This results in a fuzzy, probabilistically weighted representation over bins, allowing each input to contribute partially to multiple bins (see Figure 1a).

The use of Gaussian kernels for fuzzy binning offers several advantages over direct distance-based assignment (e.g., L^2 norm). First, the smooth exponential decay naturally reflects uncertainty in proximity, which is especially relevant when feature values lie near bin boundaries. Second, the resulting softmax distribution is differentiable and normalized, facilitating gradient-based optimization in Deep Neural Networks.

Importantly, the bin centers μ_k and shared bandwidth σ are optimized jointly with other model parameters during end-to-end training, allowing the discretization scheme to adapt to the data distribution.

Hard Binning After initial training of the network, the discretization is switched to the hard mode. In the hard setup, the input is assigned to a single bin via a non-differentiable $\arg\min$ operation over squared distances:

$$\hat{b}(x) = \text{one\_hot}\left(\arg\min_k (x - \mu_k)^2\right) \tag{3}$$

The resulting representation is a one-hot[2] vector (Figure 1b), which can be advantageous for symbolic interpretation and comparison of prototypes. However, it lacks gradient flow, making it unsuitable for end-to-end training.

In the hard binning regime, the input feature values are partitioned into contiguous intervals derived from the learned bin centers $\{\mu_k\}_{k=1}^{K}$. Specifically, each bin k is associated with the interval

$$I_k = \begin{cases} \left(-\infty, \frac{\mu_1+\mu_2}{2}\right) & \text{if } k = 1 \\ \left[\frac{\mu_{k-1}+\mu_k}{2}, \frac{\mu_k+\mu_{k+1}}{2}\right) & \text{if } 1 < k < K \\ \left[\frac{\mu_{K-1}+\mu_K}{2}, +\infty\right) & \text{if } k = K \end{cases} \tag{4}$$

such that any scalar input x is discretized into the bin k for which $x \in I_k$. This alternative representation facilitates interpretation by decision makers.

3.2. MEDIC Architecture

MEDIC is a neural network inspired by the interpretable prototypical parts-based classification paradigm proposed in [12] and integrates symbolic input binning, feature extraction, prototype learning, and class prediction based on association with learned prototypes. The overview of the entire MEDIC architecture is shown in Figure 2.

In the beginning, the raw input vector of features describing the patient is transformed by the network into a sparse high-dimensional binary representation. Each continuous feature is processed by a binning module introduced in Section 3.1. Meanwhile, categorical features undergo one-hot[2] encoding. All vectors coming from discretization are concatenated into a single d' dimensional vector.

Next, the binarized input is multiplied with a trainable set of p patching masks, encoded as a matrix $M \in \mathbb{R}^{p \times d'}$. Each mask selects and linearly combines a sparse subset of binary features, effectively defining a part of the input instance (i.e. its description by features). Intuitively, each part can be seen as a meaningful combination of clinical indicators – for example, *high blood pressure in elderly patients* or *elevated glucose and BMI*. This encourages the model to focus on patterns that are not only interpretable but also structured in a way that reflects domain knowledge.

Each of the p part vectors is passed through a shared feature extractor module, implemented as a shallow multilayer perceptron with ReLU activations. This module transforms each sparse binary part into a dense embedding – a compact vector of size h that captures abstract and informative features. Embeddings are designed to preserve meaningful relationships in the data while reducing dimensionality, enabling the model to generalize across similar patterns. From an interpretability perspective, this step summarizes each clinically relevant pattern into a low-dimensional representation that retains the most diagnostically informative aspects.

To facilitate interpretable decision-making, the network maintains a set of n learnable prototype vectors of size h. For each input part represented as embedding, the L^2 distance is computed for every prototype. This results in a $p \times n$ distance matrix, where each entry quantifies the dissimilarity between a specific part and a prototype. A max-pooling operation across parts selects the most relevant part for each prototype, yielding a vector of n minimal distances, where each entry reflects the smallest distance between a given prototype and the most similar embedding representing a part of the input

[2]A one-hot vector is a way of representing categories or intervals where only one entry is "on" (set to 1) and all others are "off" (set to 0), see 1b. For example, if a lab result like albumin is split into three ranges (low, normal, high), only one of these will be marked as active. This makes it easy to interpret into which clinical range the value falls.

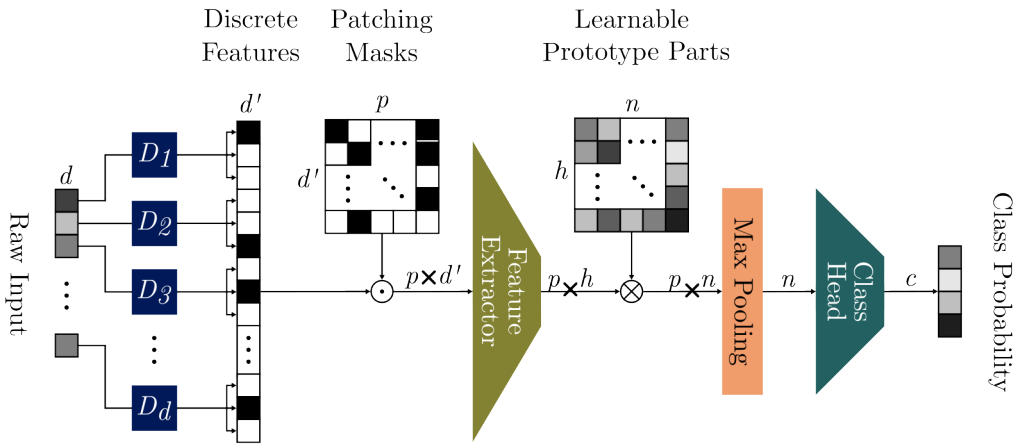


Figure 2: Overview of the Model for Explainable Diagnosis using Interpretable Concepts (MEDIC) architecture. Input features describing the patient are first discretized into binary features. These are projected using Hadamard product ($\odot$) into a set of p interpretable input parts using sparse patching masks. Parts are encoded into an embedded space, compared to n learnable prototypes using L^2 distance ($\otimes$), and the resulting distances are pooled and passed through a classification head to class assignment probabilities.

describing the patient. This enables comparison of each prototype to its best-matching part in the patient description.

To enable case-based reasoning, the network maintains a set of n learnable prototype vectors, each of dimension h. Conceptually, each prototype represents a summary of a typical clinical condition or patient case learned from data. Each prototype is anchored in real patient data and corresponds to a representative example that lies near the center of a cluster of similar cases, making it reflective of common patterns observed across many patients. For every embedding corresponding to the part of patient description, the model computes the squared Euclidean (L^2) distance to each prototype, yielding a $p \times n$ distance matrix. Each row corresponds to one patient description part and each column to a prototype.

Then a maximum-pooling operation is applied across the rows of this matrix (that is, across parts), selecting for each prototype, the input part that has the smallest distance to that prototype, effectively identifying the most similar part. This produces a distance vector of length n, which summarizes how closely the input aligns with each of the learned prototypes.

Finally, this vector of distances is passed through a linear classification layer, producing a probability distribution over c target classes. Since the classification decision is based directly on similarity to interpretable prototypes, each linked to specific input parts, the resulting predictions can be traced back and explained in terms of clinically meaningful comparisons to learned prototype parts.

3.3. Three-Stage Training Procedure

To ensure stable and interpretable network training, we adopt a three-stage training procedure that enables the model to learn hard bins – intervals, and realistic prototype parts directly from the training data, all within a gradient-based optimization framework.

Stage 1: Initialization with Fuzzy Binning and Learnable Prototypes In the initial stage, the entire network is trained end-to-end with fuzzy binning and randomly initialized learnable prototypes. This setting ensures smooth gradient flow through the discretization modules, allowing the network to co-adapt binning thresholds and part extraction masks.

Fuzzy binning uses soft Gaussian kernels (Section 3.1, Figure 1a), which provide fuzzy assignments across bins. The patching masks and prototypes are trained jointly using classification loss, combined

with auxiliary regularization terms: (1) L1 sparsity of patching masks, and (2) a diversity penalty to encourage spread among prototypes, which are further discussed in Section 3.4.

Stage 2: Hard Binning and Mask Discretization Once convergence in the training criterion is achieved, the discretization mode is switched to a hard mode by replacing the fuzzy binning with hard $\arg\min$ bin selection, freezing binning thresholds (Section 3.1, Figure 1b). Additionally, patching masks are binarized by thresholding to enforce strict binary groupings of input dimensions into parts.

This transition enables symbolic interpretability and highlights which specific input features are most relevant for each part. The rest of the network is fine-tuned using discretized inputs, preserving the interpretability of the parts.

Stage 3: Prototype Replacement with Real Parts Finally, the learned prototypes are replaced with embeddings derived from parts of actual patient records in the training data, ensuring that each prototype corresponds to a real and representative clinical case. For each prototype, the closest embedded input part is identified using the L^2 distance. These real parts are then copied into the prototype memory, replacing the synthetic prototypes. This step improves interpretability by anchoring each prototype to an actual example from the data.

This last step grounds the network's reasoning in actual data, allowing domain experts to inspect prototypical cases for each class. During this phase, the prototype embeddings are frozen, and only the classification head is fine-tuned to maintain stable performance, as accuracy would otherwise be expected to decline.

Model and training complexity From a computational standpoint, MEDIC maintains a relatively simple neural network architecture, comparable in training complexity to a shallow multi-layer perceptron with five layers. The operations within the network consist primarily of matrix multiplications and element-wise products, which are fully parallelizable on modern hardware and avoid sequential dependencies inherent in architectures such as Recurrent Neural Networks. The final stage of prototype selection, which matches learned embeddings to parts of real patient records, scales linearly with the size of the dataset, making it efficient even for larger collections. As a result, both training and inference remain computationally lightweight, with performance characteristics similar to other small feed-forward neural networks.

3.4. Objective Function and Regularization

The model is trained using cross-entropy loss, later denoted as $\mathcal{L}_{CE}$, as the standard objective for classification tasks. To improve interpretability and promote efficient structure, two regularization terms are added. The first is an ℓ_1 sparsity penalty applied to the patching mask parameters:

$$\mathcal{L}_{\text{sparsity}} = \lambda_{\text{sparsity}} \cdot \ell_1(M_{ij}) = \lambda_{\text{sparsity}} \cdot \frac{1}{pd'} \sum_{i=1}^{p} \sum_{j=1}^{d'} |M_{ij}| \,, \tag{5}$$

where $M \in \mathbb{R}^{p \times d'}$ are the patching masks. This term encourages parts to rely on a minimal set of input features describing each patient. The ℓ_1 penalty is chosen because, unlike ℓ_2, it promotes exact zeros in patching masks, effectively turning off irrelevant input dimensions, and therefore leading to sparser and therefore more interpretable part-feature associations.

The second term encourages diversity among prototypes by penalizing redundancy in their representations:

$$\mathcal{L}_{\text{diversity}} = -\lambda_{\text{diversity}} \cdot \frac{1}{N(N-1)} \sum_{i \neq j} \|\mathbf{z}_i - \mathbf{z}_j\|_2 \,, \tag{6}$$

where $\mathbf{z}_i$ and $\mathbf{z}_j$ are prototype embeddings. This promotes coverage of distinct regions in the latent space. The full training objective function is the sum of three above-mentioned loss parts:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{sparsity}} + \mathcal{L}_{\text{diversity}} \,. \tag{7}$$

4. Experiments

This section presents a comprehensive evaluation of our model, both from an interpretability and a predictive performance perspective. In the beginning, in Section 4.1 we assess the predictive accuracy of the method on three benchmark datasets, comparing its performance to selected baseline models. Subsequently, in Section 4.2 we demonstrate how the learned prototypes may be applied in practice, through a case study grounded in a real-world medical dataset. This analysis highlights the interpretability of MEDIC and its ability to form clinically plausible representations.

4.1. Predictive performance

4.1.1. Experimental setup

Data To evaluate the proposed model, three publicly available medical datasets were selected: *Cirrhosis*[3], *Chronic Kidney Disease (CKD)*[4], and *Diabetes*[5]. These datasets were chosen due to their clinical relevance and inclusion of multiple laboratory measurements such as blood test results, namely:

- Cirrhosis: bilirubin, cholesterol, albumin, copper, triglycerides, alkaline phosphatase (ALP), serum glutamic-oxaloacetic transaminase (SGOT), platelets, and prothrombin time;
- CKD: red blood cells, pus cells, pus cell clumps, blood glucose, blood urea, serum creatinine, sodium, potassium, hemoglobin, packed cell volume, white blood cell count, and red blood cell count;
- Diabetes: blood glucose and insulin.

Enumerated tests are well suited for discretization. These datasets also include additional numerical indicators, such as body mass index (BMI, in the Diabetes dataset), which further benefit from discretization by enhancing interpretability. Moreover, all three datasets exhibit a class imbalance: Cirrhosis contains 125, 19, and 168 instances for classes *death*, *censored*, and *censored due to liver transplantation* respectively; CKD dataset consists of 115 and 43 instances for classes *not CKD* and *CKD*; and Diabetes includes 500 negative and 268 positive samples for diabetes presence. These characteristics present a realistic benchmark for evaluating the model's ability to process numerical medical features while addressing class imbalance, a common challenge in clinical predictive modeling [24].

Baselines To evaluate the effectiveness of our prototype-based method, we compare it with a set of well-established baseline models commonly used in clinical machine learning tasks. Ensemble methods such as *Random Forest (RF)* [3] and *Gradient Boosting*, specifically the *XGBoost (XGB)* implementation [14], serve as strong baselines due to their robustness, ability to capture non-linear feature interactions, and proven success in medical applications [4]. We include a *Decision Tree (DT)* model [13] as a reference for interpretability, since it represents models that are interpretable by design and is also a classifier that turned out to be sufficient to solve some problems [2]. Furthermore, we incorporate a simple feedforward neural network, also known as a *Multi-Layer Perceptron (MLP)* [13], to provide a baseline comparison within the class of neural models. The MLP consists of an input layer, one or more hidden layers with nonlinear activation functions, and an output layer with softmax activation for classification. The Decision Tree, Random Forest, and MLP, utilized implementations of the scikit-learn[6] Python package. The XGB implementation comes from the XGBoost[7] package.

Criterion To compare the performance of different models, we use the geometric mean (g-mean) of sensitivity and specificity, defined as:

[3] https://archive.ics.uci.edu/dataset/878/cirrhosis+patient+survival+prediction+dataset-1
[4] https://archive.ics.uci.edu/dataset/336/chronic+kidney+disease
[5] https://www.kaggle.com/datasets/mathchi/diabetes-data-set
[6] https://scikit-learn.org/
[7] https://xgboost.readthedocs.io/

$$\text{g-mean} = \sqrt{\text{sensitivity} \times \text{specificity}} \tag{8}$$

where sensitivity (also called recall) measures the proportion of actual positive cases correctly identified and specificity measures the proportion of actual negative cases correctly identified:

$$\text{sensitivity} = \frac{TP}{TP + FN}, \quad \text{specificity} = \frac{TN}{TN + FP} \tag{9}$$

The g-mean balances performance on both classes, ensuring the model is not biased toward the majority class. This is especially useful in medical datasets with class imbalance, where one outcome (e.g., disease presence) is much rarer. Unlike accuracy, g-mean provides a more balanced and clinically meaningful measure by ensuring good performance on both positive and negative classes [25].

Hyperparameter Optimization Hyperparameter optimization (HPO) is essential to achieve strong and unbiased performance between models, particularly in settings that involve heterogeneous architectures. For all evaluated models, we used the *Tree-structured Parzen Estimator Approach* (TPE) [26] implemented in the Optuna framework[8] to perform black-box optimization of key hyperparameters. Each model was independently tuned using 100 optimization trials to maximize the g-mean metric [13]. To ensure a reliable estimation of predictive performance on unseen data, we adopt a 5-fold cross-validation framework.

For MEDIC, we recommend a structured tuning strategy informed by prior experience. Begin by setting $\lambda_{\text{sparsity}}$ to a relatively high value (e.g. ≈ 1.0) and $\lambda_{\text{diversity}} = 0$, together with a large number of prototypes (e.g. 64). Gradually decrease $\lambda_{\text{sparsity}}$ until the number of activated features in the prototype parts stabilizes within a comprehensible range, ideally fewer than 5-7 features per prototype part. Once this is achieved, incrementally increase $\lambda_{\text{diversity}}$ to promote diversity inactivated parts, ensuring that the prototype part lengths remain consistent. After arriving at interpretable and stable prototype configurations, the remaining hyperparameters, including the number of prototypes (see Table 1) can be automatically tuned using a hyperparameter optimization algorithm such as TPE [26].

Table 1 summarizes the hyperparameters tuned for each model and the corresponding search spaces. For implementation-specific details, we refer the reader to the respective baseline packages cited in the **Baselines** paragraph above.

4.1.2. Results

The performance of the model is summarized in Table 2, using the geometric mean (g-mean) metric in three datasets. MEDIC demonstrates competitive performance, achieving the best g-mean on the Cirrhosis and CKD datasets. In the diabetes data set, although XGB achieved the highest score, MEDIC followed closely, within less than a percentage point, indicating comparable effectiveness.

Table 3 shows the maximum allowed number of prototypes defined as a model setting (hyperparameter) and the number of unique prototype parts actually discovered by the MEDIC model during training. The results suggest that the model can self-regularize by reusing the same prototype multiple times when no additional meaningful feature-value sets (prototype parts) can be identified. This indicates that MEDIC avoids overfitting by focusing only on truly informative patterns, even when more prototypes are allowed.

4.2. Studying MEDIC in Action

To show that MEDIC is interpretable and present how prototype parts learned by MEDIC look like, we conducted a qualitative analysis using a case study for a single dataset – Cirrhosis. Our aim is to illustrate how the MEDIC reasoning process works by comparing individual patient cases to representative clinical patterns (prototypes) previously learned from the data.

[8] https://optuna.org

Table 1
Hyperparameters and the ranges of values allowed for selection during hyperparameter optimization.

model	hyperparameter name	type	values
DT	ccp_alpha	float	$[1e^{-5}, 0.1]$
	criterion	categorical	{ gini, entropy, log_loss }
	max_depth	integer	$[3, 30]$
	max_features	categorical	{ sqrt, log2, None }
	min_samples_leaf	integer	$[1, 20]$
	min_samples_split	integer	$[2, 20]$
RF	max_depth	integer	$[3, 30]$
	max_features	categorical	{ sqrt, log2, None }
	min_samples_leaf	integer	$[1, 20]$
	min_samples_split	integer	$[2, 20]$
	n_estimators	integer	$[50, 300]$
XGB	alpha	float	$[1e^{-3}, 10]$
	colsample_bytree	float	$[0.5, 1]$
	gamma	float	$[0, 5]$
	lambda	float	$[1e^{-3}, 10]$
	learning_rate	float	$[1e^{-3}, 0.3]$
	max_depth	integer	$[3, 15]$
	min_child_weight	float	$[0.5, 10]$
	n_estimators	integer	$[50, 500]$
	subsample	float	$[0.5, 1]$
MLP	activation	categorical	{ relu, tanh, logistic }
	alpha	float	$[1e^{-5}, 0.1]$
	hidden_layer_sizes	categorical	{ 100,), (50, 50), (100, 50), (50, 25), (30, 30, 30) }
	learning_rate_init	float	$[1e^{-5}, 0.1]$
	max_iter	integer	$[100, 1000]$
MEDIC	batch_size	integer	{ 16, 32, 64, 80, ..., 240, 256 }
	hidden_dim	int	$[2, 16]$
	learning_rate	float	$[1e^{-5}, 0.1]$
	n_prototypes	integer	{ 4, 8, 12, 16, ..., 92, 96 }

Table 2
Classification performance (g-mean) across datasets. Bold values denote the best result for each dataset.

	Cirrhosis	CKD	Diabetes
DT	0.6564	0.9889	0.7327
RF	0.6742	**1.0000**	0.7432
XGB	0.6840	0.9889	**0.7453**
MLP	0.6765	**1.0000**	0.7381
MEDIC	**0.6889**	**1.0000**	0.7367

Table 4 shows the discretized feature intervals identified by the network. These intervals represent meaningful partitions of the input space and often align with known clinical thresholds. For reference, we compare them with the standard clinical intervals provided by the American College of Clinical Pharmacy[9].

For example, the learned limit between intervals 1 and 2 for albumin is 3.7 g/dL, which closely matches the clinical lower limit of 3.5 g/dL. Similarly, the learned limits for the prothrombin time (10.52-10.93 seconds) are well within the reference range of 10–13 seconds. Triglycerides also have a limit near 137 mg / dL, close to the reference value of <150 mg/dL.

Earlier in this work, we justified using three bins per feature to intuitively capture low–normal–high

[9] https://www.accp.com/docs/sap/Lab_Values_Table_PSAP.pdf

Table 3
Comparison between the maximum number of parts allowed for the model (hyperparameter) and the number of unique prototype parts actually discovered by MEDIC for the optimal configuration found during hyperparameter optimization.

	Cirrhosis	CKD	Diabetes
Max. allowed	52	40	44
Unique found	14	10	8

Table 4
Discretized feature intervals learned by the MEDIC network on the Cirrhosis dataset.

Feature	Interval 1	Interval 2	Interval 3
Albumin	$(-\infty, 3.70)$	$[3.70, 3.82)$	$[3.82, \infty)$
ALP	$(-\infty, 3668)$	$[3668, 4113)$	$[4113, \infty)$
Bilirubin	$(-\infty, 0.79)$	$[0.79, 3.43)$	$[3.43, \infty)$
Cholesterol	$(-\infty, 345)$	$[345, 668)$	$[668, \infty)$
Copper	$(-\infty, -8.98)$	$[-8.98, 103.76)$	$[103.76, \infty)$
N_days	$(-\infty, 2152)$	$[2152, 2343)$	$[2343, \infty)$
Platelets	$(-\infty, 271)$	$[271, 291)$	$[291, \infty)$
Prothrombin	$(-\infty, 10.52)$	$[10.52, 10.93)$	$[10.93, \infty)$
SGOT	$(-\infty, 80)$	$[80, 144)$	$[144, \infty)$
Tryglicerides	$(-\infty, 137)$	$[137, 172)$	$[172, \infty)$

ranges. However, for certain features in specific disease contexts, deviations in one single direction may be clinically significant. Interestingly, in this experiment the observations suggest the network to exhibit the ability to self-organize and adjust these bins accordingly. For example, for copper, the first interval is effectively disabled by learning a negative upper bound (–8.98), which is not physiologically plausible, thus disregarding it. Likewise, for platelets and albumin, the network forms exceptionally narrow middle intervals, suggesting that small changes within this range may be critical for the classification.

Although Table 4 shows intervals ranging from $-\infty$ to ∞ for technical completeness, in practical applications these can be translated into clinically relevant and bounded intervals. For example, lower limits can be set to zero and upper limits can be capped according to known physiological limits, without affecting the model's learning performance. This translation can support physicians in interpreting the model's behavior more easily.

Subsequently, MEDIC identified several prototype parts, specific combinations of clinical features and value ranges, that it considers informative to predict patient outcomes related to cirrhosis. These prototype parts are presented in Table 5.

Next, we investigate how a specific patient case (shown in Table 6, classified as 0 – death) is internally processed and classified by MEDIC through its similarity to the nearest learned prototype parts. Each prototype part consists of a sparse conjunction of conditions over discretized or binary features, typically involving only a small number of dimensions. For example, a prototype part may specify conditions such as: Bilirubin level within [0.79, 3.43) mg/dL, absence of hepatomegaly (Hepatomegaly = 0), and drug usage indicated (Drug = 1). These concise feature subsets capture clinically meaningful patterns that contribute to the model's decisions. MEDIC's classification is driven by the similarity between the patient's description and the prototype parts, which are easily accessible and can be examined by the user, thereby offering transparent insight into the model's reasoning process.

The list of prototypes with the highest similarity to this example demonstrates how the model constructs its reasoning by combining interpretable substructures. Many of these substructures align with known clinical heuristics or highlight relevant feature interactions. For example, bilirubin, ALP (alkaline phosphatase), and N_Days (duration since patient registration) appear frequently in the most similar prototypes, highlighting their importance as clinical indicators influencing classification.

Table 5
Prototype parts identified by MEDIC for the cirrhosis dataset. With appropriate hyperparameters, MEDIC was allowed to find up to 52 prototype parts; however, many were repeated, resulting in only 14 unique prototypes.

	Prototype part
1	Bilirubin in $[0.79, 3.43) \wedge$ Hepatomegaly $= 0 \wedge$ Spiders $= 0$
2	Albumin $\in [3.82, \infty) \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ Hepatomegaly $= 0 \wedge$ Spiders $= 0$
3	Cholesterol $\in [667, \infty) \wedge$ Copper $\in [103.76, \infty) \wedge$ Hepatomegaly $= 0 \wedge$ Spiders $= 0$
4	Bilirubin $\in (-\infty, 0.79) \wedge$ Cholesterol $\in (-\infty, 345) \wedge$ Hepatomegaly $= 1 \wedge$ Spiders $= 0$
5	Albumin $\in [3.70, 3.82) \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ Cholesterol $\in (-\infty, 345) \wedge$ Hepatomegaly $= 0 \wedge$ Spiders $= 0$
6	Bilirubin $\in [0.79, 3.43) \wedge$ Cholesterol $\in (-\infty, 345) \wedge$ Hepatomegaly $= 0 \wedge$ Platelets $\in (-\infty, 271) \wedge$ Spiders $= 0$
7	Bilirubin $\in (-\infty, 0.79) \wedge$ Cholesterol $\in (-\infty, 345) \wedge$ Hepatomegaly $= 0 \wedge$ Platelets $\in (-\infty, 271) \wedge$ Spiders $= 0$
8	Albumin $\in [3.70, 3.82) \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ Cholesterol $\in (-\infty, 345) \wedge$ Hepatomegaly $= 0 \wedge$ Spiders $= 0$
9	Albumin $\in [3.82, \infty) \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ Cholesterol $\in (-\infty, 345) \wedge$ Hepatomegaly $= 1 \wedge$ Spiders $= 0$
10	ALP $\in (-\infty, 3668) \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ Hepatomegaly $= 0 \wedge$ N_Days $\in [2343, \infty) \wedge$ SGOT $\in [80, 144) \wedge$ Tryglicerides $\in (-\infty, 137)$
11	ALP $\in (-\infty, 366) \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ Drug $= 1 \wedge$ Hepatomegaly $= 0 \wedge$ N_Days $\in (-\infty, 2152) \wedge$ SGOT $\in [80, 144) \wedge$ Tryglicerides $\in [172, \infty)$
12	Albumin $\in [3.82, \infty) \wedge$ ALP $\in (-\infty, 3668) \wedge$ Drug $= 1 \wedge$ Hepatomegaly $= 0 \wedge$ N_Days $\in [2343, \infty) \wedge$ SGOT $\in (-\infty, 80) \wedge$ Tryglicerides $\in [172, \infty)$
13	ALP $\in (-\infty, 3668) \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ Drug $= 1 \wedge$ Hepatomegaly $= 0 \wedge$ N_Days $\in (-\infty, 2152) \wedge$ Prothrombin $\in (-\infty, 10.52) \wedge$ SGOT $\in [80, 144) \wedge$ Tryglicerides $\in [172, \infty)$
14	Albumin $\in [3.82, \infty) \wedge$ ALP $\in (-\infty, 3668) \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ Drug $= 1 \wedge$ Hepatomegaly $= 0 \wedge$ N_Days $\in [2343, \infty) \wedge$ Prothrombin $\in (-\infty, 10.52) \wedge$ SGOT $\in [80, 144) \wedge$ Tryglicerides $\in (-\infty, 137)$

Table 6
Feature values of the described patient case. According to MEDIC, outcome for this patient was 0 – death.

Feature	Albumin	ALP	Ascites	Bilirubin	Cholesterol	Copper	Drug	Edema	Hepatomegaly	N_Days	Platelets	Prothrombin	SGOT	Spiders	Tryglicerides
Value	2.27	728	1	0.8	370	42	1	1	1	1217	156	11	71	0	125

These prototypes highlight clinically relevant signals, such as low bilirubin, no hepatomegaly, and shorter hospital stays (N_Days), as contributors to the classification. Furthermore, the inclusion of interaction patterns, such as elevated triglycerides in the context of certain ranges of liver enzymes, reflects how the network captures more nuanced decision logic than simple thresholding.

1. **Similarity**: 0.864
   Prototype: N_Days $\in (-\infty, 2152) \wedge$ Drug $= 1 \wedge$ Hepatomegaly $= 0 \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ ALP $\in (-\infty, 366) \wedge$ SGOT $\in [80, 144) \wedge$ Tryglicerides $\in [172, \infty)$
2. **Similarity**: 0.846
   Prototype: Hepatomegaly $= 0 \wedge$ Spiders $= 0 \wedge$ Cholesterol $\in [667, \infty) \wedge$ Copper $\in [103.76, \infty)$
3. **Similarity:** 0.834
   Prototype: N_Days $\in [2343, \infty) \wedge$ Hepatomegaly $= 0 \wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ ALP $\in (-\infty, 3668) \wedge$ SGOT $\in [80, 144) \wedge$ Tryglicerides $\in (-\infty, 137)$
4. **Similarity:** 0.824

Prototype: N_Days $\in [2343, \infty) \wedge$ Drug = 1 $\wedge$ Hepatomegaly = 0 $\wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ Albumin $\in [3.82, \infty) \wedge$ ALP $\in (-\infty, 3668) \wedge$ SGOT $\in [80, 144) \wedge$ Tryglicerides $\in (-\infty, 137) \wedge$ Prothrombin $\in (-\infty, 10.52)$

5. **Similarity:** 0.798
Prototype: N_Days $\in (-\infty, 2152) \wedge$ Drug = 1 $\wedge$ Hepatomegaly = 0 $\wedge$ Bilirubin $\in [0.79, 3.43) \wedge$ ALP $\in (-\infty, 3668) \wedge$ SGOT $\in 80, 144) \wedge$ Tryglicerides $\in [172, \infty) \wedge$ Prothrombin $\in (-\infty, 10.52)$

Although MEDIC's prototype parts may resemble rules derived from DTs, they differ fundamentally in how they are used for decision making. DTs require strict rule satisfaction, whereas MEDIC allows partial matches to prototype parts, enabling more flexible and probabilistic reasoning. This tolerance to incomplete matches can improve robustness to noise, missing values, and borderline cases, which are common issues in clinical data due to measurement variability, incomplete testing, or inconsistent documentation.

5. Conclusions

This work introduced MEDIC, a novel prototype parts-based neural network architecture that transforms the approach to interpretability in machine learning for medical tabular data. Unlike conventional post-hoc explanation methods that retrospectively justify black-box decisions, MEDIC represents a paradigm shift toward inherently interpretable models that mimic clinical reasoning patterns. The core innovation lies in our three-component architecture: (1) differentiable discretization that aligns with medical thresholds, (2) sparse patching masks that identify clinically meaningful feature combinations, and (3) prototype-based reasoning that grounds predictions in case-based comparisons, all unified within an end-to-end trainable framework.

Evaluation across three clinical datasets demonstrated that MEDIC achieves competitive and sometimes superior predictive performance compared to established methods while providing transparent decision processes. In particular, the model autonomously discovered discretization thresholds that closely align with clinically recognized reference ranges, as evidenced in our cirrhosis case study where albumin and prothrombin intervals closely matched established medical guidelines. Furthermore, the prototype parts learned by the model reflected combinations of features that correspond to recognizable diagnostic patterns, suggesting that MEDIC captures meaningful representations of clinical knowledge.

The implications of this work extend beyond technical innovation only. By bridging the gap between accuracy and interpretability, MEDIC addresses a critical barrier to AI adoption in healthcare, the lack of interpretability, which undermines the trust of clinicians and regulatory acceptance. Our approach supports collaborative human-AI decision making where the model's reasoning can be verified, critiqued, and integrated with clinical expertise.

The interpretability of MEDIC is achieved by grounding each prediction in prototypical parts – concise, clinically meaningful feature patterns drawn from real patient data and presented in natural, domain-specific language. Such clarity is essential for building trust, enabling clinicians to understand and validate the model's reasoning, and ensuring that AI-assisted decisions can be confidently integrated into medical practice.

Several promising directions emerge for future research. First, incorporating domain-specific prior knowledge into the prototype learning process could further align the model's representations with established medical understanding. Second, investigating methods for dynamic prototype adaptation could enable the model to update its learned representations in response to changes in symptoms over time. This drift in symptoms may result from evolving disease variants, treatment effects, or changes in how diseases present between populations, as seen with COVID-19 [27]. Another important direction for future research is conducting formal user studies with medical professionals to assess the practical usefulness and cognitive accessibility of the explanations generated by MEDIC. Finally, conducting rigorous user studies with physicians would provide valuable insights into how this approach affects clinical decision making and how the model's explanations could be further optimized for maximum utility.

Acknowledgments

This research was funded in part by National Science Centre, Poland OPUS grant no. 2023/51/B/ST6/00545 and in part by PUT SBAD 0311/SBAD/0752 grant.

Declaration on Generative AI

The authors have not used generative AI tools in the creation of this work.

References

[1] E. Christodoulou, J. Ma, G. S. Collins, E. W. Steyerberg, J. Y. Verbakel, B. Van Calster, A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models, Journal of Clinical Epidemiology 110 (2019) 12–22. doi:https://doi.org/10.1016/j.jclinepi.2019.02.004.

[2] V. Podgorelec, P. Kokol, B. Stiglic, I. Rozman, Decision trees: An overview and their use in medicine, Journal of medical systems 26 (2002) 445–63. doi:10.1023/A:1016409317640.

[3] L. Breiman, Random forests, Machine Learning 45 (2001) 5–32. doi:10.1023/A:1010933404324.

[4] C. Vlachas, L. Damianos, N. Gousetis, I. Mouratidis, D. Kelepouris, K.-F. Kollias, N. Asimopoulos, G. F. Fragulis, Random forest classification algorithm for medical industry data, SHS Web of Conferences 139 (2022) 03008. doi:10.1051/shsconf/202213903008.

[5] S. Bharati, M. R. H. Mondal, P. Podder, A review on explainable artificial intelligence for healthcare: Why, how, and when?, IEEE Transactions on Artificial Intelligence 5 (2024) 1429–1442. doi:10.1109/TAI.2023.3266418.

[6] F. Bodria, F. Giannotti, R. Guidotti, F. Naretto, D. Pedreschi, S. Rinzivillo, Benchmarking and survey of explanation methods for black box models, Data Mining and Knowledge Discovery 37 (2023) 1719–1778. doi:10.1007/s10618-023-00933-9.

[7] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Curran Associates Inc., 2017, p. 4768–4777. doi:10.5555/3295222.3295230.

[8] 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, San Francisco, CA, USA, August 13-17, 2016, 2016, pp. 1135–1144.

[9] L. Longo, M. Brcic, F. Cabitza, J. Choi, R. Confalonieri, J. D. Ser, R. Guidotti, Y. Hayashi, F. Herrera, A. Holzinger, R. Jiang, H. Khosravi, F. Lecue, G. Malgieri, A. Páez, W. Samek, J. Schneider, T. Speith, S. Stumpf, Explainable artificial intelligence (xai) 2.0: A manifesto of open challenges and interdisciplinary research directions, Information Fusion 106 (2024) 102301. doi:doi.org/10.1016/j.inffus.2024.102301.

[10] C. Molnar, Interpretable Machine Learning, 2 ed., Independently published, 2022. URL: https://christophm.github.io/interpretable-ml-book.

[11] A. Narayanan, K. Bergen, Prototype-based methods in explainable ai and emerging opportunities in the geosciences, in: Int. Conf. on Machine Learning (ICML) 2024 AI for Science Workshop, PLMR vol. 235, 2024. doi:10.48550/arXiv.2410.19856.

[12] C. Chen, O. Li, C. Tao, A. J. Barnett, J. Su, C. Rudin, This looks like that: deep learning for interpretable image recognition, in: Proceedings of the 33rd International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA, 2019. URL: 10.5555/3454287.3455088.

[13] J. Fürnkranz, Decision Tree In: Encyclopedia of Machine Learning, Springer US, Boston, MA, 2010, pp. 263–267. doi:10.1007/978-0-387-30164-8_204.

[14] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, 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, p. 785–794. doi:10.1145/2939672.2939785.

[15] A. A. Adeniran, A. P. Onebunne, P. William, Explainable ai (xai) in healthcare: Enhancing trust and transparency in critical decision-making, World Journal of Advanced Research and Reviews 23 (2024) 2647–2658.

[16] B. Kim, R. Khanna, O. O. Koyejo, Examples are not enough, learn to criticize! criticism for interpretability, in: Advances in Neural Information Processing Systems, volume 29, Curran Associates, Inc., 2016, pp. 2288–2296. doi:doi/10.5555/3157096.3157352.

[17] J. Karolczak, J. Stefanowski, A-PETE: Adaptive prototype explanations of tree ensembles, in: Progress in Polish Artificial Intelligence Research, volume 5, Warsaw University of Technology, 2024, pp. 2–8. URL: https://pages.mini.pw.edu.pl/~estatic/pliki/PP-RAI_2024_proceedings.pdf.

[18] X. Zhang, H. Xiao, R. Gao, H. Zhang, Y. Wang, K-nearest neighbors rule combining prototype selection and local feature weighting for classification, Knowledge-Based Systems 243 (2022) 108451. doi:10.1016/j.knosys.2022.108451.

[19] A. L. Beam, I. S. Kohane, Big data and machine learning in health care, JAMA 319 (2018) 1317–1318. doi:10.1001/jama.2017.18391.

[20] J. Karolczak, J. Stefanowski, This part looks alike this: identifying important parts of explained instances and prototypes, 2025. URL: https://arxiv.org/abs/2505.05597. arXiv:2505.05597.

[21] 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 32nd AAAI Conference on Artificial Intelligence, AAAI Press, 2018. doi:10.5555/3504035.3504467.

[22] L. A. De Santi, F. I. Piparo, F. Bargagna, M. F. Santarelli, S. Celi, V. Positano, Part-prototype models in medical imaging: Applications and current challenges, BioMedInformatics 4 (2024) 2149–2172. doi:10.3390/biomedinformatics4040115.

[23] G. Singh, S. F. Stefenon, K.-C. Yow, The shallowest transparent and interpretable deep neural network for image recognition, Scientific Reports 15 (2025) 13940. doi:10.1038/s41598-025-92945-2.

[24] J. Stefanowski, Dealing with data difficulty factors while learning from imbalanced data In: Challenges in Computational Statistics and Data Mining, Springer International Publishing, Cham, 2016, pp. 333–363. doi:10.1007/978-3-319-18781-5_17.

[25] D. Brzezinski, J. Stefanowski, R. Susmaga, I. Szczech, Visual-based analysis of classification measures and their properties for class imbalanced problems, Information Sciences 462 (2018) 242–261.

[26] J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for hyper-parameter optimization, in: Advances in Neural Information Processing Systems, volume 24, Curran Associates, Inc., 2011. doi:10.5555/2986459.2986743.

[27] V.-T. Tran, R. Porcher, I. Pane, P. Ravaud, Course of post covid-19 disease symptoms over time in the compare long covid prospective e-cohort, Nature Communications 13 (2022) 1812. doi:10.1038/s41467-022-29513-z.