*electronics*

MDPI

*Article*

# SparseProto: Biomedical Image Classification with Global Features and Prototype Histograms

**Hüseyin Aydilek**

Department of Electrical and Electronics Engineering, Faculty of Engineering and Natural Sciences, Kırıkkale University, Kırıkkale 71450, Türkiye; huseyinaydilek@kku.edu.tr

**Abstract**

SparseProto combines global convolutional features with a top-$k$ prototype-assignment histogram in a biomedical image classifier with 41,187 to 42,759 parameters. We evaluate predictive performance, parameter count, counted operations, and measured latency on BloodMNIST, PathMNIST, and DermaMNIST. Explainability is not evaluated. The original three-seed comparison illustrates that parameter and operation reductions do not translate proportionally into latency reductions. The prototype assignment is sparse only after a dense similarity computation and therefore does not establish computational sparsity. In a separate five-seed rerun, the full model differed from a convolutional neural network (CNN)-only variant by +0.38, −2.21, and +0.44 percentage points on the three datasets, while on PathMNIST its graphics processing unit (GPU) latency (single-precision floating-point arithmetic, FP32; batch size 1) was 2.45 times higher. The available evidence does not establish a consistent predictive advantage or a latency benefit of adding the prototype branch.

**Keywords:** biomedical image classification; lightweight architectures; prototype-based models; surrogate metrics; equivalence testing; latency measurement; reproducibility; MedMNIST

## 1. Introduction

In contemporary medical practice, diagnostic decisions predominantly depend on image interpretation. Histopathological examination remains the principal diagnostic approach for many diseases, and specimen digitization has enabled computational analysis [1]. Peripheral blood smear analysis remains a fundamental procedure, offering critical information for differentiating hematological diseases [2]. Dermoscopy substantially improves the diagnostic accuracy of pigmented skin lesions compared with unaided visual examination [3]. Furthermore, these images provide not only diagnostic insights but also prognostic information, such as survival data extracted from colorectal histology preparations [4].

Translating imaging evidence into clinical value relies on visual interpretation by trained experts, a resource that is both limited and costly. This interpretation is also variable. For example, in breast biopsy assessment, pathologists' overall concordance with expert opinion was 75.3%, with notably lower agreement in borderline cases [5]. Similarly, morphological classification of peripheral blood cells is labor-intensive and observer-dependent, prompting the development of datasets for automated recognition systems [6]. The accuracy improvement associated with dermoscopic evaluation depends on the examiner's training level, and the heterogeneity of lesion appearances further complicates this training process [3,7].

Check for updates

Academic Editor: Yongqin Zhang

Received: 24 September 2026
Revised: 9 November 2026
Accepted: 10 November 2026
Published: 12 November 2026

**Copyright:** © 2026 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

This persistent bottleneck has driven significant research into automated biomedical image classification. Deep learning, particularly convolutional neural networks, has emerged as the predominant methodology in this domain [8], with near-expert-level performance reported in narrowly defined tasks [9]. Nevertheless, a classifier's practical value depends on its real-world application rather than benchmark performance. In many scenarios where expert interpretation is a limiting factor, computational resources are also constrained. Consequently, deployment-environment considerations and associated costs are inseparable from model accuracy.

In many biomedical imaging contexts, computation runs on resource-limited hardware near the imaging device rather than in centralized data centers. Devices such as microscope workstations, portable dermoscopes, and field-use scanners operate under strict constraints regarding memory, power consumption, and latency. As a result, models with very low parameter counts are not only of academic interest but are also a practical necessity [10]. However, a reduced parameter count does not inherently guarantee that a model is cost-effective, fast, or reliable. This determination typically relies on three surrogate criteria, each with notable limitations.

The first surrogate criterion assumes that parameter count directly reflects computational cost. While parameter count determines model storage requirements, actual latency depends on memory access patterns and platform-specific characteristics, not solely on arithmetic operations. Consequently, speed improvements cannot be inferred from parameter or floating-point operation (FLOPs) ratios [10,11]. The second surrogate criterion assumes that overall accuracy adequately represents task utility. In biomedical applications with imbalanced class distributions, high accuracy may reflect success on the majority class, obscuring poor performance on minority classes; the effects of class imbalance on deep learning performance are well documented [12]. Therefore, overall accuracy alone is insufficient to demonstrate utility without macro-metrics and class-level recall [13,14]. The third surrogate criterion assumes that prototype-based architectures inherently provide explainability. However, learned prototypes do not necessarily correspond to human-interpretable concepts, and visual localization alone does not guarantee meaningful or reliable explanations; interpretability therefore cannot be assumed solely from architectural design [15,16]. This distinction is particularly important in high-stakes applications, where interpretability should be demonstrated rather than inferred from the nominal form of the model [17].

A fourth overarching issue concerns the reliability of experimental comparisons. Data sampling, initialization, and hyperparameter choices can introduce substantial variability [18]. Statistical interpretation also depends on clearly defining the independent comparison unit [19], while data leakage can further inflate apparent performance [20].

Current literature does not comprehensively address these four challenges in combination. Prototype-based decision models have predominantly been developed using architectures with millions of parameters and evaluated on images at their original resolution, with reported successes at this scale [21–24]. Extensions of these models to medical imaging have generally maintained similar scale assumptions [25,26]. In contrast, studies utilizing the MedMNIST benchmark have primarily focused on comparisons among backbone families, input resolutions, and training protocols [27,28]. The intermediate space—specifically, integrating global convolutional representations with a parameter budget in the tens of thousands, using a histogram derived from sparse local prototype assignment—remains underexplored. More critically, systematic verification of claims, such as cross-referencing structural assertions with source code and recalculating reported metrics from stored raw outputs, is not standard practice.

The present study directly addresses this gap through a two-part research question. The first component investigates the trade-off between accuracy and resource utilization achieved by combining global features with a sparse local assignment histogram in a network of approximately 42,000 parameters. The second component examines the extent to which available experimental evidence substantiates claims regarding this trade-off.

The research uses the MedMNIST benchmark, which encompasses diverse biomedical image types within a unified classification framework [29], thereby enabling evaluation across multiple imaging domains. The three selected tasks—BloodMNIST, PathMNIST, and DermaMNIST—differ in image type, sample size, and class balance. BloodMNIST features relatively balanced blood cell microscopy images; PathMNIST includes colorectal histology with approximately 90,000 training samples; DermaMNIST is a dermoscopic dataset dominated by a single class that comprises two-thirds of the test set. This diversity is necessary to address the second research component, as the observed trade-offs vary across tasks, as detailed in Section 4. Results from these tasks should be interpreted as comparative image-classification outcomes rather than validated clinical performance; success in low-resolution benchmarks does not guarantee pattern preservation in high-resolution images.

We report both positive and negative findings. The first finding is quantitative: although the proposed PathMNIST model contains about 261 times fewer parameters than ResNet18, its GPU forward pass is only 1.14 times faster. Thus, the parameter reduction does not translate proportionally to execution speed, illustrating the first surrogate-metric issue. The second finding is structural and follows from an inspection of the implementation.

The method computes all token-to-prototype similarities before top-$k$ selection and stores them in a dense tensor, indicating sparse assignment rather than sparse computation; therefore, the $k/P$ ratio cannot substantiate speed claims. Although the parameter ratio overstates the attainable speed-up, the proposed model records the lowest forward-pass latency of the four architectures in the three-seed resource measurement. This ordering is conditional on the device and numerical precision, as the later five-seed measurement in Section 4.2 shows.

The article's contributions are as follows: it introduces a convolutional and prototype-based hybrid network with 41,187 to 42,759 learnable parameters (depending on the dataset). It compares it to ResNet18, MobileNetV3-Small, and EfficientNet-B0 across three MedMNIST tasks using an identical training protocol. This approach enables measurement of the trade-off outlined in the first research question under consistent data flow and training conditions. To address the first surrogate-metric issue, we report the three resource axes separately and quantify their divergence, showing that speed cannot be inferred from parameter ratios. To address the second and third surrogate-metric issues, we present class-level performance, calibration, and prototype-usage distributions, along with comprehensive panels of correct and incorrect classifications for each PathMNIST class. For the fourth issue, we explicitly quantify the uncertainty associated with three-seed comparisons, show that all nine paired accuracy intervals include zero, and specify the requirements for defensible equivalence or non-inferiority claims [30]. Finally, to address the second research question, we conduct an evidence audit: all test metrics of the ablation collection are recomputed from the stored probabilities; parameter counts are derived from the layer dimensions of the implementation, and the structural negative finding noted above is documented.

Two further contributions follow from this audit. First, we adopt an equivalence margin of one percentage point of accuracy as an exploratory criterion, chosen after the experiments rather than registered before them, and justified in Section 3.7.

We compare the proposed model with its ablation arms at the level of individual test samples, using exact McNemar tests and bootstrap intervals over the test set rather than over three training replicates alone, and we report seed-level intervals alongside them. Second, this analysis characterizes the difference between the full configuration and each reduced configuration: conditional on the trained models, the full configuration leads the convolutional baseline on BloodMNIST but lags on PathMNIST and DermaMNIST, whereas all seed-level intervals include zero.

## 2. Related Works

MedMNIST v2 enables model behavior comparisons across varying dataset sizes and task types by integrating diverse image types into a unified classification interface [29]. However, high performance on a low-resolution benchmark does not guarantee that the same patterns hold in the original high-resolution images. Doerrich et al. showed that effective model prototyping requires considering both input resolution and training schemes, not just architecture selection, as shown on MedMNIST+ [27]. Similarly, Prvan et al. reported favorable efficiency–accuracy trade-offs on MedMNIST using lightweight ensemble combinations of neural-network backbones [28]. Therefore, the provenance of the dataset file provided to the network at $224 \times 224$ input should be reported separately for SparseProto, as upsampling from lower resolutions cannot recover lost detail.

The data origin defines the specific biological question the benchmark addresses. The source domains of the three tasks are clinically distinct: histopathological examination has become accessible to computational analysis through digitized preparations [1]; blood smear evaluation remains a standard laboratory procedure [2]; and dermoscopy offers a quantifiable improvement in accuracy over visual inspection [3]. The benchmark images are derived from these examinations and reduced to low resolution, but they do not substitute for the original clinical assessments.

In the colorectal histology study underlying PathMNIST, Kather et al. evaluated a system that distinguished nine tissue classes in colorectal histology patches using 7180 independent images [4]. Subsequent stages of this research, such as survival modeling, represent tasks distinct from the patch classification addressed here. The BloodMNIST dataset is derived from the microscopy collection of Acevedo et al., which comprises normal peripheral blood cells [6]; thus, high classification accuracy on this dataset does not equate to diagnostic capability for malignant hematological diseases. The DermaMNIST and HAM10000 datasets consist of dermoscopic images from various sources [7]. Abhishek et al. reported lesion-level overlap concerns in DermaMNIST and showed that data cleaning can substantially affect the interpretation of experimental results [31]. Because sample sizes change with corrected data splits, results from original and corrected datasets are not directly comparable methodologically. See the Limitations section for dataset traceability.

ResNet is a foundational family of architectures that enables the optimization of deep networks through residual connections [32]. MobileNetV3 integrates hardware-aware architecture search with network adaptation tailored for mobile processors [33]. EfficientNet addresses the challenge of balanced scaling across depth, width, and resolution [34]. Collectively, these studies indicate that model capacity cannot be adequately summarized by a single parameter count or layer number. In this study, the comparators are specific members of these families; however, we did not replicate the performance reported in the original publications in models trained from scratch.

The ShuffleNet V2 study underscores the importance of memory access patterns and platform characteristics in explaining the discrepancy between FLOPs and actual inference speed [11]; similar conclusions have been reached for embedded systems in recent endpoint inference compilations [10]. This distinction is particularly relevant for SparseProto, which

computes all token-prototype similarities, performs top-$k$ selection, and scatters results into a dense tensor. While reducing parameter count lowers storage requirements, separate measurements are necessary for latency and power consumption. A robust comparison should therefore standardize input size, batch size, numerical precision, warm-up procedures, and synchronization, and clearly specify unsupported operations. The Green AI literature [35] also recommends reporting computational cost alongside accuracy.

Compactness can be achieved not only by designing small architectures but also through model compression techniques such as knowledge distillation from a large teacher network to a smaller student network, pruning, and quantization, all of which constitute a well-established body of research [10,36]. The three-seed comparison concerns a small network trained from scratch. The five-seed collection in Section 4.2 adds an equal-budget CNN and a distilled equal-budget CNN; pruning and quantization were not evaluated.

ProtoPNet bases class decision-making on part similarities by comparing image regions to learned prototypes, thereby providing visual examples associated with each decision rather than only a prediction label [21]. ProtoTree assigns prototypes to decision-tree nodes and conveys local explanations through the image's decision path within the tree [22], reducing the number of prototypes and presenting decisions hierarchically. SparseProto's histogram representation does not produce a decision-tree structure. Deformable ProtoPNet addresses the limitations of fixed spatial order by adapting the relative positions of prototype parts to the image [23]. PIP-Net links prototype presence to a sparse scoring structure that contributes evidence to the class and emphasizes prototype alignment with human visual concepts [24]. ProtoPFormer minimizes background attention in Transformer representations by employing both global and local prototype branches [37]. These three methods each contribute to sparsity, spatial adaptation, or global–local knowledge in distinct ways; combining global features with prototype histograms in SparseProto is not functionally equivalent to any single approach.

Zhu et al. develop nonparametric prototypes by clustering features from pretrained foundation models and assess both the distinctiveness and comprehensiveness of the resulting explanations [38]. This methodology directly addresses whether multiple prototypes represent redundant visual concepts. The squared cosine diversity loss in SparseProto reduces inter-vector similarity but does not quantify semantic diversity. Additionally, systems utilizing pretrained foundation-model features cannot be directly compared to small CNNs trained from scratch without controlling for pre-training data and total model cost. The training-from-scratch paradigm interacts differently with medical imaging tasks compared to natural image tasks [39].

Moffett et al. examine the impact of cosine similarity on training and evaluation outcomes in prototype-part models [40]. Their findings indicate that L2 normalization and cosine similarity in SparseProto should not be regarded as methodological contributions in isolation. The contribution is therefore assessed through the measurable behavior of the combined small feature extractor, top-$k$ histogram, and global branch.

Generating image-level representations from local features extends beyond prototype-based explanation models. NetVLAD, for example, incorporates a layer that aggregates local features around learnable centers during end-to-end training [41]. In contrast, SparseProto employs the spatial average of top-$k$ assignment weights rather than aggregating residual vectors relative to learned centers. Consequently, meaningful comparators for evaluating the histogram-based branch include not only large CNNs but also codebook- or feature-aggregation layers of comparable size. The five-seed collection includes one local-pooling control of equal budget (Section 4.2); learned residual aggregation of the NetVLAD type was not evaluated.

Sparsemax is a transformation that can directly produce probability outputs with zero values [42]. In contrast, SparseProto first selects $k$ similarities and then applies softmax exclusively to these selected values. The support sets and derivative behaviors of these mechanisms differ. Additionally, sparse class scoring in PIP-Net operates at a different level than per-token top-$k$ selection [24]. Therefore, when referring to "sparse" mechanisms, it is important to clearly distinguish between assignment, classification weights, and the computations performed. The validated feature for SparseProto is the sparse assignment executed following comprehensive similarity calculations.

Barnett et al. investigated case-based prototype explanations in mammographic mass classification [43]. Huy et al. explored concept-based similarity reasoning in medical image analysis and developed a concept atlas that physicians can refine [44]. Additional studies have adapted prototype-based architectures to medical imaging; for example, class part prototypes in breast cancer histology were designed to enhance both accuracy and interpretability [25], and a method combining capsule networks with prototype learning linked explanations to expert-defined appearance attributes [26]. The unifying feature of these approaches is that explanations are anchored to measurable criteria. These studies show that medical explanations require more than presenting similar images. For clinical relevance, it is necessary to examine the relationship between the highlighted image region and the target finding, expert evaluation, and potential sources of error.

Huang et al. introduced quantitative methods for assessing explanation consistency and stability in part-prototype networks [45]. In a human-centered investigation, Davoodi et al. found that part-prototype classifiers are not inherently interpretable by design [16]. The PixPNet study relates prototypes to displayed image regions through the receptive field to support visual localization [15]. Guo et al. demonstrated that classification confidence in modern networks may not correspond to empirical accuracy, and that post-calibration methods such as temperature scaling can address this issue [46]. For SparseProto, a false positive with confidence near 1 is not inherently contradictory, and area under the receiver operating characteristic curve (AUC) is not a suitable calibration metric. When calibration is applied, its parameters should be estimated on validation data before evaluation on the test set. The current experiments do not include post hoc temperature scaling. Reinke et al. discussed potential errors arising from metric selection in image analysis, while Maier-Hein et al. systematically addressed metric selection based on task characteristics [13,14]. The effects of class imbalance on deep learning performance and recommended countermeasures have also been reviewed [12]. For multi-class tasks with class imbalance, it is essential to report overall accuracy, macro-metrics, and class-level recall together. This approach highlights the difference between overall performance and the weakest class, as illustrated in the SparseProto PathMNIST case study.

Bouthillier et al. demonstrated that data sampling, initial weights, and hyperparameter selection contribute to variability in comparison results [18]. Demšar's work on multiple dataset comparisons underscores the importance of clearly defining the independent comparison unit [19]. In this article, the three seeds measure training variability on the same fixed test set and do not represent three independent patient cohorts. The seed, dataset, and test sample levels are not aggregated into a single sample size. Kapoor and Narayanan's analysis of data leakage further shows that incorporating test information into model selection and data-processing decisions can lead to overestimated results [20].

The Limitations section defines the scope of the empirical comparisons.

## 3. Materials and Methods

### *3.1. Datasets and Preprocessing*

We evaluated four models on three medical image datasets. To reduce model variability from initial conditions and assess result stability, we repeated all main experiments under three random seeds (42, 52, and 62). Thus, we evaluated 36 independent experimental runs consisting of four models, three datasets, and three random seeds.

In addition to the main comparisons, we performed an ablation analysis of 45 runs to examine the contribution of the proposed method's components to performance. We also examined seven hyperparameter configurations to evaluate the model's sensitivity to key hyperparameters. This experimental design evaluates not only the proposed method's overall classification success but also the contribution of its architectural components and its stability under hyperparameter variations.

To validate the results, we recalculated performance metrics using the class labels and prediction probabilities obtained during the testing phase. This recomputation covers the 45 ablation runs; the outputs of the main comparison were not included in it. To evaluate model complexity, we determined the number of trainable parameters directly from the network architecture's layer dimensions. We created the training, validation, and testing splits according to standard protocols for each dataset, and Table 1 presents the sample sizes [29].

**Table 1.** Sample sizes of the datasets.

| Datasets | Classes | Training | Validation | Test |
|---|---|---|---|---|
| BloodMNIST | 8 | 11,959 | 1712 | 3421 |
| PathMNIST | 9 | 89,996 | 10,004 | 7180 |
| DermaMNIST | 7 | 7007 | 1003 | 2005 |

We loaded the training, validation, and testing sections of each dataset independently and validated the sample sizes by comparing them with the defined partitions. We scaled the input image pixel intensities to the range [0, 1]. We converted single-channel grayscale images to three-channel format by duplicating them along the channel axis to match the three-channel input structure of the network architectures used. When necessary, we resized images to 224 $\times$ 224 pixels using bilinear interpolation. To ensure a consistent preprocessing protocol across experiments, we applied no data augmentation and did not use ImageNet-based mean-standard deviation normalization.

During training, we randomly shuffled the sample order each epoch, while preserving the original order of the validation and test sets. We excluded the final training batch, when smaller than the minibatch size, from optimization. However, because we re-shuffle training samples each epoch, the excluded samples may vary between epochs, preventing systematic exclusion of specific samples from training. In the validation and testing phases, we included all samples in the evaluation.

The original loading procedure permits a fallback to an alternative resolution release. Training uses the training split, model selection uses validation, and the selected model is evaluated once on the test split. Data provenance and split-independence limitations are consolidated in the Limitations section.

### *3.2. Model Architecture*

The model consists of a convolutional feature extractor and two branches that operate on the local features it generates. Figure 1 shows the forward pass. The input is passed through five convolution stages. The first stage is bias-free convolution, batch normalization, and sigmoid linear unit (SiLU) activation (CBA); the next four stages are

depthwise 3 × 3 convolution, pointwise 1 × 1 convolution, and depthwise separable (DS) blocks with residual linkage. The stage recursion is given by Equation (1). In each residual DS stage, $S_b$ denotes the shortcut transformation consisting of a 1 × 1 convolution and batch normalization when the dimension or stride changes; otherwise, it is the identity transformation.

$$X^{(0)} = x \in \mathbb{R}^{3\times 224\times 224}, \qquad X^{(b)} = \mathcal{B}_b\left(X^{(b-1)}\right), \quad b = 1,\ldots,5. \tag{1}$$

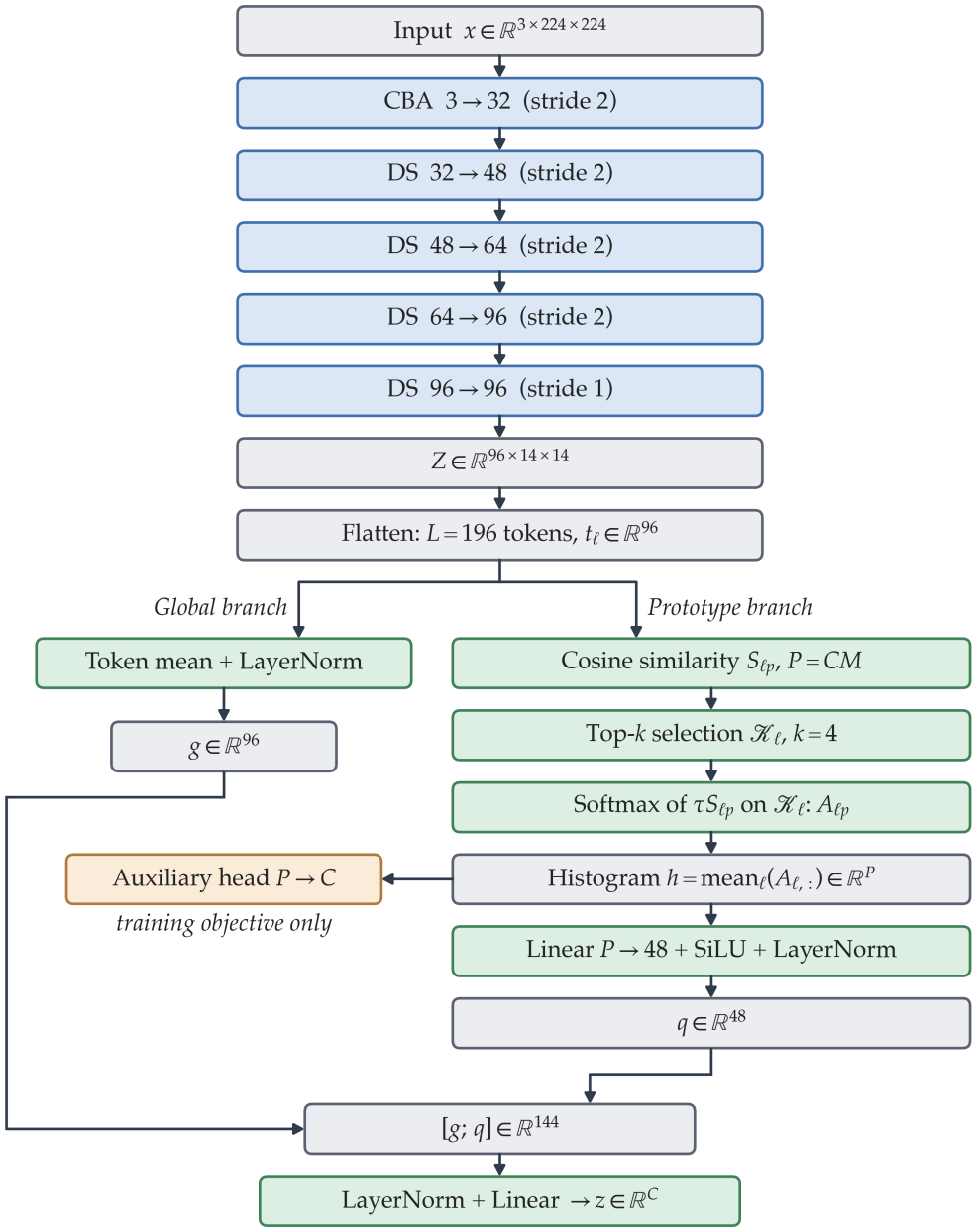



**Figure 1.** Complete SparseProto architecture. $B$ is omitted from the tensor sizes; $C$ is the number of classes and $P = CM$ the number of prototypes. The global branch computes the global feature $g$, and the prototype branch computes the usage histogram $h$ and the prototype feature $q$. CBA denotes convolution, batch normalization, and activation; DS denotes depthwise separable convolution. SiLU denotes the sigmoid linear unit, and LN denotes layer normalization. The auxiliary head is used only in the training objective.

Flattening the spatial dimensions of the final feature map yields $L$ tokens, each of dimension $d$. Equation (2) and Table 2 specify the relevant dimensions at each layer.

$$Z = X^{(5)} \in \mathbb{R}^{d\times H\times W}, \quad d = 96, \quad H = W = 14, \qquad t_\ell = Z_{:,i,j}, \quad L = HW = 196. \tag{2}$$

**Table 2.** Layer sizes of the feature extractor.

| Block | Channel Transformation | Stride | Output $C \times H \times W$ |
|---|---|---|---|
| CBA | $3 \to 32$ | 2 | $32 \times 112 \times 112$ |
| DS 1 | $32 \to 48$ | 2 | $48 \times 56 \times 56$ |
| DS 2 | $48 \to 64$ | 2 | $64 \times 28 \times 28$ |
| DS 3 | $64 \to 96$ | 2 | $96 \times 14 \times 14$ |
| DS 4 | $96 \to 96$ | 1 | $96 \times 14 \times 14$ |

The global branch produces the global feature g by applying layer normalization to the token mean as defined by Equation (3).

$$g = \mathrm{LN}_{96}(\bar{t}) \in \mathbb{R}^{96}, \qquad \bar{t} = \frac{1}{L}\sum_{\ell=1}^{L} t_\ell. \tag{3}$$

The prototype branch contains $P$ learnable prototype vectors and computes the cosine similarity between each token and each prototype, as described in Equation (4). When the number of classes is $C$ and the number of prototypes per class is $M$, $P = C \times M$. With the default $M = 4$, $P$ equals 32, 36, and 28 for BloodMNIST, PathMNIST, and DermaMNIST, respectively.

$$S_{\ell p} = \left\langle \frac{t_\ell}{\|t_\ell\|_2}, \frac{v_p}{\|v_p\|_2} \right\rangle, \qquad V = [v_1, \ldots, v_P] \in \mathbb{R}^{P\times d}, \qquad P = CM. \tag{4}$$

According to Equation (5), the index set of the highest $k$ similarities at each location is selected, and these values are scaled by a learnable multiplier, which is adjusted in the subsequent pass. In this context, the multiplier is defined as the similarity multiplied by the temperature, rather than as a temperature division.

$$\mathcal{K}_\ell = \underset{p\le P}{\arg\operatorname{topk}} S_{\ell p}, \qquad |\mathcal{K}_\ell| = k, \qquad \tau = \mathrm{clip}(\tilde{\tau}, 1, 30). \tag{5}$$

The selected prototypes are weighted using Equation (6), and all unselected assignments are zero. Averaging the assignment matrix over positions yields the $P$-dimensional usage histogram, defined by Equation (7) with nonnegative entries that sum to unity.

$$A_{\ell p} = \begin{cases} \dfrac{\exp(\tau S_{\ell p})}{\sum_{r\in\mathcal{K}_\ell}\exp(\tau S_{\ell r})}, & p \in \mathcal{K}_\ell, \\ 0, & p \notin \mathcal{K}_\ell. \end{cases} \tag{6}$$

$$h = \frac{1}{L}\sum_{\ell=1}^{L} A_{\ell,:} \in \mathbb{R}^P, \qquad \sum_{p=1}^{P} h_p = 1, \qquad h_p \ge 0. \tag{7}$$

The histogram is converted into a 48-dimensional feature by passing it through a linear projection, SiLU, and layer normalization, as in Equation (8). Combining the global and

prototype features creates a 144-dimensional representation, and Equation (9) gives the classification logits produced by the final layer.

$$q = \mathrm{LN}_{48}\big(\sigma(W_q h + b_q)\big) \in \mathbb{R}^{48}, \qquad W_q \in \mathbb{R}^{48\times P}. \tag{8}$$

$$z = W_o\,\mathrm{LN}_{144}([g;q]) + b_o \in \mathbb{R}^{C}, \qquad \widehat{y} = \arg\max_{c}\ \mathrm{softmax}(z)_c. \tag{9}$$

### *3.3. Training Objective*

The full-model loss is based on the main classification loss, an auxiliary classification loss fed from the prototype histogram, and a diversity term that reduces the inter-prototype squared cosine similarity. Equation (10) defines the diversity term, and Equation (11) defines the total loss. The diversity term does not distinguish between same or different classes. CNN-only uses the main cross-entropy loss.

$$\mathcal{L}_{\mathrm{div}} = \frac{1}{P(P-1)} \sum_{p\neq r} \left\langle \frac{v_p}{\|v_p\|_2}, \frac{v_r}{\|v_r\|_2} \right\rangle^2. \tag{10}$$

$$\mathcal{L} = \mathrm{CE}(z,y) + \lambda_{\mathrm{aux}}\,\mathrm{CE}(W_a h + b_a, y) + \lambda_{\mathrm{div}}\mathcal{L}_{\mathrm{div}}, \qquad \lambda_{\mathrm{aux}} = 0.20, \quad \lambda_{\mathrm{div}} = 0.01. \tag{11}$$

### *3.4. Computational Characteristics*

Equation (12) summarizes the method's computational character. Since the entire similarity matrix and the zeroed-out dense assignment tensor are generated in memory, the $k/P$ ratio does not directly mean that the number of calculated operations decreases proportionally; the dense similarity calculation remains necessary, and selection introduces an additional cost that depends on the algorithm and on $k$. Equation (12) gives a heap-based selection bound (Section: Limitations). The mechanism is therefore more accurately described as sparse prototype assignment than as sparse computation. The model performs no prototype-to-class mapping and no class-specific masking.

$$\mathrm{cost}(S) = \Theta(BLPd), \qquad \mathrm{cost}(\mathrm{top}\,k) = O(BLP\log(1+k)). \tag{12}$$

### *3.5. Training and Model Selection*

All models are initialized from scratch, without pretrained weights. We use the AdamW optimizer with a learning rate of 0.001, weight decay of 0.0001, a batch size of 64, and 100 training epochs. We reduce the learning rate by a factor of 0.1 after epochs 50 and 75. Training is carried out with automatic mixed precision, and gradients are clipped to a maximum norm of 1.0. Prototype similarities, assignment weights, the histogram, and the auxiliary head are computed in FP32 within a block excluded from autocasting. A copy of the assignment tensor is returned in the token data type for the usage analysis of Section 3.8; under autocast this type is half precision.

Figure 2 summarizes the training, resumption, validation, and model-selection procedure. Each dataset–model–seed combination constitutes a separate run. A combination for which an archived result already exists is not retrained and that result is reused; otherwise the random number generators are seeded, the data are loaded, and the model and optimizer are initialized. Where a checkpoint is available, the model, optimizer, schedule, and gradient-scaler states are restored and training resumes from that point. Within each epoch, the training samples are shuffled with a seeded generator, and the forward pass, loss computation, backpropagation, and optimizer step are applied minibatch-wise. Checkpoints are written every 25 minibatches. At the end of each epoch, we perform validation and, if the selection criterion is met, update the best checkpoint.

Model selection accepts an AUC increase greater than 0.0001, or an accuracy increase when the absolute AUC difference is at most 0.0001; both reference values are then updated. This rule governed all 89 recorded runs and permits small AUC decreases. Python (version 3.13.14), NumPy (version 2.4.6), and PyTorch CPU/CUDA generators are seeded; cuDNN autotuning and TF32 remain enabled, and deterministic algorithms are not enforced. Reproducibility limitations are discussed in the Limitations section.

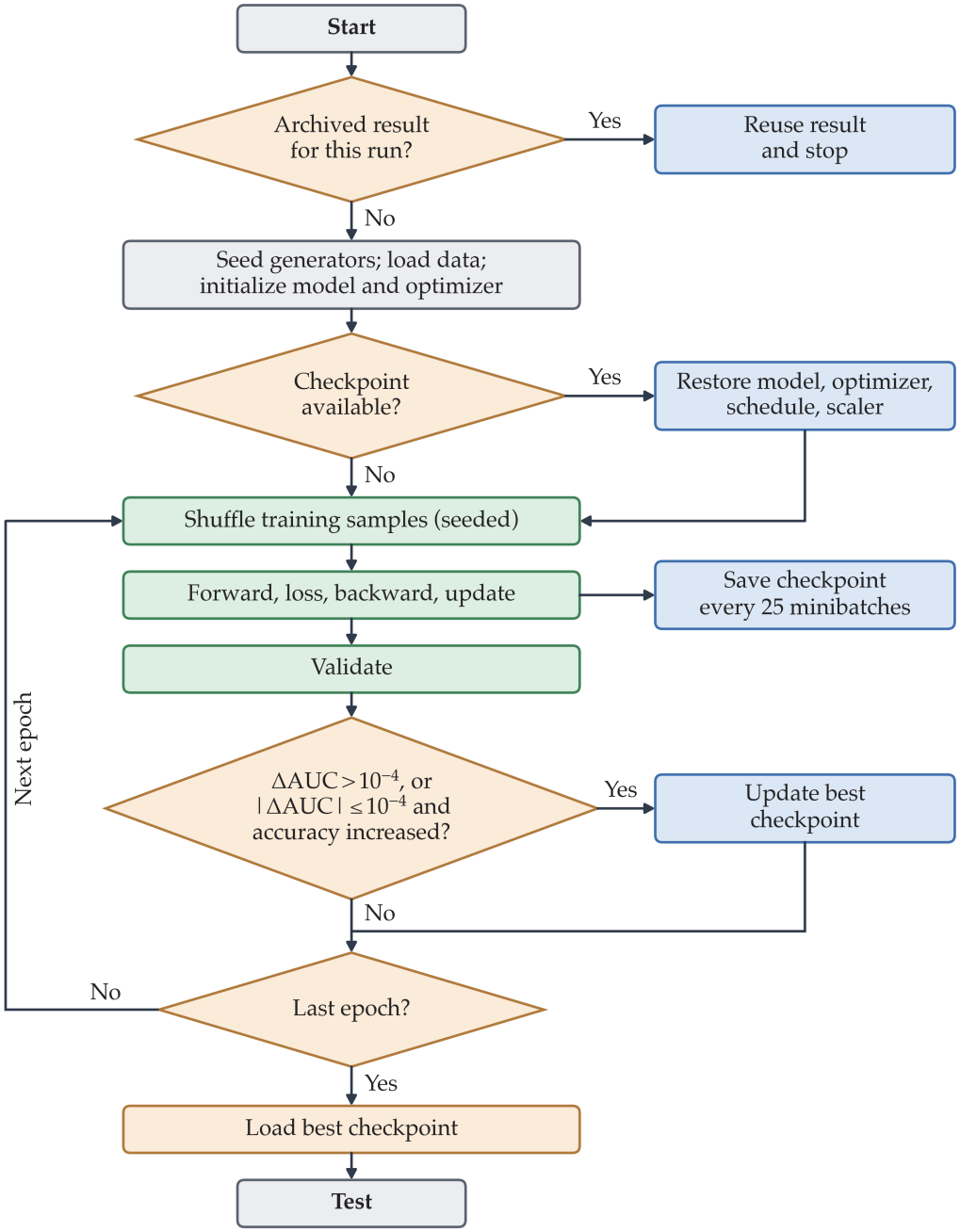



**Figure 2.** Training and model-selection procedure. The branches for reusing an archived result and for resuming from a checkpoint reproduce the original implementation. Checkpoint and configuration limitations are discussed in the Limitations section.

### *3.6. Evaluation Metrics*

Accuracy is the rate of correct predictions across all samples. Macro precision, recall and F1 are equally weighted averages of the class measures. The multi-class Matthews correlation coefficient (MCC) is calculated from the confusion matrix. The area under the

receiver operating characteristic curve (AUC) is calculated using a one-vs.-rest approach in each column of one-hot labels and averaged over the classes. The expected calibration error is defined by Equation (13) over 15 equally wide confidence intervals [46].

$$\mathrm{ECE} = \sum_{m=1}^{15} \frac{|B_m|}{N} |\mathrm{acc}(B_m) - \mathrm{conf}(B_m)|. \tag{13}$$

For each run, we verified the finiteness of probabilities, their range, row sums, argmax estimates, and label sequences. The primary discrimination metrics align with the archive within tolerances of $10^{-9}$ for discrimination, $10^{-7}$ for calibration error, and $10^{-6}$ for probability row sums. See the Limitations section for the scope of the audit. Results are reported as mean ± sample standard deviation across three training seeds. We computed the exploratory 95% range for paired accuracy differences using Equation (14), assuming independent training replicates and approximately normal differences; this does not account for sampling uncertainty in new patients. Including zero in an interval is not evidence of equivalence; such a conclusion requires a threshold specified in advance and a correspondingly powered design [30]. The margin adopted here is defined in the following subsection.

### *3.7. Statistical Analysis*

Two sources of uncertainty are distinguished throughout. Training variability is summarized by the sample standard deviation over the three seeds. Test-set sampling uncertainty is quantified separately by a nonparametric bootstrap over test samples with 10,000 resamples, in which the resampled statistic is the accuracy difference averaged over the three trained models; the resulting intervals are conditional on those three models and do not extend to new patients or specimens. Where per-sample predictions are available for both arms of a comparison, the exact McNemar test is applied to the discordant pairs of each seed separately, since the three seeds are replicates of training and not independent samples of the population. An equivalence margin of one percentage point of accuracy is used, and the two one-sided tests are implemented as the requirement that the two-sided 90% bootstrap interval of the difference fall entirely within ±1 percentage point, this being the interval level that corresponds to the procedure [30]. Both the 90% interval that carries the decision and the 95% interval that describes the difference are reported for every comparison. The margin was not registered before the experiments, so this analysis is exploratory rather than confirmatory. One percentage point is an evaluative choice rather than a quantity derived from the data. No clinical or deployment criterion fixes the size of an accuracy difference that would matter for these three tasks, and the between-seed standard deviation of the proposed model, which is 0.21, 1.10, and 1.90 percentage points on BloodMNIST, PathMNIST, and DermaMNIST, describes measurement variability rather than practical importance; the two quantities should not be equated. Because the choice is to this extent arbitrary, every equivalence decision is reported at three margins, ±0.5, ±1, and ±2 percentage points, so that its dependence on the margin can be inspected. Proportions such as class-level recall are reported with Wilson intervals. Seed-level paired intervals, computed with Equation (14), are reported for both the main comparison and the ablation arms; sample-level paired inference was performed for the ablation arms only. The records available for each collection are set out in the data availability statement.

$$\Delta_s = a_s^{\mathrm{SP}} - a_s^{\mathrm{cmp}}, \qquad \bar{\Delta} \pm t_{0.975,2} \frac{s_\Delta}{\sqrt{3}}, \qquad t_{0.975,2} = 4.302653. \tag{14}$$

### *3.8. Prototype Usage Analysis*

The normalized usage entropy defined by Equation (15) measures how evenly prototype usage is distributed. Here, $n_p$ counts positive assignment events rather than summing assignment magnitudes; the convention $0 \log 0 = 0$ is used.

$$\begin{aligned} \widetilde{H} &= -\frac{1}{\log P} \sum_{p=1}^{P} \pi_p \log \pi_p, \\ \pi_p &= \frac{n_p}{\sum_r n_r}, \qquad n_p = \#\{(\ell, x) : A_{\ell p} > 0\}. \end{aligned} \tag{15}$$

### *3.9. Resource Measurement*

The three-seed resource measurement uses PyTorch 2.11.0+cu128, an NVIDIA L4 GPU (NVIDIA Corporation, Santa Clara, CA, USA), and an Intel Xeon E5-2650 v4 CPU (2.20 GHz; Intel Corporation, Santa Clara, CA, USA). Inputs are $224 \times 224$ pixels, with batch size 1. GPU timing uses half-precision floating-point (FP16) autocast, 30 warm-ups, 200 timed forward passes, and CUDA synchronization around each pass; CPU timing uses single-precision floating-point (FP32) arithmetic, 10 warm-ups, 100 passes, and up to four threads. Newly initialized evaluation models are timed without data transfer. The later repeated-session protocol is reported in Section 4.2.

The counter treats one fused multiply–add as one operation and reports unsupported operations as warnings. Counter coverage is discussed in the Limitations section [47].

### *3.10. Qualitative Error Analysis*

For qualitative analysis, we reviewed test predictions from the run with seed 42 of the PathMNIST main experiment alongside panels generated from the original test images. The qualitative analysis was performed on the archived predictions; no model was retrained and no new inference was carried out. For each true class, up to three correct and three incorrect examples were drawn at random with selection seed 2026; in an additional panel, the incorrect examples are ranked by the softmax confidence of the predicted class, the smallest test index being listed first.

The true and predicted labels of all 7180 test rows used for the panels agreed exactly with the main archive, and the SHA-256 checksums of the probability and prediction files of that run matched the recorded provenance. See the Limitations section for the scope of these checks. Each panel entry reports the image, its test index, the true class, the predicted class, the prediction confidence, and the probability assigned to the true class; images are enlarged with nearest-neighbour interpolation for display purposes only. Class names follow the MedMNIST distribution and are used in the class-level results in Section 4 and in the panels. We selected the two classes examined in detail in the main text post hoc because they had the lowest test-set recall; this selection constitutes an exploratory error review. We also provide full panels for the other seven classes to avoid focusing exclusively on selected results.

## 4. Results

Table 3 reports the main classification results of the four architectures on the three datasets, averaged over the three seeds.

**Table 3.** Classification performance of SparseProto and three comparator architectures on BloodMNIST, PathMNIST, and DermaMNIST over three training seeds.

| Dataset | Model | Accuracy (%) | AUC | Macro F1 | ECE |
|---|---|---|---|---|---|
| BloodMNIST | EfficientNet-B0 | 98.685 ± 0.051 | 0.9992 ± 0.0002 | 0.9876 ± 0.0006 | 0.0108 ± 0.0010 |
| | MobileNetV3-S | 98.412 ± 0.045 | 0.9992 ± 0.0001 | 0.9855 ± 0.0003 | 0.0132 ± 0.0010 |
| | ResNet18 | 98.480 ± 0.058 | 0.9993 ± 0.0001 | 0.9863 ± 0.0005 | 0.0108 ± 0.0004 |
| | SparseProto | 98.499 ± 0.211 | 0.9993 ± 0.0002 | 0.9859 ± 0.0022 | 0.0122 ± 0.0011 |
| PathMNIST | EfficientNet-B0 | 92.539 ± 1.687 | 0.9902 ± 0.0029 | 0.9082 ± 0.0182 | 0.0622 ± 0.0140 |
| | MobileNetV3-S | 92.210 ± 0.902 | 0.9904 ± 0.0019 | 0.9015 ± 0.0051 | 0.0638 ± 0.0081 |
| | ResNet18 | 91.188 ± 1.950 | 0.9920 ± 0.0014 | 0.8981 ± 0.0175 | 0.0746 ± 0.0164 |
| | SparseProto | 90.548 ± 1.096 | 0.9922 ± 0.0015 | 0.8844 ± 0.0112 | 0.0751 ± 0.0102 |
| DermaMNIST | EfficientNet-B0 | 75.096 ± 0.879 | 0.9196 ± 0.0094 | 0.4677 ± 0.0208 | 0.0526 ± 0.0320 |
| | MobileNetV3-S | 71.388 ± 2.497 | 0.8932 ± 0.0135 | 0.4458 ± 0.0117 | 0.1760 ± 0.0874 |
| | ResNet18 | 75.860 ± 0.724 | 0.9277 ± 0.0025 | 0.5205 ± 0.0418 | 0.0660 ± 0.0215 |
| | SparseProto | 73.067 ± 1.903 | 0.9212 ± 0.0045 | 0.4762 ± 0.0295 | 0.0437 ± 0.0130 |

Mean ± sample standard deviation; three seeds. AUC is a macro one-vs.-rest metric. ECE uses a 15-interval scale.

In BloodMNIST, SparseProto achieves an accuracy of 98.499 ± 0.211% and a macro-F1 score of 0.9859 ± 0.0022. EfficientNet-B0 attains a slightly higher average accuracy of 98.685%, representing a 0.185 percentage point increase. SparseProto's AUC is 0.999295, while ResNet18's AUC is 0.999293; this negligible difference does not constitute a practical advantage.

In PathMNIST, SparseProto accuracy is 90.548 ± 1.096%, and the macro-F1 value is 0.8844 ± 0.0112. EfficientNet-B0 and MobileNetV3-Small have average accuracies of 92.539% and 92.210%, respectively. Although SparseProto has the highest average AUC among the main models, it does not maintain this ranking in terms of accuracy. In DermaMNIST, SparseProto achieves 73.067 ± 1.903% accuracy and a macro-F1 of 0.4762 ± 0.0295. ResNet18 is higher, with 75.860% accuracy and a macro-F1 of 0.5205. Table 4 reports macro precision, macro recall, and MCC for the same runs. The results show a trade-off between accuracy and size by dataset, rather than a single overall winner.

**Table 4.** Additional classification metrics for SparseProto and three comparator architectures on the three MedMNIST datasets over three training seeds.

| Dataset | Model | Macro Precision | Macro Recall | MCC |
|---|---|---|---|---|
| BloodMNIST | EfficientNet-B0 | 0.9877 ± 0.0005 | 0.9875 ± 0.0007 | 0.9846 ± 0.0006 |
| | MobileNetV3-S | 0.9854 ± 0.0006 | 0.9855 ± 0.0001 | 0.9814 ± 0.0005 |
| | ResNet18 | 0.9866 ± 0.0008 | 0.9860 ± 0.0004 | 0.9822 ± 0.0007 |
| | SparseProto | 0.9862 ± 0.0023 | 0.9856 ± 0.0021 | 0.9825 ± 0.0025 |
| PathMNIST | EfficientNet-B0 | 0.9183 ± 0.0169 | 0.9130 ± 0.0134 | 0.9156 ± 0.0185 |
| | MobileNetV3-S | 0.9098 ± 0.0060 | 0.9079 ± 0.0056 | 0.9117 ± 0.0100 |
| | ResNet18 | 0.9123 ± 0.0125 | 0.9033 ± 0.0139 | 0.9012 ± 0.0209 |
| | SparseProto | 0.8939 ± 0.0109 | 0.8910 ± 0.0071 | 0.8929 ± 0.0117 |
| DermaMNIST | EfficientNet-B0 | 0.5494 ± 0.0577 | 0.4565 ± 0.0160 | 0.4932 ± 0.0133 |
| | MobileNetV3-S | 0.4965 ± 0.0105 | 0.4381 ± 0.0087 | 0.4461 ± 0.0255 |
| | ResNet18 | 0.5590 ± 0.0120 | 0.5086 ± 0.0589 | 0.5056 ± 0.0230 |
| | SparseProto | 0.5698 ± 0.0798 | 0.4764 ± 0.0354 | 0.4900 ± 0.0327 |

Mean ± sample standard deviation; three seeds.

Table 5 reports the number of parameters, the number of counted operations, and the measured latencies for each of the four models in each dataset.

**Table 5.** Parameter counts, counted operations, and measured inference latency of SparseProto and three comparator architectures on the three MedMNIST datasets.

| Dataset | Model | Parameters (M) | Counted Operations (G) | GPU (ms/img) | CPU (ms/img) |
|---|---|---|---|---|---|
| BloodMNIST | EfficientNet-B0 | 4.017796 | 0.400390 | 11.973 | 22.997 |
| | MobileNetV3-S | 1.526056 | 0.058634 | 7.568 | 10.071 |
| | ResNet18 | 11.180616 | 1.818558 | 3.402 | 19.037 |
| | SparseProto | 0.041969 | 0.033691 | 3.176 | 4.984 |
| PathMNIST | EfficientNet-B0 | 4.019077 | 0.400391 | 11.977 | 23.397 |
| | MobileNetV3-S | 1.527081 | 0.058635 | 7.634 | 10.447 |
| | ResNet18 | 11.181129 | 1.818558 | 3.411 | 19.361 |
| | SparseProto | 0.042759 | 0.033767 | 2.983 | 5.006 |
| DermaMNIST | EfficientNet-B0 | 4.016515 | 0.400388 | 12.014 | 22.391 |
| | MobileNetV3-S | 1.525031 | 0.058633 | 7.690 | 10.335 |
| | ResNet18 | 11.180103 | 1.818557 | 3.371 | 19.471 |
| | SparseProto | 0.041187 | 0.033616 | 3.002 | 4.950 |

Parameter counts were verified against the layer dimensions of the architecture. Operation counts exclude operations unsupported by the counter; one fused multiply–add is counted as one operation. The reported values are not complete FLOP counts. Latencies are means at batch size 1 (GPU: FP16 autocast; CPU: FP32). The CNN-only variant was not part of this measurement; its latency is reported in Section 4.2.

The number of learnable parameters obtained from the layer dimensions is 41,969 for BloodMNIST, 42,759 for PathMNIST, and 41,187 for DermaMNIST, in exact agreement with the archived values. The feature extractor contains 35,312 parameters, and the global LayerNorm contains 192 parameters; the remaining difference across datasets arises from the prototype bank, the auxiliary head, the projection, and the final classification head. The count also includes the learnable parameters of the auxiliary classifier.

The absolute measurements qualify the parameter ratio in two directions. In the direction of caution, the ordering of the comparators is itself not predicted by operation counts: MobileNetV3-Small executes 1.74 times the counted operations of SparseProto but requires 2.56 times its GPU latency, whereas ResNet18 executes 53.9 times the operations for only 1.14 times the latency, memory traffic and limited parallelism being possible explanations for this discrepancy [10,11]; this hypothesis was not tested, because the recorded profiling output reports operator times rather than memory traffic. In the direction of support, in this measurement SparseProto records the lowest forward-pass latency of the four architectures on both devices and in all three datasets: 2.98 to 3.18 ms per image on GPU, against 3.37 to 3.41 ms for ResNet18, 7.57 to 7.69 ms for MobileNetV3-Small, and 11.97 to 12.01 ms for EfficientNet-B0; and 4.95 to 5.01 ms per image on CPU, against 19.04 to 19.47 ms, 10.07 to 10.45 ms, and 22.39 to 23.40 ms respectively. This advantage is not proportional to the parameter budget, and it is conditional on the device and precision (Section 4.2).

The parameter count for SparseProto is approximately 35.7–37.0 times smaller than that of MobileNetV3-Small, 94.0–97.5 times smaller than EfficientNet-B0, and 261.5–271.4 times smaller than ResNet18, depending on the dataset. In PathMNIST, the ResNet18-to-SparseProto operation-count ratio is about 53.9, with GPU latency at 1.14 and CPU latency at 3.87. Because operation counting is incomplete and execution characteristics vary, a universal speedup cannot be inferred solely from parameter or FLOPs ratios [11]. Figure 3 presents accuracy alongside parameter budgets.

Table 6 reports the seed-matched accuracy differences between SparseProto and each comparator.

**Table 6.** Seed-matched accuracy differences between SparseProto and three comparator architectures on the three MedMNIST datasets, in percentage points.

| Dataset | Comparator | Mean Difference (pp) | 95% t-Interval (pp) |
|---|---|---|---|
| BloodMNIST | EfficientNet-B0 | −0.185 | [−0.836, +0.466] |
| | MobileNetV3-S | +0.088 | [−0.421, +0.596] |
| | ResNet18 | +0.019 | [−0.631, +0.670] |
| PathMNIST | EfficientNet-B0 | −1.992 | [−7.969, +3.986] |
| | MobileNetV3-S | −1.662 | [−6.471, +3.147] |
| | ResNet18 | −0.641 | [−7.900, +6.618] |
| DermaMNIST | EfficientNet-B0 | −2.028 | [−8.587, +4.530] |
| | MobileNetV3-S | +1.679 | [−5.341, +8.699] |
| | ResNet18 | −2.793 | [−8.036, +2.450] |

Difference = SparseProto − comparator; pp = percentage point. Matched seeds are 42, 52, and 62. Differences are calculated and centered per seed with full precision, so they may be separated to the third decimal place from the difference of rounded means in Table 3. Intervals represent training variability over a fixed test set.

All nine paired accuracy intervals contain zero. The width of these intervals reflects training variability alone and does not license a conclusion of equivalence; with three replicates, the smallest attainable two-sided $p$-value of a sign test is 0.25, so a sign test at the seed level cannot reach conventional significance whatever the size of the difference. Figure 4 also shows differences across individual seeds. These intervals describe training variability on a fixed test set, and test-set sampling uncertainty is not represented in them: the four architectures are evaluated on the same images, so their errors are correlated. A predetermined limit and a corresponding study design are required to conclude that the model with fewer parameters offers equivalent or non-inferior performance [30].

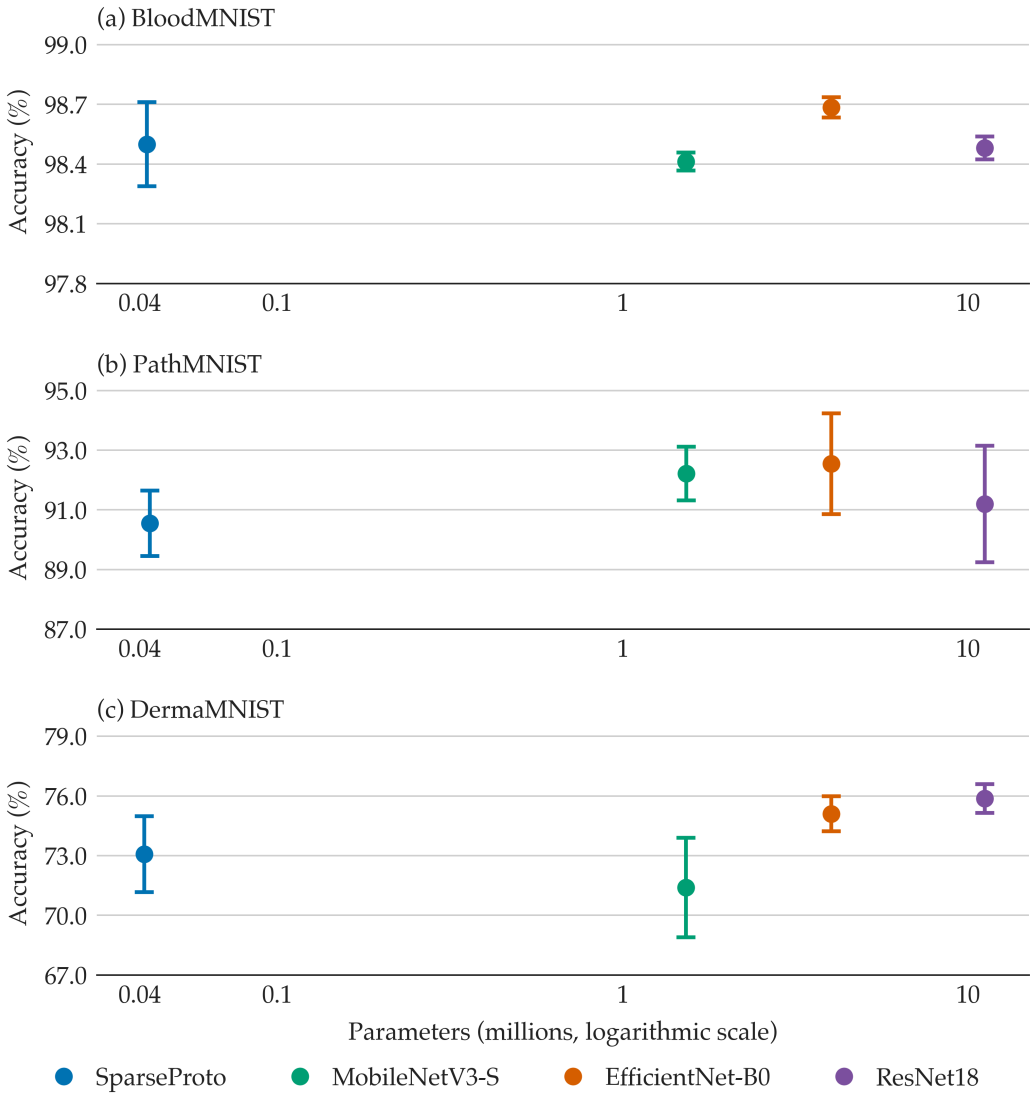



**Figure 3.** Accuracy and number of parameters. Points represent the three seed means, bars represent the sample standard deviation. The horizontal axis is logarithmic; vertical intervals differ between datasets. Panels (**a**), (**b**), and (**c**) show BloodMNIST, PathMNIST, and DermaMNIST, respectively.

The mean ECE values of SparseProto are 0.01217, 0.07510, and 0.04366 for BloodMNIST, PathMNIST, and DermaMNIST, respectively. In DermaMNIST, class 5 (melanocytic nevi) constitutes 1341 of 2005 test samples. In 23 samples of class 3 (dermatofibroma), the model makes only 1, 1, and 0 correct predictions for seeds 42, 52, and 62, respectively; recall values are 4.35%, 4.35%, and 0%.

In the run with seed 42 of the full PathMNIST model, validation accuracy was 99.720% (epoch 82), while test accuracy was 90.460%. This main run is distinct from the sensitivity run with the same nominal configuration in the hyperparameter sensitivity analysis below (99.710%, epoch 80) and from the ablation run (99.680%, epoch 79); the three values are separate trainings, not one value reported inconsistently. A difference in acquisition center is one possible explanation for this gap, but not the only one [29]. Without image and specimen identities, we cannot separate the contribution of data drift from other causes [20,31].

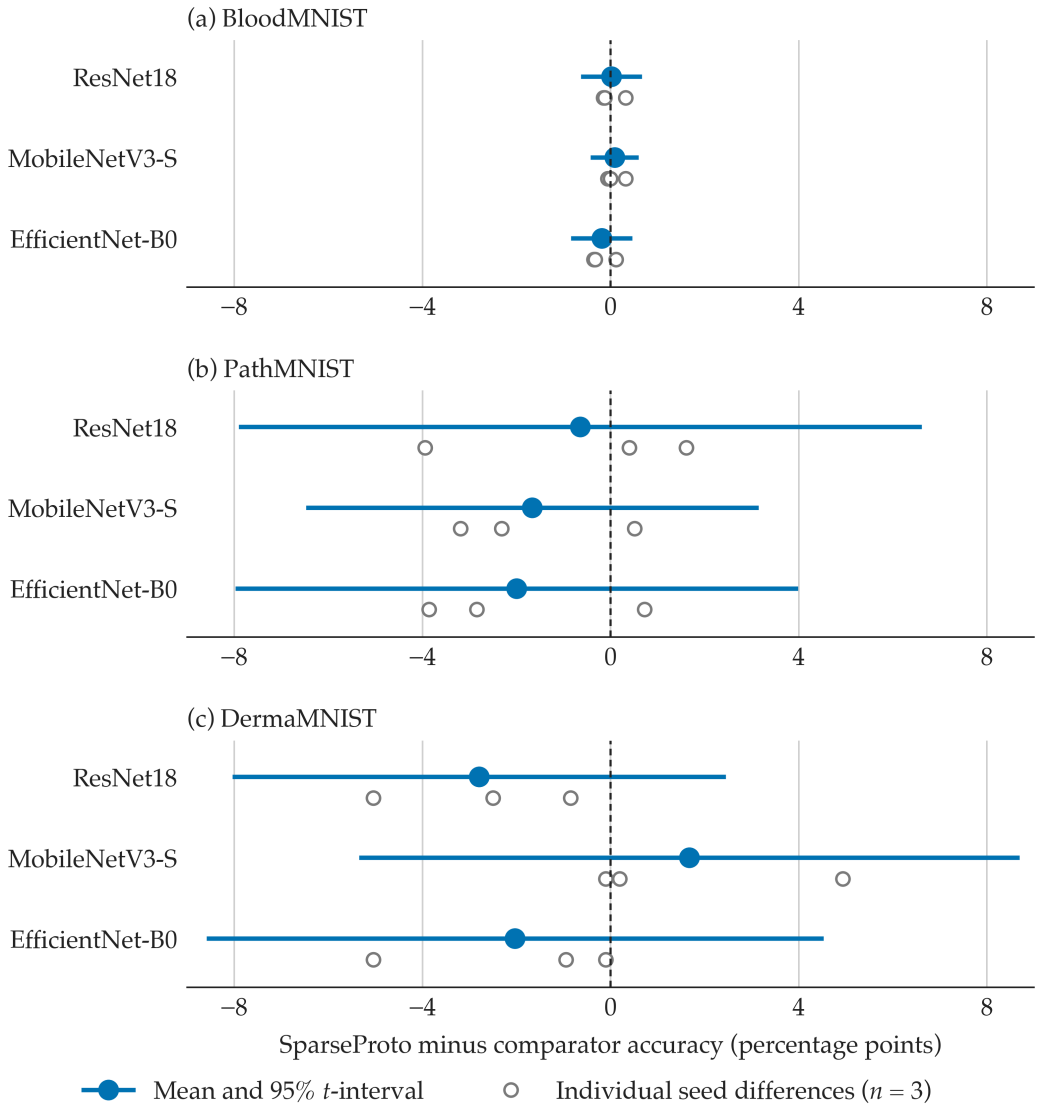



**Figure 4.** Accuracy differences between SparseProto and the comparator. Filled points are the average, lines are the exploratory 95% confidence intervals based on Student's $t$ distribution with two degrees of freedom, and empty points are individual seed differences. The positive difference favors SparseProto. The dashed vertical line marks zero difference. Panels (**a**), (**b**), and (**c**) show BloodMNIST, PathMNIST, and DermaMNIST, respectively.

Table 7 compares the four functional ablation arms within their own experimental collection. In PathMNIST, the full model is 2.12 percentage points lower than the CNN-only model; in DermaMNIST, it is 2.04 percentage points lower than the prototype-only model. The sample-level comparison reported below quantifies this pattern. On BloodMNIST the full configuration exceeds the CNN-only and the prototype-only arms, but it does not exceed the dense-assignment arm, whose mean accuracy is 98.529% against 98.509%; on none of the three datasets is the full configuration the highest-scoring arm.

The main and ablation full-model results differ across all nine pairs. For example, the PathMNIST mean accuracies are 90.548% in the main experiment and 89.327% in ablation. The two full-model configurations are architecturally and hyperparametrically identical; nondeterministic computation, resumption from a checkpoint, or the reuse of archived results may account for the discrepancy [48]. Because the archive does not record which of these applies, the higher value from the main experiment was not transferred to the ablation.

**Table 7.** Classification performance of the full SparseProto model and its ablation variants on the three MedMNIST datasets over three training seeds.

| Dataset | Variant | Accuracy (%) | AUC | Macro F1 |
|---|---|---|---|---|
| BloodMNIST | CNN-only | 98.061 ± 0.346 | 0.9992 ± 0.0000 | 0.9808 ± 0.0037 |
| | Dense prototype assignment | 98.529 ± 0.118 | 0.9993 ± 0.0002 | 0.9853 ± 0.0012 |
| | Prototype-only | 98.256 ± 0.122 | 0.9987 ± 0.0000 | 0.9836 ± 0.0008 |
| | SparseProto (full) | 98.509 ± 0.175 | 0.9992 ± 0.0002 | 0.9857 ± 0.0018 |
| PathMNIST | CNN-only | 91.448 ± 0.937 | 0.9928 ± 0.0018 | 0.8887 ± 0.0141 |
| | Dense prototype assignment | 89.912 ± 1.609 | 0.9918 ± 0.0020 | 0.8814 ± 0.0172 |
| | Prototype-only | 89.717 ± 0.817 | 0.9901 ± 0.0020 | 0.8760 ± 0.0059 |
| | SparseProto (full) | 89.327 ± 1.210 | 0.9912 ± 0.0011 | 0.8741 ± 0.0134 |
| DermaMNIST | CNN-only | 74.530 ± 0.502 | 0.9205 ± 0.0046 | 0.4702 ± 0.0275 |
| | Dense prototype assignment | 74.647 ± 0.948 | 0.9207 ± 0.0037 | 0.4854 ± 0.0258 |
| | Prototype-only | 75.145 ± 1.099 | 0.9231 ± 0.0071 | 0.4915 ± 0.0248 |
| | SparseProto (full) | 73.101 ± 2.104 | 0.9210 ± 0.0044 | 0.4848 ± 0.0241 |

Values are mean ± sample standard deviation over three seeds. The full model reference is from the ablation collection.

### *4.1. Sample-Level Comparison of the Ablation Arms*

Because per-sample predictions are available for the ablation collection, the performance differences between configurations can be assessed directly rather than inferred from three seed means. Table 8 reports, for each dataset, the accuracy difference between the full model and each remaining arm, the bootstrap interval of that difference over the test set, the equivalence verdicts at margins of ±0.5, ±1, and ±2 percentage points, and the range of the per-seed exact McNemar *p*-values.

**Table 8.** Sample-level paired comparison of the full model against the remaining ablation arms.

| Dataset | Comparison | Δ Accuracy (pp) | 90% CI (pp) | 95% CI (pp) | Seed-Level 95% t-Interval (pp) | Equivalence (±0.5/±1/±2 pp) | McNemar *p* Range |
|---|---|---|---|---|---|---|---|
| BloodMNIST | full − CNN-only | +0.45 | 0.22 to 0.67 | 0.19 to 0.72 | −0.10 to 0.99 | No/Yes/Yes | 0.007 to 0.341 |
| BloodMNIST | full − prototype-only | +0.25 | 0.07 to 0.44 | 0.03 to 0.48 | −0.05 to 0.56 | Yes/Yes/Yes | 0.134 to 0.636 |
| BloodMNIST | full − dense assignment | −0.02 | −0.18 to 0.14 | −0.21 to 0.17 | −0.36 to 0.32 | Yes/Yes/Yes | 0.362 to 1.000 |
| PathMNIST | full − CNN-only | −2.12 | −2.44 to −1.81 | −2.50 to −1.74 | −6.46 to 2.22 | No/No/No | $8.3 \times 10^{-33}$ to 0.194 |
| PathMNIST | full − prototype-only | −0.39 | −0.69 to −0.09 | −0.75 to −0.04 | −3.14 to 2.36 | No/Yes/Yes | $3.5 \times 10^{-7}$ to 0.690 |
| PathMNIST | full − dense assignment | −0.59 | −0.84 to −0.33 | −0.88 to −0.28 | −1.97 to 0.80 | No/Yes/Yes | $8.3 \times 10^{-5}$ to 0.880 |
| DermaMNIST | full − CNN-only | −1.43 | −2.23 to −0.62 | −2.38 to −0.48 | −5.68 to 2.82 | No/No/No | $1.0 \times 10^{-4}$ to 0.820 |
| DermaMNIST | full − prototype-only | −2.04 | −2.88 to −1.21 | −3.04 to −1.06 | −7.95 to 3.86 | No/No/No | $8.2 \times 10^{-7}$ to 0.863 |
| DermaMNIST | full − dense assignment | −1.55 | −2.16 to −0.93 | −2.28 to −0.80 | −8.89 to 5.80 | No/No/No | $6.7 \times 10^{-7}$ to 0.708 |

Difference = full model − arm, in percentage points of accuracy, averaged over seeds 42, 52, and 62. Intervals are nonparametric bootstrap intervals over 10,000 test-set resamples and are conditional on the three trained models. The 90% interval carries the equivalence decision; the 95% interval describes the difference. The seed-level interval follows Equation (14) on the three paired per-run accuracies and reflects training variability. Equivalence is declared when the two-sided 90% interval lies entirely within the stated margin. The three verdicts correspond to margins of ±0.5, ±1, and ±2 percentage points. McNemar *p*-values are exact and are computed separately for each seed.

All nine seed-level intervals in Table 8 include zero. Conditional on the archived models and fixed test sets, the full model leads CNN-only and prototype-only on BloodMNIST by 0.45 and 0.25 percentage points, respectively. Both 95% bootstrap intervals exclude zero and fall within the exploratory one-point margin. CNN-only leads on PathMNIST by 2.12 points and on DermaMNIST by 1.43 points, with equivalence unestablished at all three margins. Top-$k$ and dense assignment meet the one-point margin on BloodMNIST and PathMNIST, while dense assignment leads by 1.55 points on DermaMNIST. Widening the margin to two points changes no verdicts. Narrowing it to half a point changes three, covering CNN-only on BloodMNIST and prototype-only and dense assignment on PathMNIST. Per-seed McNemar results also vary, with PathMNIST CNN-only $p$-values ranging from $8 \times 10^{-33}$ to 0.194.

The hyperparameter sensitivity analysis covers only PathMNIST and seed 42 with $k \in \{1,2,4,8\}$ and $M \in \{1,2,4,8\}$; Table 9 reports the results. Validation accuracy increases from 99.490% to 99.730% as $k$ increases from 1 to 8. The difference between $k = 4$ and 8 is only 0.020 percentage points, i.e., approximately two additional correct predictions on 10,004 samples. $M = 1$ and $M = 4$ give the same validation accuracy. These results, with a single seed, a single dataset, and no interaction screening, do not prove this is the universally best setting [18].

**Table 9.** Validation-set accuracy and area under the receiver operating characteristic curve for the PathMNIST hyperparameter sensitivity experiments, seed 42.

| Sweep | $k$ | $M$ | Seed | Selected Epoch | Val. ACC (%) | Val. AUC |
|---|---|---|---|---|---|---|
| top-$k$ | 1 | 4 | 42 | 85 | 99.490 | 0.99995743 |
| top-$k$ | 2 | 4 | 42 | 72 | 99.700 | 0.99997499 |
| top-$k$ | 4 | 4 | 42 | 80 | 99.710 | 0.99997953 |
| top-$k$ | 8 | 4 | 42 | 80 | 99.730 | 0.99998434 |
| $M$ | 4 | 1 | 42 | 86 | 99.710 | 0.99998061 |
| $M$ | 4 | 2 | 42 | 60 | 99.660 | 0.99996733 |
| $M$ | 4 | 4 | 42 | 80 | 99.710 | 0.99997953 |
| $M$ | 4 | 8 | 42 | 92 | 99.680 | 0.99997085 |

All records are for PathMNIST and seed 42. The $k = 4$, $M = 4$ configuration occurs in two experimental series; these occurrences are not independent settings. No conclusions are drawn regarding test-set recall.

Table 10 summarizes how assignments are distributed across the prototype bank in each of the nine main runs.

**Table 10.** Prototype assignment usage and normalized usage entropy for the nine main SparseProto runs across three datasets and three training seeds.

| Dataset | Seed | Prototype | Normalized Entropy | Top-Four Share (%) |
|---|---|---|---|---|
| BloodMNIST | 42 | 32 | 0.7585 | 51.05 |
| | 52 | 32 | 0.7376 | 53.49 |
| | 62 | 32 | 0.7377 | 44.78 |
| PathMNIST | 42 | 36 | 0.7811 | 45.83 |
| | 52 | 36 | 0.7881 | 48.55 |
| | 62 | 36 | 0.7541 | 47.21 |
| DermaMNIST | 42 | 28 | 0.6709 | 76.30 |
| | 52 | 28 | 0.6822 | 74.68 |
| | 62 | 28 | 0.7080 | 73.29 |

The overall utilization of all prototypes is positive. Usage rankings are calculated separately for each seed.

Normalized usage entropy ranges from 0.738–0.758 in BloodMNIST, 0.754–0.788 in PathMNIST, and 0.671–0.708 in DermaMNIST. The four most frequently used prototypes carry 73.3–76.3% of DermaMNIST assignments. In Figure 5 the prototypes are ranked independently within each seed; an identical rank does not correspond to an identical prototype index. The usage analysis counts non-zero assignment events rather than summing assignment magnitudes. The counted tensor is the half-precision copy described in Section 3.5, so very small positive weights may be rounded to zero in the counts; the FP32 histogram passed to the classifier is unaffected. The small shortfalls in the number of active assignments observed on PathMNIST are consistent with this behaviour, but the exact cause was not tested. Counting the selected top-$k$ indices directly would separate numerical rounding from genuine selection frequency. In DermaMNIST, class 5 (melanocytic nevi) has the largest raw activation count in each prototype column. Since this class constitutes 66.88% of the test set, raw numbers do not reflect class-specific prototype purity. Raw activation counts should therefore be complemented by class-normalized usage and additional qualitative assessment when interpreting prototype behavior [16,45].

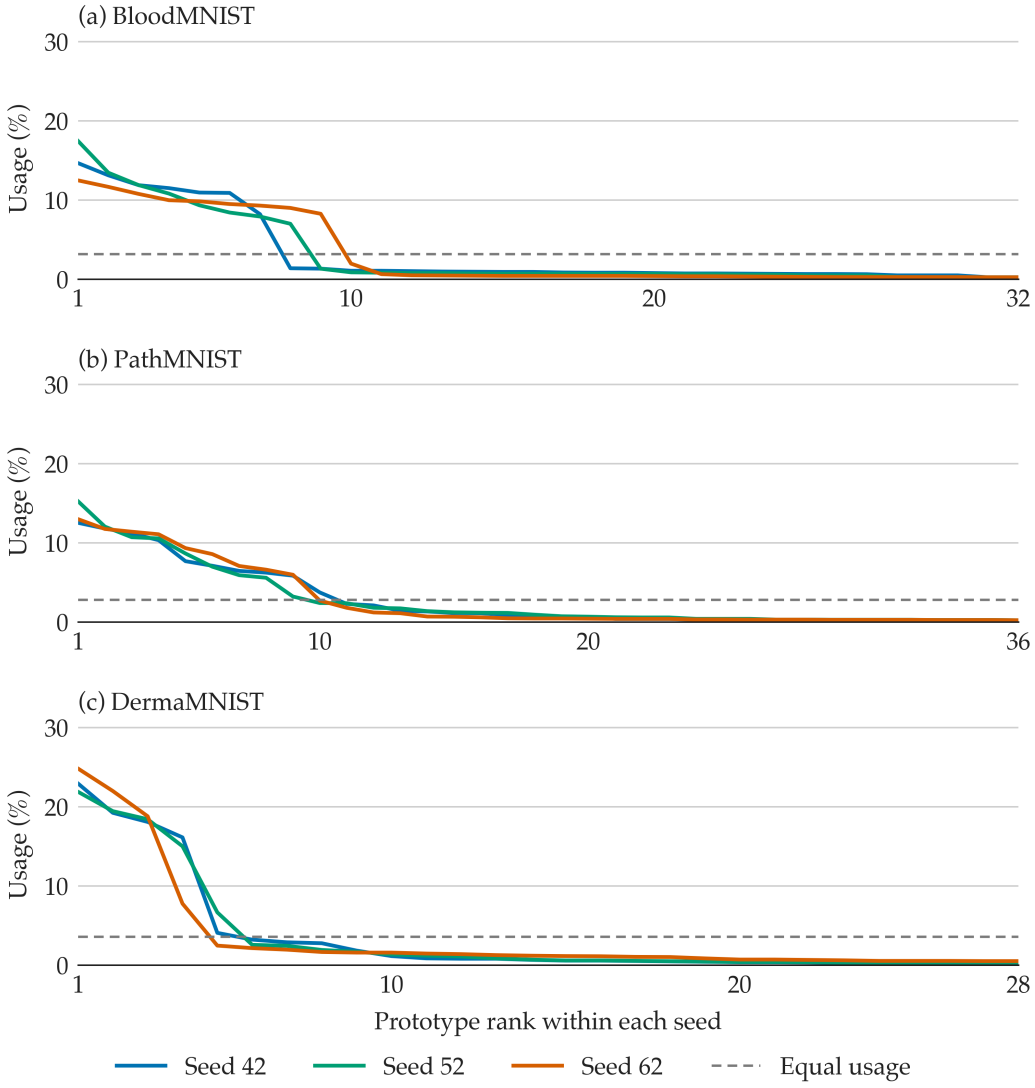



**Figure 5.** Ranked prototype usage. Each curve represents the usage distribution within a seed. The dashed line indicates equal usage. The ranks do not match prototype IDs and do not measure the quality of the visual explanation. Panels (**a**), (**b**), and (**c**) show BloodMNIST, PathMNIST, and DermaMNIST, respectively.

In the run with seed 42 examined here, 6495 of 7180 test images were correctly classified, and 685 were misclassified; accuracy was 90.460%. We evaluated this single run separately from the three seed averages in Table 3. Class-level counts are given in Table 11. Class 7 (cancer-associated stroma) had the lowest recall at 55.582%, followed by class 5 (smooth muscle) at 78.547%. In class 1 (background), all 847 samples were correctly classified; the corresponding error panels for this class are therefore empty.

Figure 6 shows random correct examples, random incorrect examples, and highest-confidence errors for class 7. In the random incorrect panel, examples with indexes 1606 and 2351 were estimated as class 2, and the example with index 4093 was estimated as class 4. All of the examples with indexes 76, 761, and 1041 in the high-confidence error panel were assigned to class 5. The 1.0000 confidence value shown in these examples does not mean that the decision is correct. The findings illustrate that confidence and accuracy should be evaluated separately [46].

Figure 7 shows the assignments of class 1 and class 2 among the random errors of class 5. The three errors with the highest confidence were assigned to class 2. Focusing on these two classes is a post hoc choice based on class-level recall; it is not a predetermined confirmatory subgroup analysis. Accordingly, visual appearance alone was not used to attribute the observed errors to staining, texture, or any specific biological pattern; such associations may reflect correlational or shortcut features rather than causal mechanisms [49].

**Table 11.** Class-level test results in the PathMNIST main SparseProto run.

| Class | Test Samples | Correct | Incorrect | Recall (%) | 95% CI (%) |
|---|---|---|---|---|---|
| 0 (adipose) | 1338 | 1144 | 194 | 85.501 | 83.51–87.29 |
| 1 (background) | 847 | 847 | 0 | 100.000 | 99.55–100.00 |
| 2 (debris) | 339 | 329 | 10 | 97.050 | 94.66–98.39 |
| 3 (lymphocytes) | 634 | 632 | 2 | 99.685 | 98.86–99.91 |
| 4 (mucus) | 1035 | 949 | 86 | 91.691 | 89.85–93.22 |
| 5 (smooth muscle) | 592 | 465 | 127 | 78.547 | 75.06–81.66 |
| 6 (normal colon mucosa) | 741 | 692 | 49 | 93.387 | 91.36–94.96 |
| 7 (cancer-associated stroma) | 421 | 234 | 187 | 55.582 | 50.81–60.26 |
| 8 (colorectal adenocarcinoma epithelium) | 1233 | 1203 | 30 | 97.567 | 96.55–98.29 |

Recall is the ratio of correctly classified samples to the total number of samples in that class. Class-level proportions are accompanied by Wilson intervals, because the smaller classes carry substantial uncertainty: the recall of class 7 is 55.58% (234 of 421; 95% CI 50.81% to 60.26%) and that of class 5 is 78.55% (465 of 592; 95% CI 75.06% to 81.66%). The corresponding problem is more severe on DermaMNIST, where class 3 contains 23 test samples: the observed recall of 4.35% has a 95% interval of 0.77% to 20.99%, and the observed 0% has an interval of 0% to 14.31%, neither of which supports any statement about the behaviour of the model on that class. Training seed 42; sample selection seed 2026. This table does not include a new training run.

The complete set of panels spans all classes and comprises 73 sample placements distributed over 27 panels, corresponding to 69 unique test indices. Figures 8, 9, 10, 11, 12, 13 and 14 cover classes 0, 1, 2, 3, 4, 6, and 8, respectively. For example, class 3 has only two incorrect samples; the same two images appear in both error panels, and these repetitions are not counted as new errors. Figures 6 and 7, along with other class panels, support error review; they do not show which regions the prototypes used to make the decision [15,45].

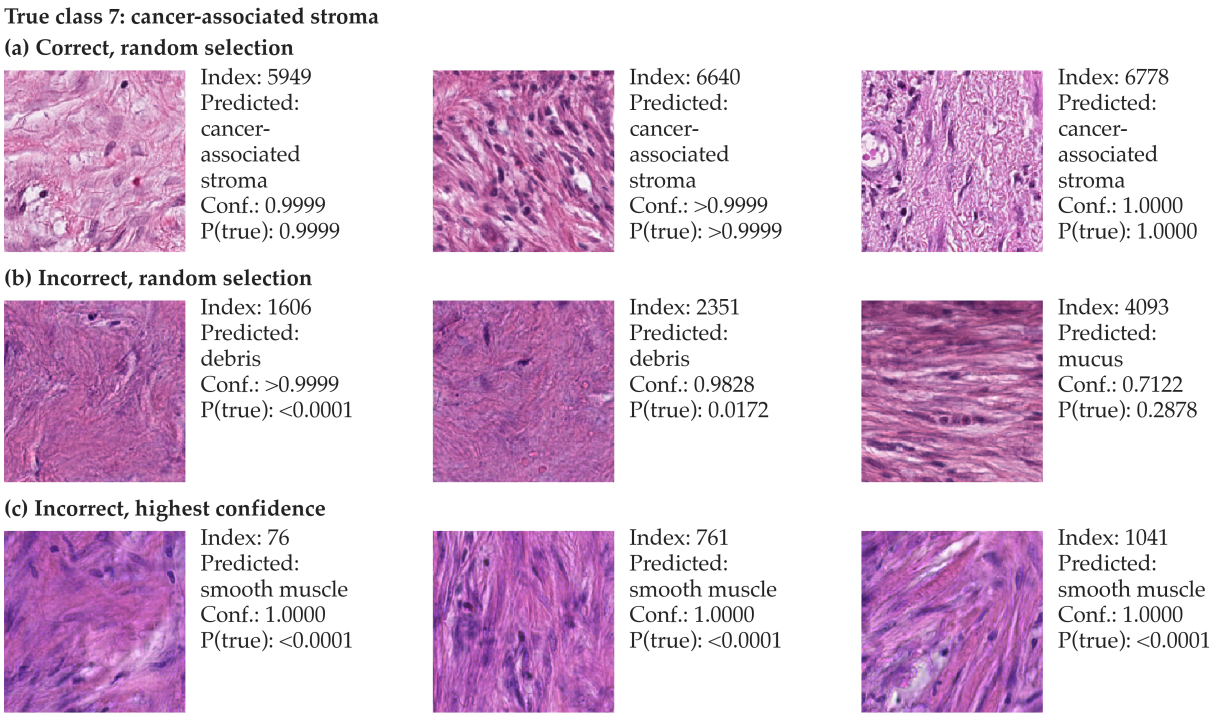



**Figure 6.** (**a**) Randomly selected correct classifications; (**b**) randomly selected incorrect classifications; (**c**) highest-confidence incorrect classifications for PathMNIST class 7 (cancer-associated stroma). Training seed 42, selection seed 2026. Section 3.10 describes the image-order assumption. Panels are not a success estimate representing all test samples of the class. Conf. denotes the softmax probability of the predicted class; P(true) denotes the softmax probability assigned to the true class.

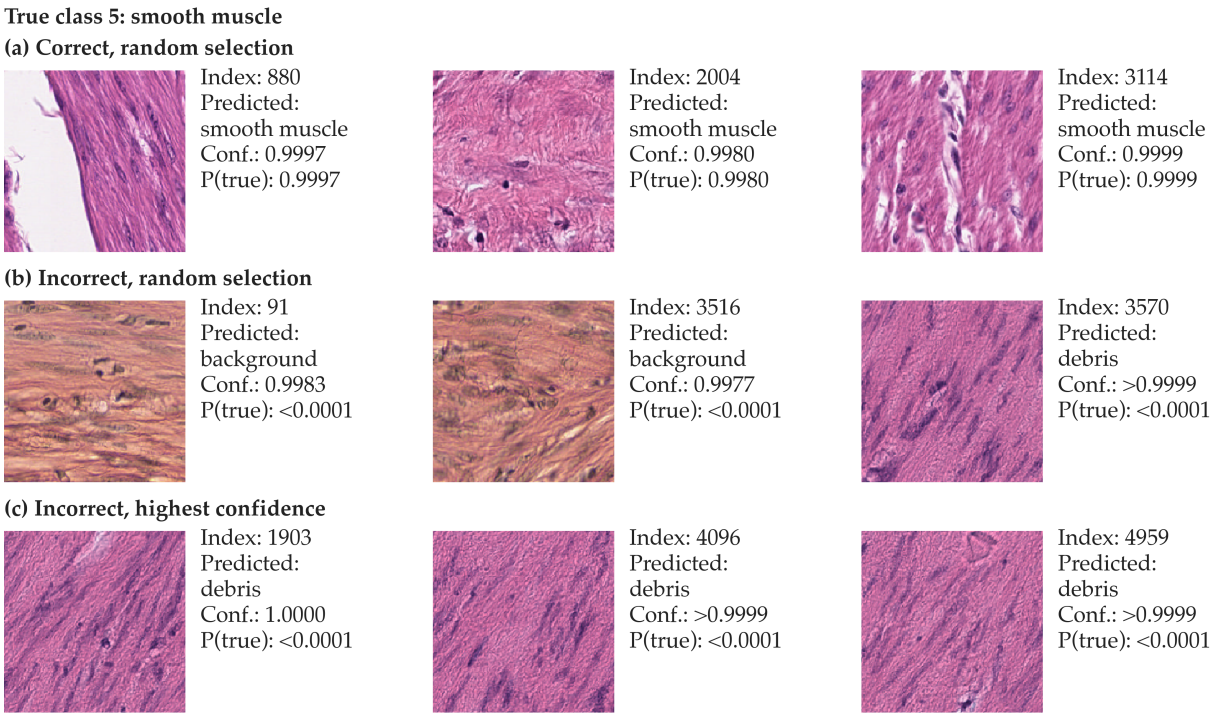



**Figure 7.** (**a**) Randomly selected correct classifications; (**b**) randomly selected incorrect classifications; (**c**) highest-confidence incorrect classifications for PathMNIST class 5 (smooth muscle). Training seed 42, selection seed 2026. Section 3.10 describes the image-order assumption. Confidence is the softmax output of the predicted class; it is not a measure of explanation quality. Conf. denotes the softmax probability of the predicted class; P(true) denotes the softmax probability assigned to the true class.

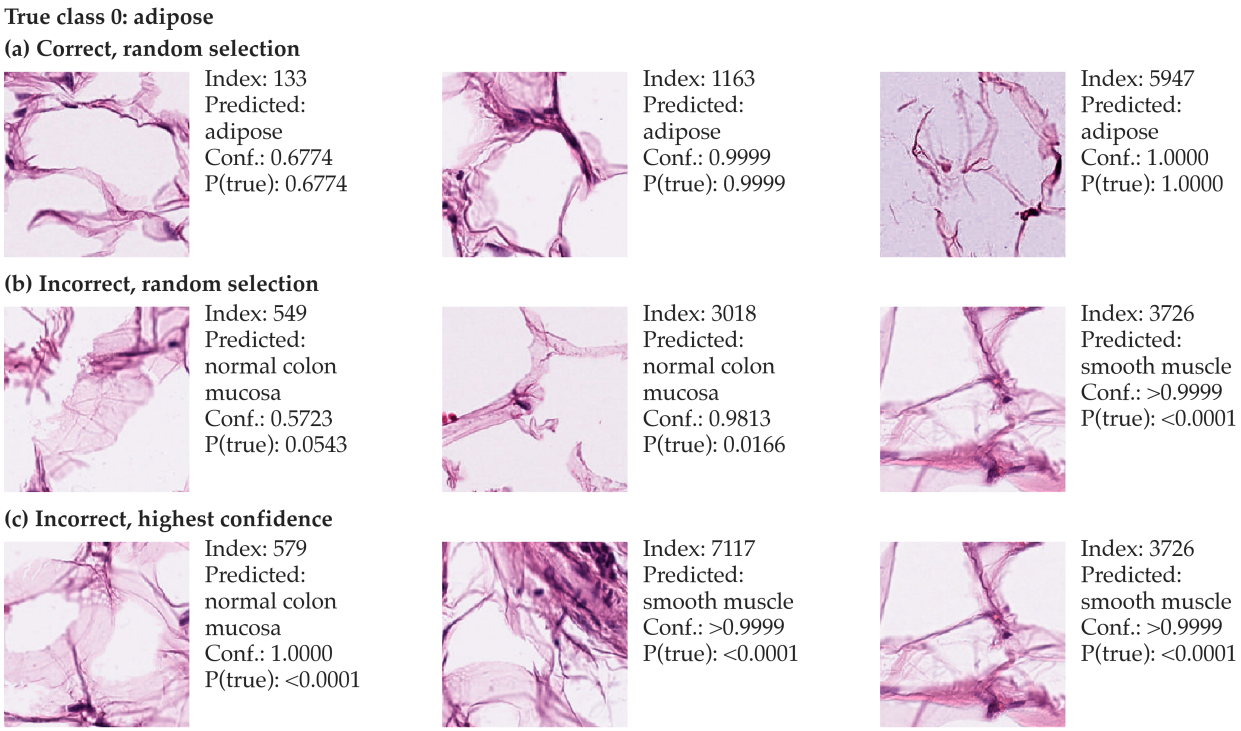



**Figure 8.** (**a**) Randomly selected correct classifications; (**b**) randomly selected incorrect classifications; (**c**) highest-confidence incorrect classifications for PathMNIST class 0 (adipose). Training seed 42, selection seed 2026. Conf. denotes the softmax probability of the predicted class; P(true) denotes the softmax probability assigned to the true class.

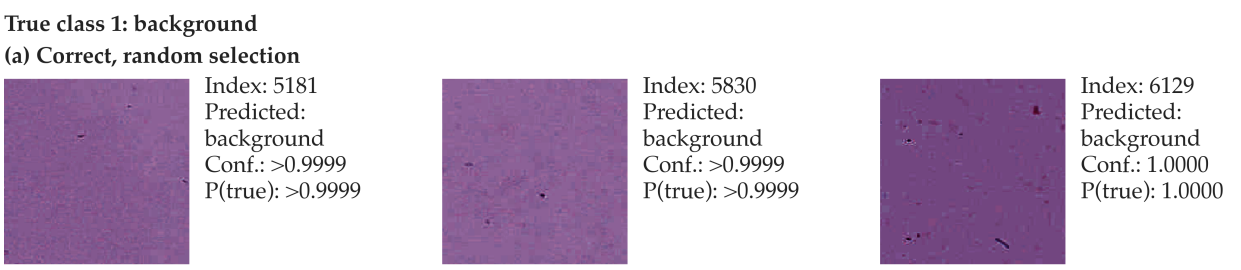



**Figure 9.** (**a**) Randomly selected correct classifications; (**b**) randomly selected incorrect classifications; (**c**) highest-confidence incorrect classifications for PathMNIST class 1 (background). Training seed 42, selection seed 2026. All 847 test examples in this class are correctly classified; there are no incorrect examples. Conf. denotes the softmax probability of the predicted class; P(true) denotes the softmax probability assigned to the true class.

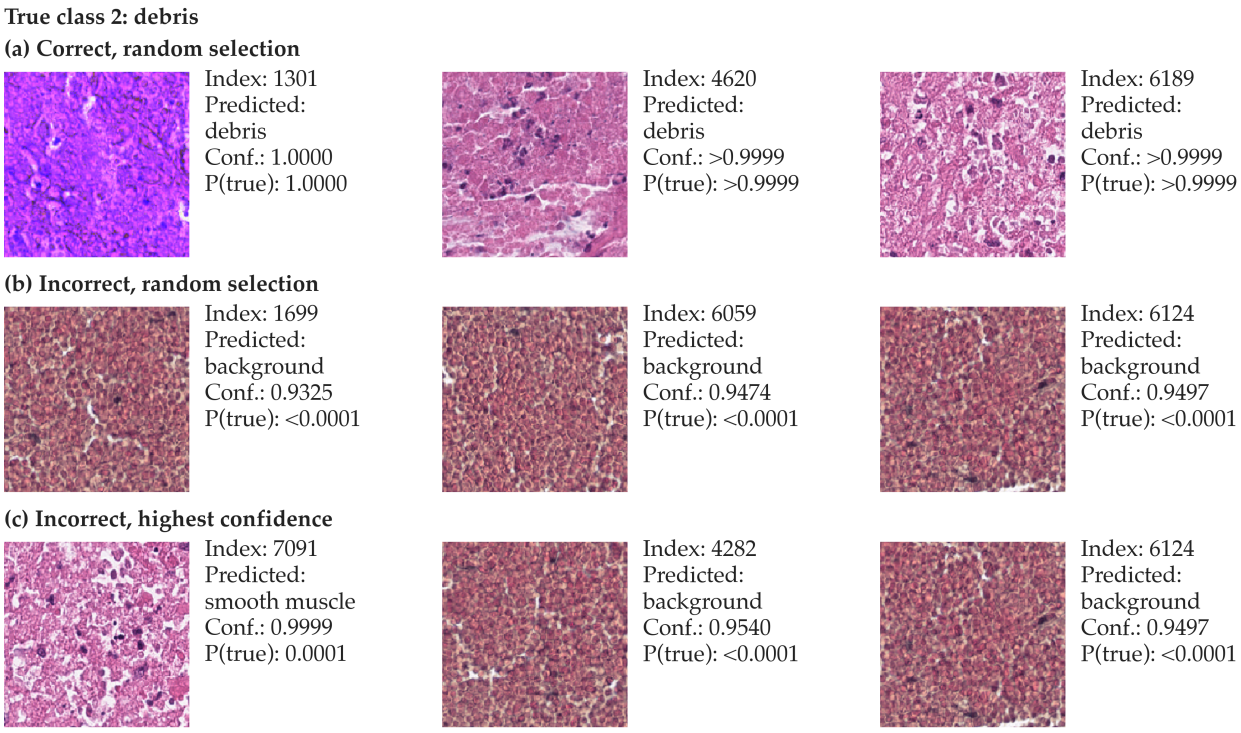



**Figure 10.** (**a**) Randomly selected correct classifications; (**b**) randomly selected incorrect classifications; (**c**) highest-confidence incorrect classifications for PathMNIST class 2 (debris). Training seed 42, selection seed 2026. Conf. denotes the softmax probability of the predicted class; P(true) denotes the softmax probability assigned to the true class.

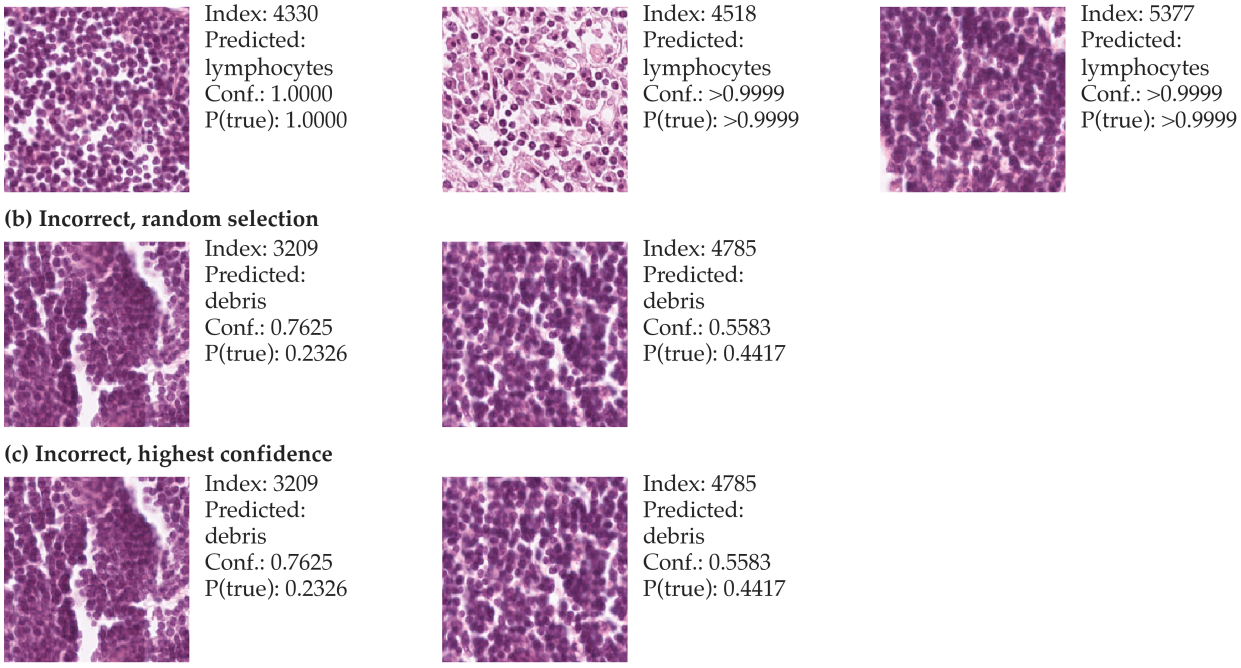



**Figure 11.** (**a**) Randomly selected correct classifications; (**b**) randomly selected incorrect classifications; (**c**) highest-confidence incorrect classifications for PathMNIST class 3 (lymphocytes). Training seed 42, selection seed 2026. There are only two incorrect examples in this class; the same two examples are shown in both error panels. Conf. denotes the softmax probability of the predicted class; P(true) denotes the softmax probability assigned to the true class.

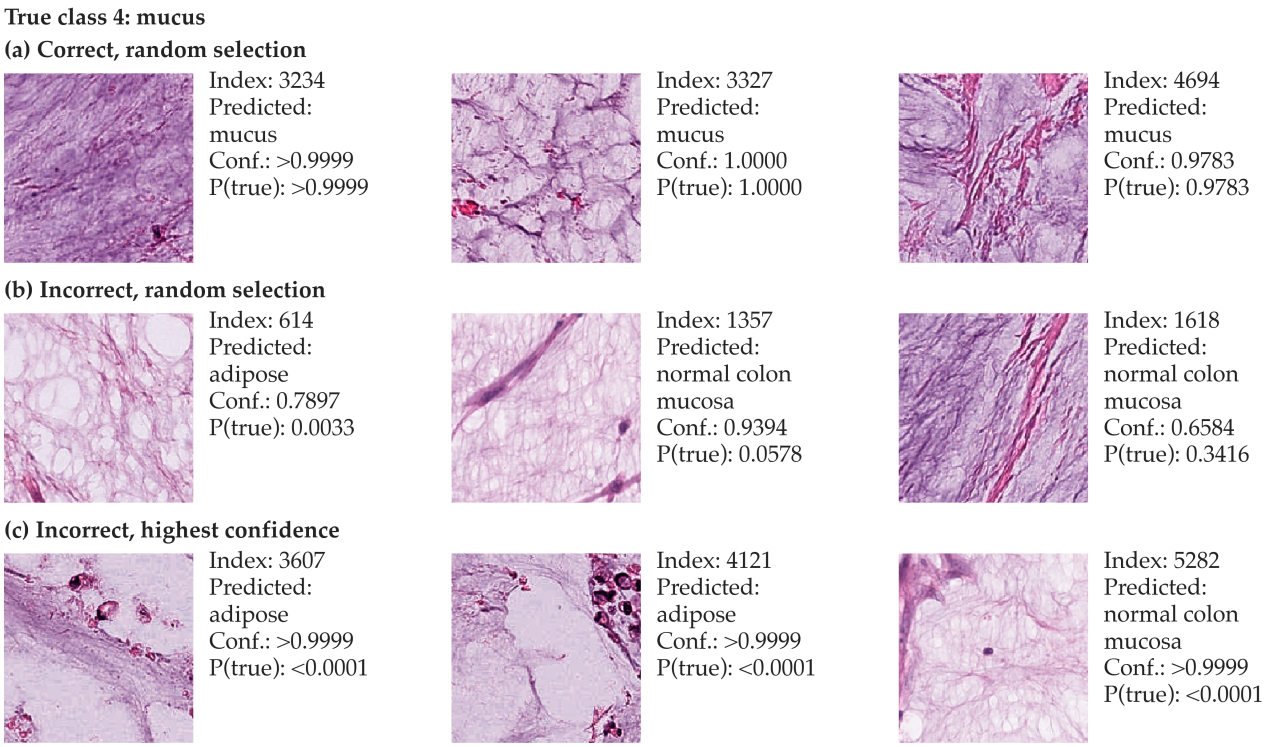



**Figure 12.** (**a**) Randomly selected correct classifications; (**b**) randomly selected incorrect classifications; (**c**) highest-confidence incorrect classifications for PathMNIST class 4 (mucus). Training seed 42, selection seed 2026. Conf. denotes the softmax probability of the predicted class; P(true) denotes the softmax probability assigned to the true class.

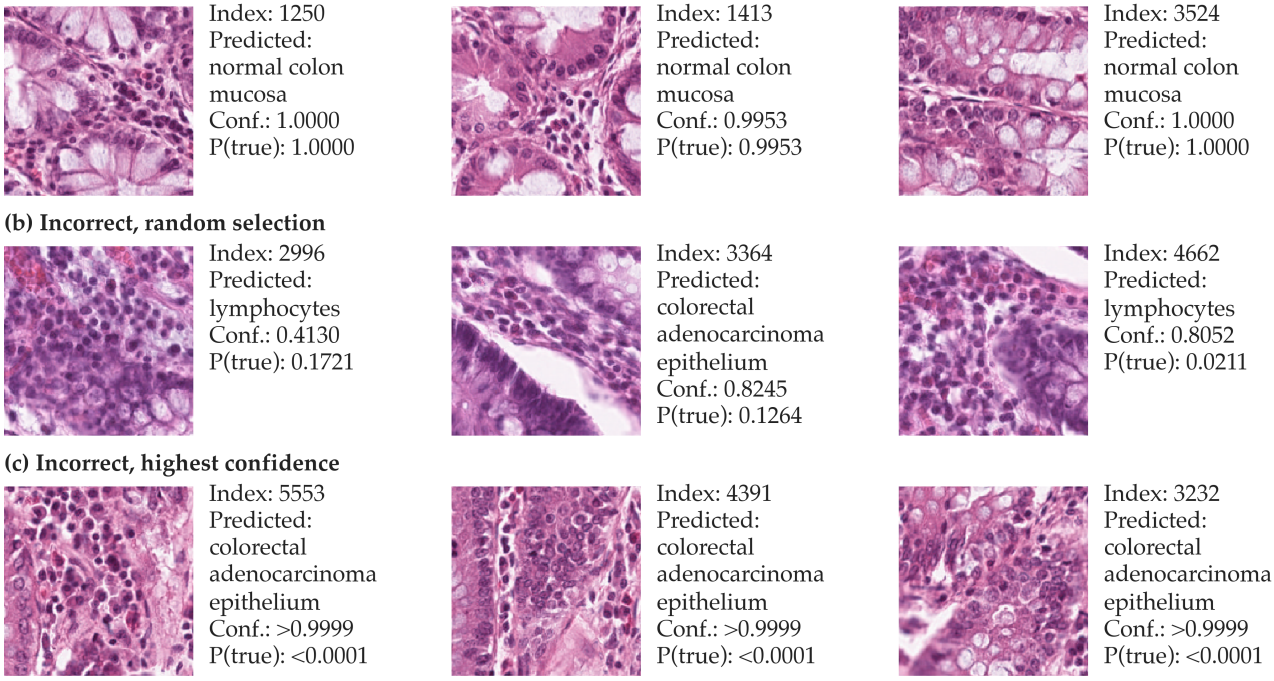



**Figure 13.** (**a**) Randomly selected correct classifications; (**b**) randomly selected incorrect classifications; (**c**) highest-confidence incorrect classifications for PathMNIST class 6 (normal colon mucosa). Training seed 42, selection seed 2026. Conf. denotes the softmax probability of the predicted class; P(true) denotes the softmax probability assigned to the true class.

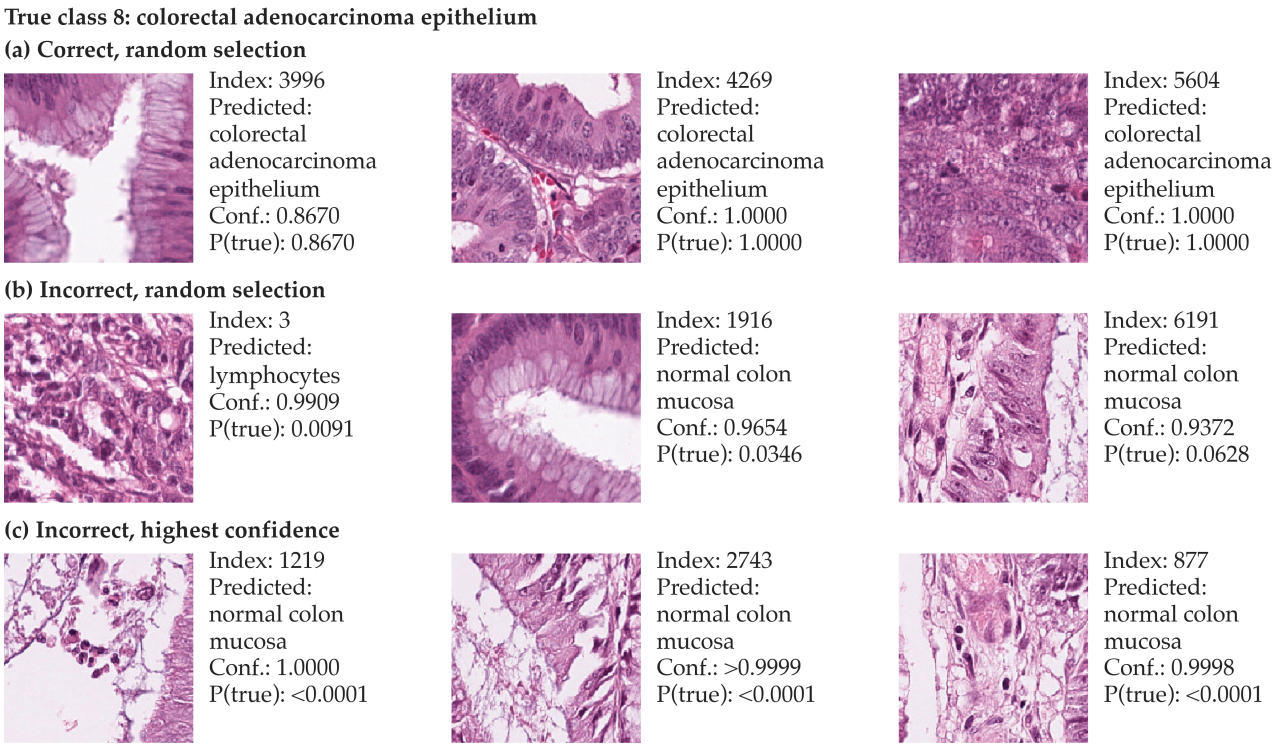



**Figure 14.** (**a**) Randomly selected correct classifications; (**b**) randomly selected incorrect classifications; (**c**) highest-confidence incorrect classifications for PathMNIST class 8 (colorectal adenocarcinoma epithelium). Training seed 42, selection seed 2026. Conf. denotes the softmax probability of the predicted class; P(true) denotes the softmax probability assigned to the true class.

### *4.2. Supplementary Five-Seed Rerun and Exploratory Sensitivity Analysis*

This subsection reports a separate, later experimental collection; its values are not pooled with the three-seed results. The collection comprises 14 configurations on BloodMNIST, PathMNIST, and DermaMNIST with five seeds (42, 52, 62, 72, and 82), giving the 210 training jobs validated by its completion record: SparseProto; six reduced variants (CNN-only, prototype-only, dense assignment, and removal of the auxiliary loss, the diversity loss, or both); an equal-budget CNN; an equal-budget CNN with local feature pooling; an equal-budget CNN distilled from a ResNet18 teacher; and ResNet18, ResNet50, MobileNetV3-Small, and EfficientNet-B0, with ResNet50 added as a larger-capacity reference. The seven runs of the $k$ and $M$ sweep reported at the end of this subsection are additional to these jobs.

The collection records PyTorch 2.11.0+cu130, CUDA 13.0, an NVIDIA RTX PRO 6000 Blackwell Server Edition GPU, and an Intel Xeon E5-2650 v4 (2.20 GHz). cuDNN benchmarking and TensorFloat-32 (TF32) arithmetic are disabled, deterministic cuDNN and PyTorch algorithms are enforced, and the CUBLAS workspace is fixed. Training uses the data files whose SHA-256 checksums are listed in Table 12, with no resolution fallback. During inference, four logged samples produced non-finite logits under FP16 autocast and were re-evaluated in FP32.

**Table 12.** SHA-256 checksums of the data files used in the five-seed collection.

| File | SHA-256 |
|---|---|
| `bloodmnist_224.npz` | `534bfb674fcfcfecf691aaca942ced2181d3affae9528a645ae51d12e9f3e1a1` |
| `pathmnist_224.npz` | `f167745634a90a47cd04e55aa69186ff0d0c4d5d1898045261ee12f83e22dd3e` |
| `dermamnist_224.npz` | `0143c9ae425867916d9d2a7769ed1a9b7b8c5eb9324a4bb384be6ef4d42da22d` |

Table 13 reports the five-seed results of all configurations.

**Table 13.** Five-seed results of all configurations (separate experimental collection).

| Dataset | Configuration | Parameters | Accuracy (%) | Macro F1 (%) | MCC |
|---|---|---|---|---|---|
| BloodMNIST | SparseProto (full) | 41,969 | 98.42 ± 0.07 | 98.55 ± 0.11 | 0.9816 ± 0.0009 |
| | CNN-only | 36,472 | 98.04 ± 0.30 | 98.09 ± 0.33 | 0.9771 ± 0.0034 |
| | Prototype-only | 41,009 | 98.17 ± 0.21 | 98.28 ± 0.15 | 0.9786 ± 0.0025 |
| | Dense assignment | 41,969 | 98.39 ± 0.06 | 98.52 ± 0.07 | 0.9812 ± 0.0007 |
| | No auxiliary loss | 41,969 | 98.15 ± 0.21 | 98.20 ± 0.22 | 0.9784 ± 0.0025 |
| | No diversity loss | 41,969 | 98.43 ± 0.16 | 98.53 ± 0.15 | 0.9817 ± 0.0019 |
| | No regularizers | 41,969 | 98.13 ± 0.20 | 98.16 ± 0.21 | 0.9782 ± 0.0024 |
| | Equal-budget CNN | 42,088 | 97.82 ± 0.36 | 97.82 ± 0.37 | 0.9746 ± 0.0042 |
| | Equal-budget local pooling | 41,993 | 98.14 ± 0.14 | 98.22 ± 0.16 | 0.9783 ± 0.0017 |
| | Distilled CNN | 42,088 | 98.08 ± 0.14 | 98.14 ± 0.12 | 0.9775 ± 0.0017 |
| | ResNet18 | 11,180,616 | 98.39 ± 0.12 | 98.51 ± 0.16 | 0.9812 ± 0.0014 |
| | ResNet50 | 23,524,424 | 98.24 ± 0.05 | 98.29 ± 0.06 | 0.9795 ± 0.0006 |
| | MobileNetV3-S | 1,526,056 | 98.45 ± 0.13 | 98.56 ± 0.16 | 0.9819 ± 0.0015 |
| | EfficientNet-B0 | 4,017,796 | 98.71 ± 0.20 | 98.79 ± 0.23 | 0.9850 ± 0.0024 |
| PathMNIST | SparseProto (full) | 42,759 | 88.20 ± 2.00 | 86.50 ± 1.89 | 0.8670 ± 0.0223 |
| | CNN-only | 36,569 | 90.41 ± 1.32 | 87.96 ± 1.45 | 0.8912 ± 0.0147 |
| | Prototype-only | 41,703 | 88.68 ± 1.71 | 86.77 ± 1.37 | 0.8726 ± 0.0181 |
| | Dense assignment | 42,759 | 88.90 ± 1.41 | 87.24 ± 1.14 | 0.8751 ± 0.0150 |
| | No auxiliary loss | 42,759 | 89.49 ± 1.34 | 87.43 ± 1.28 | 0.8813 ± 0.0145 |
| | No diversity loss | 42,759 | 89.84 ± 1.33 | 87.95 ± 1.19 | 0.8853 ± 0.0144 |
| | No regularizers | 42,759 | 89.81 ± 1.09 | 87.68 ± 0.90 | 0.8849 ± 0.0121 |
| | Equal-budget CNN | 42,582 | 90.59 ± 1.02 | 88.36 ± 0.79 | 0.8931 ± 0.0110 |
| | Equal-budget local pooling | 42,597 | 91.30 ± 1.93 | 88.85 ± 1.78 | 0.9014 ± 0.0214 |
| | Distilled CNN | 42,582 | 89.95 ± 1.51 | 87.99 ± 1.50 | 0.8862 ± 0.0167 |
| | ResNet18 | 11,181,129 | 92.02 ± 0.51 | 90.31 ± 0.57 | 0.9097 ± 0.0056 |
| | ResNet50 | 23,526,473 | 91.09 ± 1.07 | 89.67 ± 1.06 | 0.8998 ± 0.0115 |
| | MobileNetV3-S | 1,527,081 | 92.40 ± 1.22 | 89.83 ± 1.50 | 0.9136 ± 0.0135 |
| | EfficientNet-B0 | 4,019,077 | 92.37 ± 1.70 | 90.52 ± 1.60 | 0.9138 ± 0.0185 |
| DermaMNIST | SparseProto (full) | 41,187 | 74.81 ± 0.41 | 48.37 ± 4.04 | 0.4789 ± 0.0169 |
| | CNN-only | 36,375 | 74.37 ± 1.32 | 48.46 ± 2.39 | 0.5005 ± 0.0209 |
| | Prototype-only | 40,323 | 75.53 ± 0.75 | 46.93 ± 2.45 | 0.5106 ± 0.0150 |
| | Dense assignment | 41,187 | 74.43 ± 0.80 | 47.31 ± 2.98 | 0.4740 ± 0.0128 |
| | No auxiliary loss | 41,187 | 74.58 ± 0.42 | 46.15 ± 1.77 | 0.4779 ± 0.0212 |
| | No diversity loss | 41,187 | 74.62 ± 0.50 | 47.61 ± 4.49 | 0.4723 ± 0.0198 |
| | No regularizers | 41,187 | 74.38 ± 0.70 | 46.03 ± 1.82 | 0.4752 ± 0.0226 |
| | Equal-budget CNN | 41,223 | 74.21 ± 0.77 | 47.12 ± 4.50 | 0.4732 ± 0.0375 |
| | Equal-budget local pooling | 41,016 | 74.56 ± 0.98 | 46.66 ± 3.17 | 0.4710 ± 0.0356 |
| | Distilled CNN | 41,223 | 75.75 ± 1.00 | 52.01 ± 2.81 | 0.5189 ± 0.0335 |
| | ResNet18 | 11,180,103 | 75.53 ± 1.00 | 49.59 ± 2.30 | 0.5145 ± 0.0197 |
| | ResNet50 | 23,522,375 | 75.00 ± 0.33 | 46.72 ± 2.38 | 0.4789 ± 0.0124 |
| | MobileNetV3-S | 1,525,031 | 72.20 ± 1.44 | 44.34 ± 4.41 | 0.4465 ± 0.0389 |
| | EfficientNet-B0 | 4,016,515 | 74.89 ± 1.38 | 44.61 ± 5.15 | 0.4930 ± 0.0198 |

Mean ± sample standard deviation over five seeds.

The five-seed collection does not support universal accuracy superiority. On PathMNIST, SparseProto attains 88.20% mean accuracy versus 92.37% for EfficientNet-B0 and 92.40% for MobileNetV3-Small. On DermaMNIST, ResNet18 achieves a higher mean macro-F1 (49.59%) than SparseProto (48.37%). Accordingly, the compact architecture is presented as a performance–resource trade-off, not a state-of-the-art classifier.

Table 14 reports the paired differences. On BloodMNIST the full model exceeds the CNN-only variant by 0.38 points (hierarchical 95% interval 0.03 to 0.75); on PathMNIST it is 2.21 points lower (−4.52 to −0.68); on DermaMNIST the interval includes zero. The comparison with the equal-budget CNN follows the same pattern. Seed-level *t*-intervals are

wider: for the CNN-only comparison they are 0.01 to 0.75, −5.35 to 0.92, and −1.22 to 2.10 points on the three datasets, respectively.

**Table 14.** Paired accuracy differences in the five-seed collection (full SparseProto minus comparator, percentage points).

| Comparator | BloodMNIST | PathMNIST | DermaMNIST |
|---|---|---|---|
| CNN-only | +0.38 [0.03, 0.75] | −2.21 [−4.52, −0.68] | +0.44 [−1.00, 1.93] |
| Prototype-only | +0.25 [−0.04, 0.56] | −0.48 [−2.28, 1.50] | −0.72 [−1.89, 0.52] |
| Dense assignment | +0.03 [−0.16, 0.23] | −0.70 [−2.54, 0.70] | +0.38 [−0.55, 1.41] |
| No auxiliary loss | +0.27 [−0.05, 0.60] | −1.29 [−2.35, −0.33] | +0.23 [−0.56, 1.14] |
| No diversity loss | −0.01 [−0.22, 0.20] | −1.65 [−3.75, −0.02] | +0.19 [−0.16, 0.71] |
| No regularizers | +0.29 [−0.02, 0.61] | −1.61 [−3.91, 0.49] | +0.43 [−0.50, 1.46] |
| Equal-budget CNN | +0.60 [0.18, 1.03] | −2.39 [−4.67, −0.52] | +0.60 [−0.72, 1.86] |
| Equal-budget local pooling | +0.28 [−0.01, 0.56] | −3.10 [−5.59, −0.50] | +0.25 [−0.94, 1.58] |
| Distilled CNN | +0.34 [0.04, 0.68] | −1.75 [−3.65, 0.13] | −0.94 [−2.20, 0.34] |
| ResNet18 | +0.04 [−0.30, 0.39] | −3.83 [−5.38, −2.26] | −0.72 [−2.09, 0.80] |
| ResNet50 | +0.18 [−0.13, 0.50] | −2.89 [−4.73, −1.00] | −0.19 [−1.37, 1.02] |
| MobileNetV3-S | −0.03 [−0.34, 0.27] | −4.21 [−5.61, −2.49] | +2.61 [1.14, 4.11] |
| EfficientNet-B0 | −0.29 [−0.61, 0.04] | −4.17 [−6.64, −1.67] | −0.08 [−1.72, 1.71] |

Mean difference [hierarchical bootstrap 95% interval]. The bootstrap resamples matched seed pairs and shared test indices (2000 resamples); it is exploratory and does not represent independent patients. No equivalence claim is made for this collection.

Table 15 reports the latency measurement of this collection for PathMNIST. Each value summarizes three independent sessions of 30 timed forward passes after 12 warm-up passes, with synchronization around each GPU pass and newly initialized weights. On PathMNIST at batch size 1 and FP32, SparseProto is 2.45 times slower than CNN-only on GPU (1.269/0.519 ms) and 1.17 times slower on CPU (1.970/1.680 ms). In this condition it is faster than ResNet18, MobileNetV3-Small, and EfficientNet-B0. Under GPU FP16, however, ResNet18 is faster (1.026 versus 1.502 ms); at CPU batch size 32, MobileNetV3-Small is faster (20.21 versus 26.14 ms). The GPU FP16 comparison reverses the ordering in Table 5 and shows that the ordering depends on device and precision. Table 15 is not comparable with Table 5, which was obtained on different hardware. The recorded profiling output reports operator times and not memory traffic, so it does not test the memory-traffic hypothesis stated above.

**Table 15.** Latency in the five-seed collection on PathMNIST (ms per forward pass).

| Configuration | GPU FP32, Batch 1 | GPU FP16, Batch 1 | GPU FP32, Batch 64 | CPU FP32, Batch 1 | CPU FP32, Batch 32 |
|---|---|---|---|---|---|
| SparseProto (full) | 1.269 ± 0.005 | 1.502 ± 0.004 | 1.899 ± 0.002 | 1.970 ± 0.010 | 26.14 ± 0.16 |
| CNN-only | 0.519 ± 0.001 | 0.673 ± 0.001 | 1.187 ± 0.005 | 1.680 ± 0.003 | 24.26 ± 0.11 |
| Equal-budget CNN | 0.503 ± 0.001 | 0.651 ± 0.001 | 1.194 ± 0.004 | 1.764 ± 0.010 | 24.42 ± 0.12 |
| Equal-budget local pooling | 0.561 ± 0.003 | 0.748 ± 0.002 | 1.242 ± 0.004 | 1.853 ± 0.011 | 24.62 ± 0.03 |
| Distilled CNN | 0.502 ± 0.001 | 0.657 ± 0.003 | 1.192 ± 0.005 | 1.796 ± 0.003 | 25.37 ± 0.05 |
| ResNet18 | 1.345 ± 0.016 | 1.026 ± 0.001 | 5.479 ± 0.002 | 6.400 ± 0.017 | 72.75 ± 0.43 |
| ResNet50 | 3.825 ± 0.001 | 2.295 ± 0.003 | 22.457 ± 0.010 | 16.652 ± 0.070 | 323.79 ± 4.48 |
| MobileNetV3-S | 1.601 ± 0.002 | 2.214 ± 0.003 | 2.473 ± 0.010 | 4.155 ± 0.004 | 20.21 ± 0.21 |
| EfficientNet-B0 | 2.521 ± 0.001 | 3.209 ± 0.002 | 11.576 ± 0.030 | 9.917 ± 0.013 | 147.22 ± 0.72 |

Mean ± standard deviation of the three session means. Values are per forward pass, not per image; divide by the batch size for the time per image.

The seven $k$ and $M$ settings in Table 16 are exploratory, and each uses only seed 42. Although $k = 4$, $M = 8$ yields the highest test accuracy and macro-F1 among these runs, test-set performance was not used to select or retune the primary architecture. See the Limitations section for the scope of the comparisons.

**Table 16.** Exploratory PathMNIST top-$k$ and prototypes-per-class sweep, seed 42 only.

| Run | $k$ | $M$ | Accuracy (%) | Macro F1 (%) |
|---|---|---|---|---|
| S1 | 1 | 4 | 89.71 | 87.81 |
| S2 | 2 | 4 | 89.79 | 87.65 |
| S3 | 4 | 4 | 90.15 | 87.94 |
| S4 | 8 | 4 | 89.79 | 87.67 |
| S5 | 4 | 1 | 88.89 | 86.81 |
| S6 | 4 | 2 | 89.47 | 87.63 |
| S7 | 4 | 8 | 91.03 | 88.70 |

## 5. Discussion

SparseProto was the fastest of the four architectures in the original measurements, while the five-seed measurements show that latency rankings depend on device, precision, and batch size. For PathMNIST, ResNet18 has 261.5 times as many parameters and 53.9 times as many counted operations as SparseProto, but takes only 3.87 times as long on CPU and 1.14 times as long on GPU in the original measurements. Memory access and platform characteristics help explain why operation counts and latency diverge [11].

The five-seed results show dataset-dependent accuracy differences and higher PathMNIST latency than CNN-only (Tables 14 and 15). Implementation verification identifies the full token-to-prototype similarity matrix, materialized before top-$k$ selection, as the main target for a redesign aimed at reducing the cost of the assignment.

*Limitations*

The three-seed archive lacks file checksums and image identifiers; cached-file checks establish only presence and non-zero size, so an incorrect or outdated file cannot be excluded. Resolution fallback means that $224 \times 224$ inputs do not establish use of the 224-resolution release. The five-seed collection instead records checksums and disables fallback. Patient, lesion, and specimen identifiers are absent, preventing verification of split independence; an external-center PathMNIST test set does not verify the provenance of the archived run. DermaMNIST also inherits documented data-quality concerns [31]. A traceable evaluation requires image identifiers for each split and grouping identifiers where available. The qualitative panels rely on the original test-set ordering because the image-identifier manifest was not retained. Label agreement alone cannot detect substitutions within a class.

Seeding alone does not ensure bit-level reproducibility [48]. The three-seed checkpoints omit random-generator states, so resumed runs can differ from uninterrupted runs. Reused results and checkpoint configurations are not checked against the active implementation; outputs from earlier settings cannot be distinguished reliably. The unresolved discrepancy between main and ablation full-model records further limits comparisons between collections.

Three seeds provide imprecise estimates of training variability, and five seeds only partly address this uncertainty. Sample-level inference in the original archive covers the ablation collection; main-comparator tests rely on seed-level means. Archived validation probabilities were unavailable, so validation metrics could not be independently recomputed. Small DermaMNIST classes and absent grouping identifiers limit inference about new patients or specimens. Replicate counts should follow a prospective power analysis, and future evaluation should use patient-, lesion-, or specimen-level sampling. See Table 17.

Equal-budget CNN, local-pooling, and distillation controls are available only in the five-seed collection. The experiments did not include pruning or matched comparisons with ProtoPNet, PIP-Net, ProtoTree, medical prototype models, or NetVLAD-type aggre-

gation [21,22,24–26,36,41]. The baselines share the data stream, epoch budget, optimizer, and initial learning rate. Training from scratch can affect architectures differently [39]. CNN-only changes both the representation path and the loss by omitting the auxiliary and diversity terms, so the comparison estimates the joint effect of these changes.

**Table 17.** Records contained in the author-maintained research archive.

| Collection | Runs | Summary Statistics | Per-Sample Outputs | Epoch-Level History | Checkpoints |
|---|---|---|---|---|---|
| Main comparison (4 models × 3 datasets × 3 seeds) | 36 | Yes | Yes | Yes | Yes |
| Ablation (5 recorded configurations × 3 datasets × 3 seeds) | 45 | Yes | Yes | Yes | Yes |
| Hyperparameter sensitivity (PathMNIST, seed 42) | 8 records, 7 unique settings | Validation only | No | Yes | Yes |
| Efficiency measurements (4 models × 3 datasets) | 12 | Yes | Not applicable | Not applicable | Not applicable |

Per-sample outputs comprise the test probabilities, predicted and true labels, confusion matrix, and per-class report. Summary statistics are the per-run rows of the collection-level tables. The 45 ablation runs include nine runs of a redundant configuration that does not change the computation and is not reported in Table 7.

The heap-based selection bound in Equation (12) is illustrative and has not been verified for the library implementation. The operation counter excludes top-$k$ selection, softmax, scatter, normalization, and associated activations; its totals therefore describe covered operations rather than complete computational cost [47]. Sparse assignment follows dense similarity computation. Each latency collection uses one host, and model ordering changes with hardware, precision, and batch size. Only one mean per model/device was retained for the original measurement, preventing estimation of its standard deviation; the profiling record does not measure memory traffic. Embedded-device performance requires direct measurement [10,11].

High-confidence errors and qualitative panels do not establish explanation quality. Without similarity maps, localization, stability, consistency tests, or expert assessment, prototype interpretability remains unverified [15,16,45]. Future work should count selected top-$k$ indices directly to distinguish rounding from selection frequency and test a formulation that avoids materializing the dense similarity matrix.

## 6. Conclusions

SparseProto combines global convolutional features with a top-$k$ prototype histogram using approximately 41–43 thousand parameters. In the five-seed collection, its mean accuracy is 98.42% on BloodMNIST, 88.20% on PathMNIST, and 74.81% on DermaMNIST. It trails the highest-accuracy large comparator by 0.29, 4.20, and 0.72 percentage points, respectively: EfficientNet-B0, MobileNetV3-Small, and ResNet18. Parameter compactness therefore comes with dataset-dependent predictive trade-offs. See Table 13.

The five-seed collection does not establish a universal latency advantage. On PathMNIST, ResNet18 is faster under GPU FP16 at batch size 1 (1.026 versus 1.502 ms per pass), and MobileNetV3-Small is faster on CPU FP32 at batch size 32 (20.21 versus 26.14 ms). At batch size 1 and FP32, SparseProto is faster than these large comparators but slower than CNN-only by factors of 2.45 on GPU and 1.17 on CPU. Device, numerical precision, and batch size must therefore accompany any efficiency claim. See Table 15.

The full configuration shows no consistent accuracy advantage over CNN-only (Table 14). Future work should isolate architectural and loss changes, verify data provenance, and measure complete computational cost.

**Funding:** This research received no external funding.

**Data Availability Statement:** The source datasets, BloodMNIST, PathMNIST, and DermaMNIST, are publicly available through the MedMNIST v2 distribution at https://medmnist.com/ [29]. The three-seed research archive and accompanying analysis script are available from the author upon reasonable request. The five-seed collection described in Section 4.2 is a separate set of records. The archive contains per-sample test probabilities and predictions for each of the 36 main and 45 ablation runs, together with training histories, checkpoints, per-class reports, and the summary tables of the main comparison, the efficiency measurements, and the hyperparameter sensitivity study. All test metrics of the ablation collection were recomputed from the stored probabilities and agree with the archived values to within $1.1 \times 10^{-16}$; the stored predictions coincide with the arg max of the stored probabilities in all 45 runs, and the probability rows sum to unity to within $2.4 \times 10^{-7}$. Table 17 sets out, for each collection, which records the archive contains. The sweep comprises eight records corresponding to seven unique settings, because the configuration $k = 4$, $M = 4$ is repeated in both the $k$ series and the $M$ series. The qualitative panels and the class-level counts of Table 11 derive from the per-sample records of the main PathMNIST run with seed 42. The prototype-usage values of Table 10 derive from the per-run usage records of the main comparison, and the DermaMNIST class-level values derive from its per-class reports. The accompanying analysis script enables recomputation of the results from the 45 ablation runs, paired comparisons and equivalence assessments reported in Table 8, resource ratios, and class-level confidence intervals based on the counts in Table 11. The bootstrap is seeded, and one resample grid of the test set is shared by all comparisons within a dataset, so the intervals are reproducible exactly. Appendix A gives the reconstruction limits and summarizes the notation.

**Acknowledgments:** During the preparation of this manuscript, the author used ChatGPT (GPT-5; OpenAI) to assist with language translation, academic phrasing, and the revision of written expression. The original research methodology, its implementation, the conduct of the experiments, and the generation, analysis, and interpretation of the results were undertaken by the author. The author retains full responsibility for the accuracy, originality, integrity, and scientific content of the manuscript.

**Conflicts of Interest:** The author declares no conflicts of interest.

## Abbreviations

| | |
|---|---|
| ACC | Accuracy |
| AI | Artificial intelligence |
| AUC | Area under the receiver operating characteristic curve |
| CBA | Convolution, batch normalization, and activation |
| CI | Confidence interval |
| CNN | Convolutional neural network |
| CPU | Central processing unit |
| CUDA | Compute Unified Device Architecture |
| DS | Depthwise separable |
| ECE | Expected calibration error |
| FLOPs | Floating-point operations |

| | |
|---|---|
| FP32 | 32-bit floating-point precision |
| GenAI | Generative artificial intelligence |
| GPU | Graphics processing unit |
| LN | Layer normalization |
| MCC | Matthews correlation coefficient |
| pp | Percentage points |
| SiLU | Sigmoid linear unit |
| TF32 | TensorFloat-32 |

## Appendix A. Interpretation and Reproducibility of the Reported Analyses

### *Appendix A.1. Available Evidence and Reconstruction Limits*

The inventory in Table 17 lists the records available for each collection. Recomputing intervals from the class counts reported in Table 11 is possible without reconstructing individual predictions. Reproducing the qualitative panels, however, requires the corresponding images, prediction records, and their ordering. A complete reproducibility package should identify the code revision, dependency versions, dataset release and checksums, preprocessing settings, seeds, and any archived checkpoints or resumed runs.

### *Appendix A.2. Notation*

Table A1 summarizes the principal notation used in the model and evaluation.

**Table A1.** Principal notation used in the model and evaluation.

| Symbol | Definition |
|---|---|
| $B$ | Minibatch size |
| $C$ | Number of target classes |
| $d$ | Token dimension; $d = 96$ in the reported architecture |
| $H, W$ | Spatial dimensions of the final feature map |
| $L$ | Number of spatial tokens; $L = HW = 196$ |
| $M$ | Prototypes per class used to determine the bank size |
| $P$ | Number of prototypes; $P = CM$ |
| $k$ | Number of selected prototypes per token |
| $S_{\ell p}$ | Cosine similarity between token $\ell$ and prototype $p$ |
| $A_{\ell p}$ | Assignment weight for token $\ell$ and prototype $p$ |
| $h$ | Image-level prototype-assignment histogram |
| $g$ | Global feature vector |
| $q$ | Projected 48-dimensional prototype feature vector |
| $\tau$ | Learnable, clipped similarity multiplier |

## References

1. van der Laak, J.; Litjens, G.; Ciompi, F. Deep learning in histopathology: The path to the clinic. *Nat. Med.* **2021**, *27*, 775–784. https://doi.org/10.1038/s41591-021-01343-4.
2. Bain, B.J. Diagnosis from the blood smear. *N. Engl. J. Med.* **2005**, *353*, 498–507. https://doi.org/10.1056/NEJMra043442.
3. Kittler, H.; Pehamberger, H.; Wolff, K.; Binder, M. Diagnostic accuracy of dermoscopy. *Lancet Oncol.* **2002**, *3*, 159–165. https://doi.org/10.1016/S1470-2045(02)00679-4.
4. Kather, J.N.; Krisam, J.; Charoentong, P.; Luedde, T.; Herpel, E.; Weis, C.A.; Gaiser, T.; Marx, A.; Valous, N.A.; Ferber, D.; et al. Predicting survival from colorectal cancer histology slides using deep learning: A retrospective multicenter study. *PLoS Med.* **2019**, *16*, e1002730. https://doi.org/10.1371/journal.pmed.1002730.
5. Elmore, J.G.; Longton, G.M.; Carney, P.A.; Geller, B.M.; Onega, T.; Tosteson, A.N.A.; Nelson, H.D.; Pepe, M.S.; Allison, K.H.; Schnitt, S.J.; et al. Diagnostic concordance among pathologists interpreting breast biopsy specimens. *JAMA* **2015**, *313*, 1122–1132. https://doi.org/10.1001/jama.2015.1405.

6. Acevedo, A.; Merino, A.; Alférez, S.; Molina, Á.; Boldú, L.; Rodellar, J. A dataset of microscopic peripheral blood cell images for development of automatic recognition systems. *Data Brief* **2020**, *30*, 105474. https://doi.org/10.1016/j.dib.2020.105474.
7. Tschandl, P.; Rosendahl, C.; Kittler, H. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Sci. Data* **2018**, *5*, 180161. https://doi.org/10.1038/sdata.2018.161.
8. Litjens, G.; Kooi, T.; Bejnordi, B.E.; Setio, A.A.A.; Ciompi, F.; Ghafoorian, M.; van der Laak, J.A.W.M.; van Ginneken, B.; Sánchez, C.I. A survey on deep learning in medical image analysis. *Med. Image Anal.* **2017**, *42*, 60–88. https://doi.org/10.1016/j.media.2017.07.005.
9. Esteva, A.; Kuprel, B.; Novoa, R.A.; Ko, J.; Swetter, S.M.; Blau, H.M.; Thrun, S. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **2017**, *542*, 115–118. https://doi.org/10.1038/nature21056.
10. Ngo, D.; Park, H.C.; Kang, B. Edge Intelligence: A Review of Deep Neural Network Inference in Resource-Limited Environments. *Electronics* **2025**, *14*, 2495. https://doi.org/10.3390/electronics14122495.
11. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In *Computer Vision—ECCV 2018*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2018; Volume 11218, pp. 122–138. https://doi.org/10.1007/978-3-030-01264-9_8.
12. Johnson, J.M.; Khoshgoftaar, T.M. Survey on deep learning with class imbalance. *J. Big Data* **2019**, *6*, 27. https://doi.org/10.1186/s40537-019-0192-5.
13. Reinke, A.; Tizabi, M.D.; Baumgartner, M.; Eisenmann, M.; Heckmann-Nötzel, D.; Kavur, A.E.; Rädsch, T.; Sudre, C.H.; Acion, L.; Antonelli, M.; et al. Understanding metric-related pitfalls in image analysis validation. *Nat. Methods* **2024**, *21*, 182–194. https://doi.org/10.1038/s41592-023-02150-0.
14. Maier-Hein, L.; Reinke, A.; Godau, P.; Tizabi, M.D.; Buettner, F.; Christodoulou, E.; Glocker, B.; Isensee, F.; Kleesiek, J.; Kozubek, M.; et al. Metrics reloaded: Recommendations for image analysis validation. *Nat. Methods* **2024**, *21*, 195–212. https://doi.org/10.1038/s41592-023-02151-z.
15. Carmichael, Z.; Lohit, S.; Cherian, A.; Jones, M.J.; Scheirer, W.J. Pixel-Grounded Prototypical Part Networks. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV 2024), Waikoloa, HI, USA, 3–8 January 2024; pp. 4768–4779. https://doi.org/10.1109/WACV57701.2024.00470.
16. Davoodi, O.; Mohammadizadehsamakosh, S.; Komeili, M. On the interpretability of part-prototype based classifiers: A human centric analysis. *Sci. Rep.* **2023**, *13*, 23088. https://doi.org/10.1038/s41598-023-49854-z.
17. Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.* **2019**, *1*, 206–215. https://doi.org/10.1038/s42256-019-0048-x.
18. Bouthillier, X.; Delaunay, P.; Bronzi, M.; Trofimov, A.; Nichyporuk, B.; Szeto, J.; Mohammadi Sepahvand, N.; Raff, E.; Madan, K.; Voleti, V.; et al. Accounting for Variance in Machine Learning Benchmarks. In Proceedings of the Conference on Machine Learning and Systems (MLSys 2021), Virtual, 5–9 April 2021; Volume 3, pp. 747–769.
19. Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
20. Kapoor, S.; Narayanan, A. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns* **2023**, *4*, 100804. https://doi.org/10.1016/j.patter.2023.100804.
21. Chen, C.; Li, O.; Tao, C.; Barnett, A.J.; Su, J.K.; Rudin, C. This Looks Like That: Deep Learning for Interpretable Image Recognition. In Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019; Volume 32, pp. 8930–8941.
22. Nauta, M.; van Bree, R.; Seifert, C. Neural Prototype Trees for Interpretable Fine-Grained Image Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2021), Virtual, 19–25 June 2021; pp. 14933–14943. https://doi.org/10.1109/CVPR46437.2021.01469.
23. Donnelly, J.; Barnett, A.J.; Chen, C. Deformable ProtoPNet: An Interpretable Image Classifier Using Deformable Prototypes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2022), New Orleans, LA, USA, 18–24 June 2022; pp. 10265–10275. https://doi.org/10.1109/CVPR52688.2022.01002.
24. Nauta, M.; Schlötterer, J.; van Keulen, M.; Seifert, C. PIP-Net: Patch-Based Intuitive Prototypes for Interpretable Image Classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2023), Vancouver, BC, Canada, 17–24 June 2023; pp. 2744–2753. https://doi.org/10.1109/CVPR52729.2023.00269.
25. Choukali, M.A.; Amirani, M.C.; Valizadeh, M.; Abbasi, A.; Komeili, M. Pseudo-class part prototype networks for interpretable breast cancer classification. *Sci. Rep.* **2024**, *14*, 10341. https://doi.org/10.1038/s41598-024-60743-x.
26. Gallée, L.; Lisson, C.S.; Ropinski, T.; Beer, M.; Götz, M. Proto-Caps: Interpretable medical image classification using prototype learning and privileged information. *PeerJ Comput. Sci.* **2025**, *11*, e2908. https://doi.org/10.7717/peerj-cs.2908.
27. Doerrich, S.; Di Salvo, F.; Brockmann, J.; Ledig, C. Rethinking model prototyping through the MedMNIST+ dataset collection. *Sci. Rep.* **2025**, *15*, 7669. https://doi.org/10.1038/s41598-025-92156-9.
28. Prvan, M.; Musić, J.; Čoko, D.; Kristić, A. Lightweight Neural Network Ensemble Models for Medical Image Classification with MedMNIST Dataset. *Electronics* **2026**, *15*, 1470. https://doi.org/10.3390/electronics15071470.

29. Yang, J.; Shi, R.; Wei, D.; Liu, Z.; Zhao, L.; Ke, B.; Pfister, H.; Ni, B. MedMNIST v2—A large-scale lightweight benchmark for 2D and 3D biomedical image classification. *Sci. Data* **2023**, *10*, 41. https://doi.org/10.1038/s41597-022-01721-8.
30. Lakens, D. Equivalence Tests: A Practical Primer for t Tests, Correlations, and Meta-Analyses. *Soc. Psychol. Personal. Sci.* **2017**, *8*, 355–362. https://doi.org/10.1177/1948550617697177.
31. Abhishek, K.; Jain, A.; Hamarneh, G. Investigating the quality of DermaMNIST and Fitzpatrick17k dermatological image datasets. *Sci. Data* **2025**, *12*, 196. https://doi.org/10.1038/s41597-025-04382-5.
32. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. https://doi.org/10.1109/CVPR.2016.90.
33. Howard, A.; Sandler, M.; Chu, G.; Chen, L.C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; et al. Searching for MobileNetV3. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV 2019), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324. https://doi.org/10.1109/ICCV.2019.00140.
34. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the 36th International Conference on Machine Learning (ICML 2019), Long Beach, CA, USA, 9–15 June 2019; Proceedings of Machine Learning Research, Volume 97, pp. 6105–6114.
35. Schwartz, R.; Dodge, J.; Smith, N.A.; Etzioni, O. Green AI. *Commun. ACM* **2020**, *63*, 54–63. https://doi.org/10.1145/3381831.
36. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge Distillation: A Survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. https://doi.org/10.1007/s11263-021-01453-z.
37. Xue, M.; Huang, Q.; Zhang, H.; Hu, J.; Song, J.; Song, M.; Jin, C. ProtoPFormer: Concentrating on Prototypical Parts in Vision Transformers for Interpretable Image Recognition. In Proceedings of the 33rd International Joint Conference on Artificial Intelligence (IJCAI 2024), Jeju, Republic of Korea, 3–9 August 2024; pp. 1516–1524. https://doi.org/10.24963/ijcai.2024/168.
38. Zhu, Z.; Fan, L.; Pagnucco, M.; Song, Y. Interpretable Image Classification via Non-parametric Part Prototype Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2025), Nashville, TN, USA, 11–15 June 2025; pp. 9762–9771. https://doi.org/10.1109/CVPR52734.2025.00912.
39. Raghu, M.; Zhang, C.; Kleinberg, J.; Bengio, S. Transfusion: Understanding Transfer Learning for Medical Imaging. In Proceedings of the Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019; Volume 32, pp. 3347–3357.
40. Moffett, L.; Willard, F.; Machado, M.; Mokel, E.; Donnelly, J.; Guo, Z.; Costarino, A.; Yang, J.; Kim, G.; Barnett, A.J.; et al. Cosine Similarity is Almost All You Need (for Prototypical-Part Models). In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV 2026), Tucson, AZ, USA, 6–10 March 2026; pp. 1586–1596. https://doi.org/10.1109/WACV61042.2026.00160.
41. Arandjelović, R.; Gronat, P.; Torii, A.; Pajdla, T.; Sivic, J. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), Las Vegas, NV, USA, 27–30 June 2016; pp. 5297–5307. https://doi.org/10.1109/CVPR.2016.572.
42. Martins, A.; Astudillo, R. From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. In Proceedings of the 33rd International Conference on Machine Learning (ICML 2016), New York, NY, USA, 19–24 June 2016; Proceedings of Machine Learning Research, Volume 48, pp. 1614–1623.
43. Barnett, A.J.; Schwartz, F.R.; Tao, C.; Chen, C.; Ren, Y.; Lo, J.Y.; Rudin, C. A case-based interpretable deep learning model for classification of mass lesions in digital mammography. *Nat. Mach. Intell.* **2021**, *3*, 1061–1070. https://doi.org/10.1038/s42256-021-00423-x.
44. Huy, T.D.; Tran, S.K.; Nguyen, P.; Tran, N.H.; Sam, T.B.; van den Hengel, A.; Liao, Z.; Verjans, J.W.; To, M.S.; Phan, V.M.H. Interactive Medical Image Analysis with Concept-based Similarity Reasoning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2025), Nashville, TN, USA, 11–15 June 2025; pp. 30797–30806. https://doi.org/10.1109/CVPR52734.2025.02868.
45. Huang, Q.; Xue, M.; Huang, W.; Zhang, H.; Song, J.; Jing, Y.; Song, M. Evaluation and Improvement of Interpretability for Self-Explainable Part-Prototype Networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV 2023), Paris, France, 1–6 October 2023; pp. 2011–2020. https://doi.org/10.1109/ICCV51070.2023.00192.
46. Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K.Q. On Calibration of Modern Neural Networks. In Proceedings of the 34th International Conference on Machine Learning (ICML 2017), Sydney, Australia, 6–11 August 2017; Proceedings of Machine Learning Research, Volume 70, pp. 1321–1330.
47. FAIR (Meta). fvcore: Collection of Common Code Shared Among Different Research Projects in FAIR Computer Vision Team, Version 0.1.5.post20221221; Python Package. 2022. Available online: https://pypi.org/project/fvcore/0.1.5.post20221221/ (accessed on 3 November 2026).

48. PyTorch Contributors. Reproducibility. In *PyTorch 2.11 Documentation*; PyTorch Foundation: San Francisco, CA, USA, 2026. Available online: https://docs.pytorch.org/docs/2.11/notes/randomness.html (accessed on 3 November 2026).
49. Geirhos, R.; Jacobsen, J.H.; Michaelis, C.; Zemel, R.; Brendel, W.; Bethge, M.; Wichmann, F.A. Shortcut learning in deep neural networks. *Nat. Mach. Intell.* **2020**, *2*, 665–673. https://doi.org/10.1038/s42256-020-00257-z.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.