In the Supplementary Material we first reexamine the notation used in the paper in Section A. Then we provide a detailed comparison of SAE to generalised methods and baselines in Section B. Then we summarise the training and evaluation of SAE as an Algorithm in Section C. We provide additional details on the experimental setup in Section D. Then we describe the practical implementation details and hardware cost of SAE in Section E. We provide additional results for the baseline performance evaluation in Section F. In Section G, we investigate the impact of changing the number of inputs N and the number of exits K on the OOD test sets. In Section H, we examine the impact of N and K on the depth preference of the learnt exit weights. Lastly, in Section I, we detail the impact of the introduced hyperparameters on the performance of SAE.

A Notation

The Table 1 summarises the notation used in the paper. In this work, we assume a standard single exit NN, a singular member of the naive ensemble, to be a function which maps an input x to an output $\hat{y}$ for target y, with learnable weights $\mathbf{w}$. The function can be decomposed into D layers $h^D(\cdot) \circ f^D(\cdot) \circ f^{D-1}(\cdot) \circ \cdots \circ f^1(\cdot)$, where $f^j(\cdot)$ is the j-th layer and $h^D(\cdot)$ is the prediction head at the end of the network giving the output $p(y|x,\mathbf{w}) = p(y|\hat{y}); \hat{y} = h^D(f^D(f^{D-1}(\dots f^1(x))))$. For generality, we assume a layer $f^j(\cdot)$ to be an arbitrary weighted operation, such as a combination of convolution, batch normalisation, pooling or activation layers [32], residual block [15] or a transformer encoder layer [37] and a prediction head $h(\cdot)$ to be a fully-connected layer preceded by a global average pooling layer and any necessary parametric or non-parametric reshaping operation to convert the hidden representation into the desired output shape.

B Detailed Comparison to Generalised Methods and Baselines

Generalised Methods: A distinctive feature of SAE is its ability to determine the most effective exit depth for each input, a capability previously investigated in SE NN, EE and MIMMO [2, 8], but not in MIMO [14]. MIMMO [8], where $N \geq 2$ and $K = D$, is the closest approach to SAE. However, MIMMO did not consider creating a search space generalising the previous methods or the introduction of the new in-between approaches. It also learns the exit depth for each input but uses all the exits during evaluation. As empirically observed, small capacity NNs might not have enough capacity to make meaningful predictions at all depths, hence the need for the K parameter. SAE introduces the K parameter and has to choose an appropriate sampling strategy for the depth variables, which allows for the selection of the most effective exits for each input, features not present in MIMMO. Additionally, SAE diverges from the MIMMO by not adopting a common prediction head across all exits, which we empirically found to degrade performance. Moreover, SAE introduced the hyperparameter scheduling of $i(t)$ and the additional regularisation terms $\alpha(t)$, $T(t)$, which were not considered in MIMMO. Last but not least, MIMMO was demonstrated only for classification using a convolutional architecture, while SAE is demonstrated for classification and regression using a variety of architectures.

In essence, SAE's objective is not to rival the generalised methods but to amalgamate them into a cohesive framework facilitated by setting N and K and empirically investigate

SYMBOL	DESCRIPTION
NN	NEURAL NETWORK
$\mathcal{D}$	DATASET
$\lvert\mathcal{D}\rvert$	NUMBER OF SAMPLES IN THE DATASET
D	DEPTH OF THE NETWORK
$\{f^j(\cdot)\}_{j=1}^D$	LAYERS OF THE NN
$\{h^j(\cdot)\}_{j=1}^D$	PREDICTION HEADS OF THE NN
$\mathbf{w}$	WEIGHTS OF THE NN
N	NUMBER OF SIMULTANEOUSLY PROCESSED INPUTS/NUMBER OF SUBMEMBERS
K	MAXIMUM NUMBER OF EXITS USED BY SUBMEMBER
$\{d_i\}_{i=1}^N$	LATENT DEPTH VARIABLES FOR EACH SUBMEMBER IN N
$\{\theta_i^j\}_{i,j=1}^{N,K}$	WEIGHTS OF THE j-TH EXIT OF THE i-TH SUBMEMBER
θ^*	LARGEST DEPTH WEIGHTS SELECTED WITH RESPECT TO K
$\{l_i^j\}_{i,j=1}^{N,K}$	LOGITS OF THE j-TH EXIT OF THE i-TH SUBMEMBER
x	INPUT TO THE NN
y	TARGET OUTPUT
$\{\hat{y}_i^j\}_{i,j=1}^{N,K}$	PREDICTED OUTPUTS OF THE j-TH EXIT OF THE i-TH SUBMEMBER
$\mathbf{x} = \{x_i\}_{i=1}^N$	N-MEMBER INPUT
$\mathbf{X}_i = \{\mathbf{x}_i^{(j)}\}_{j=1}^{\lvert\mathcal{D}\rvert}$	INPUT SAMPLES FOR THE i-TH SUBMEMBER ACROSS THE DATASET $\mathcal{D}$
$Y_i = \{y_i^{(j)}\}_{j=1}^{\lvert\mathcal{D}\rvert}$	TARGET OUTPUTS FOR THE i-TH SUBMEMBER ACROSS THE DATASET $\mathcal{D}$
$\mathcal{L}$	LOSS FUNCTION
$q(\cdot)$	VARIATIONAL DISTRIBUTION
B	BATCH SIZE
H	IMAGE HEIGHT
W	IMAGE WIDTH
C	IMAGE CHANNELS
F	FEATURE SIZE
S	SEQUENCE LENGTH
P	PATCH SIZE
O	NUMBER OF OUTPUTS
$\alpha(t)$	RE-WEIGHTING FACTOR BETWEEN DATA FIT AND REGULARIZATION AT TIME t
$T(t)$	TEMPERATURE PARAMETER AT TIME t
$i(t)$	INPUT REPETITION PROBABILITY AT TIME t

Table 1: Notation used in the paper.

the trade-offs between the generalised approaches and their novel in-between counterparts explorable through our proposed search space and problem formulation.

Naive NN Ensemble The naive ensemble [19] comprises N independent models, each with the same architecture as the base model. Each model is trained independently on the same training data, scaling the training time linearly with the number of inputs N. During the evaluation, input is processed through each model, and the predictions are averaged across all models, scaling the inference time or consumed memory linearly with the number of inputs N.

Batch Ensemble The Batch Ensemble [38] is contained in a single architecture, making it more efficient than the naive ensemble and requiring only a single training and inference run. However, to apply elementwise rank-1 weight matrices to the input and output of each layer, scaling with the number of inputs N, all the linear and convolutional layers must be re-implemented. This re-implementation might require custom kernels for the forward and backward passes for efficient computation. This increases the number of FLOPs and marginally increases the number of parameters for each layer. Practically, Batch Ensemble defines the parameters for each input in N manually through the rank-1 weight decomposition.

Monte Carlo Dropout Monte Carlo Dropout [11] is contained in a single architecture, making it more efficient than the naive ensemble and requiring only a single training run. A dropout layer [34] is added before each linear and convolutional layer, randomly dropping out a fraction of the input nodes. Therefore, dropout requires an efficient random number generator and dropout mask application before each linear or convolutional layer. The repeated forward passes are used to obtain the Monte Carlo samples for the predictions, scaling the inference time linearly with the number of inputs N. At the same time, the memory consumed by parameters stays constant.

Compared to the baseline methods, SAE is trained and evaluated in a single run, making it more efficient than the compared methods. At the same time, it does not require any re-implementation of the layers or random number generators, making it more efficient than Batch Ensemble and Monte Carlo Dropout. It automatically learns the weights which react to the features of the input data, not requiring manual parameterisation, making it more efficient than Batch Ensemble.

C Algorithm

The Algorithm 1 summarises the training and evaluation procedures for the Single Architecture Ensemble (SAE) framework. The SAE training operates over a fixed number of steps t_{end}. For each step, it iterates over a batch size B, $\mathbf{x}^{(b)} = \{\mathbf{x}_i^{(b)}\}_{i=1}^N$ and $\mathbf{y}^{(b)} = \{y_i^{(b)}\}_{i=1}^N$, sampled from the dataset $\mathcal{D}$ according to the input repetition factor $i(t)$ and N. It then computes predictions $\{\hat{y}_i^{j,(b)}\}_{i,j=1}^{N,D}$ using the current model and samples the depth variables from a distribution $q(d|\theta)$ for all N. Using these predictions and the sampled depth variables, the loss $\mathcal{L}(\mathbf{w},\theta)$ is calculated, and $\mathbf{w},\theta$ are updated through backpropagation. The evaluation involves retaining the top K parameters of θ while setting the rest of the logits to negative infinity. It repeats input $\mathbf{x}^* = \{x_i\}_{i=1}^N$ N times and computes the weighted prediction $\hat{y}^*$.

D Experimental Setup

Our code is implemented in PyTorch. No specific hardware optimisations were used to accelerate the training or evaluation for any methods. We fixed the number of epochs to 50 for all datasets except TinyImageNet, for which we set it to 100. We use Adam optimiser, starting with a learning rate of 3e-4 and a weight decay of 1e-5, except TinyImageNet, for which we set the weight decay to 0, and cosine annealing learning rate scheduler [24] across all datasets. We used batch size 64 for PneumoniaMNIST and RetinaMNIST and 128 for TinyImageNet and BloodMNIST. For Batch Ensemble, we had to find a batch size close to the other experiments' batch size and a multiple of $1 \leq N \leq 4$. Therefore, for experiments where batch size 64 was used, we used 72 for Batch Ensemble, and for experiments where batch size 128 was used, we used 144 for Batch Ensemble. The batch repetition factor [14] for SAE was set to 2, except for TinyImageNet, where we set it to 1. We used gradient

Algorithm 1 Single Architecture Ensemble (SAE)

```
1: procedure TRAIN($\mathcal{D}$,$\mathbf{w}$,$\theta$,$B$,$N$,$D$,$K$,$\alpha(t)$,$T(t)$,$i(t)$, $t_{end}$)
2:     for $t = 1$ to $t_{end}$ do
3:         for $b = 1$ to $B$ do
4:             Sample $\mathbf{x}^{(b)}$, $\mathbf{y}^{(b)}$ from $\mathcal{D}$ at batch index $b$ with respect to input repetition factor $i(t)$ and $N$
5:             Predict $\{\hat{y}_i^{j,(b)}\}_{i,j=1}^{N,D}$
6:         end for
7:         Sample $\{d_i\}_{i=1}^{N}$
8:         Compute $\mathcal{L}(\mathbf{w},\theta)$ using Equation 3
9:         Backpropagate and update $\mathbf{w}$, $\theta$
10:    end for
11: end procedure
12: procedure EVALUATE($\mathcal{D}$,$\mathbf{w}$,$\theta$,$N$,$D$,$K$,$T(t_{end})$)
13:    Keep top $K$ $\theta^*$ and set the rest to -inf for all $i \in N$; initialise top $K$ active exits
14:    Repeat input $\mathbf{x}^* = \{x_i\}_{i=1}^{N}$ $N$ times
15:    Predict $\{\hat{y}_i^j\}_{i,j=1}^{N,K}$
16:    Predict $\hat{y}^* = \frac{1}{N}\sum_{i,j=1}^{N,K} \hat{y}_i^j \theta_i^{*j}$
17: end procedure
```

clipping with a maximum norm of 5.0 to avoid exploding gradients across all experiments. We used the default PyTorch initialisation for all layers. We fixed the base hyparameters (HP)s, such as the learning rate, weight decay, batch size, and number of epochs for all methods, to the same values for fair comparison and to demonstrate the ease of implementing SAE based on existing solutions.

We used the validation data to guide the multi-objective Bayesian optimisation (MOBO) from syne-tune [28, 31] to perform HPO. We used 10% of the training data as validation data across all datasets. We enable 50 random initialisations for exploration. We used 2 GTX 1080 and 2 RTX 2080 GPUs in one machine, giving us 4 workers for the HPO. We run for 8 hours on 4 GPUs for BloodMNIST, PneumoniaMNIST, and RetinaMNIST and 8 days on 4 GPUs for TinyImageNet. For SAE we set the search space for $i_{start}(t)$ to $[0.0, 1.0]$ and $i_{end}(t)$ to $[0.0, 1.0]$. For $\alpha_{start}(t)$ and $\alpha_{end}(t)$ we set the search space to $[0.00001, 1.0]$ on a log scale. For $T_{start}(t)$ and $T_{end}(t)$ we set the search space to $[0.001, 1.0]$ on a log scale. We set the search space for the dropout rate to $[0.0, 1.0]$ on a linear scale. Dropout was not added before the first layer and after the last layer in any architecture. $N = 4$ Monte Carlo samples are used during evaluation for Dropout. For ViT, it was added only where dropout would usually be added in the original architecture.

For TinyImageNet, we performed experiments on a ResNet with [3,4,6,3] residual blocks per stage, strides [2,2,2,2] and channels [64,128,256,512] within which we only enabled the search for the number of inputs N and a depth $D \in \{1,4\}$, in addition to the regularisation parameters. Here, depth refers to the stage depth, not the overall depth of the network. With $D = 4$, this is the standard ResNet-34 architecture.

For ViT and RetinaMNIST, we enable to vary the encoder depth $D \in \{1,5\}$, width multiplier for base hidden dimension 192 as $W \in \{1,4\}$, the patch size 4, 4 heads and embedding dimension 192 and no dropout enabled by default. The depth refers to the number of encoder layers in ViT. The width multiplier multiplies the base hidden dimension of the ViT encoder.

For ViT, we add a token for each additional input, but we design the final layer as a pooling of all the tokens to arrive at the prediction. We empirically observed that this leads to better performance than just the added tokens for prediction.

For VGG and BloodMNIST, we enable to vary $D \in \{1,5\}$ and $W \in \{1,4\}$ with base width $[4,8,16,32,32]$ per stage and $[1,1,1,1,1],[1,1,2,2,2]$ blocks per stage, we omit the repeated memory-expensive linear layers at the end of the network and replace them with a single predictive linear layer. Each block consisted of a convolutional layer, batch normalisation, and ReLU; the first three stages included a max pooling layer in the last block of each stage. The depth refers to the number of stages in VGG. The width multiplier multiplies the base width of the VGG network per stage.

For FC and PneumoniaMNIST, we design a residual fully connected network where the input is processed through an initial linear layer, upscaled to the hidden dimension, and then processed through a series of residual blocks, of a linear layer, batch normalisation, ReLU, we enable to vary $D \in \{1,5\}$ and $W \in \{1,4\}$ with base width 128. The depth refers to the number of residual blocks in FC. The width multiplier multiplies the base width of the FC network. We varied the search space size from the smallest, for example, on TinyImageNet, to the largest, for instance, on BloodMNIST, to demonstrate the adaptability of SAE to different search spaces. Note that for SAE, the depth is permanently fixed to the maximum.

The classification on TinyImageNet, BloodMNIST and RetinaMNIST was performed by processing the outputs of the last layer through a softmax function. The mean over all predictions was used as the final prediction for all the methods. For regression on RetinaMNIST, one output node was used as the mean, and the second output was processed through an exponential function to obtain the variance for minimising the Gaussian negative log-likelihood per example. The mean over all predictions was computed as the mean across all the mean outputs. The variance was computed as the mean of the variances plus the variance of the means by the law of total variance.

The SAE implementation is available via YAMLE [9] at `https://github.com/martinferianc/yamle`.

D.1 Datasets

We used the following datasets for our experiments:

MedMNIST MedMNIST [40] offers diverse medical imaging tasks: binary/multi-class classification and regression in a standardised, MNIST-like format that enables benchmarking with a reasonable computational cost. Given its multi-fidelity nature covering grayscale or RGB images, class imbalance and varying number of samples in the datasets, it presents a more realistic and challenging benchmark than other clean datasets such as MNIST, FashionMNIST or CIFAR-10/100.

- **PneumoniaMNIST** is derived from a dataset of pediatric chest X-ray images aimed at binary classification between pneumonia and normal cases. It includes 5,856 images originally pre-processed to 28×28 grayscale images.
- **BloodMNIST** consists of 17,092 blood cell microscope images from individuals without infection or disease, categorised into 8 classes representing different cell types. The images are RGB and originally pre-processed to 28×28.

- **RetinaMNIST** originates from the DeepDRiD challenge, offering 1,600 retina fundus images for regression to grade diabetic retinopathy severity on a five-level scale. The images are RGB and originally pre-processed to 28×28.

In pre-processing, we pad the images to 32 pixels, random cropping and normalisation are applied to the training data, and normalisation is only used for the validation and test data. For RetinaMNIST, we normalised the ordered levels of the regression target to $[0,1]$, and we modelled the prediction as parameters of a Gaussian distribution with a mean and variance output to enable us to model the uncertainty in the predictions and benchmark the negative log-likelihood.

TinyImageNet TinyImageNet [21] is a subset of the ImageNet dataset [5] with 200 classes and 500 images per class, resulting in 100,000 training images and 10,000 test images. Compared to MedMNIST, TinyImageNet is a more challenging dataset with higher-resolution images and more classes. However, the dataset is balanced, and the images are relatively clean. The images are RGB and pre-processed initially to 64×64. In pre-processing, random cropping and normalisation are applied to the training data, and normalisation is only used for the validation and test data. For TinyImageNet, we used 10% of the training data as validation data.

We considered datasets of varying image sizes (64×64 against 28×28) and varying numbers of classes (200 against 2 and 8), fidelity (RGB against grayscale) and class imbalance (balanced against imbalanced), sizes (100,000, 17,092, 5,856 and 1,600) to demonstrate the adaptability of SAE to different datasets and environments. Naturally, given the dataset size, architecture, and parameterisation, the training time for the experiments varied, which also varied the number of HPO runs we performed within our budget. This aimed to demonstrate the robustness of SAE to different search budgets.

OOD Datasets To create the OOD test data, we used the augmentations from [16] and applied them to the test data before normalisation. These corruptions include, for example, adding snow or fog to the image, changing the brightness or saturation of the image or blurring the image across 5 intensities. We averaged the performance across all the augmentations and severities except FSGM to obtain scalar metrics for the OOD test data. For PneumoniaMNIST, we had to convert the grayscale image to RGB, apply the augmentations, and then convert it back to grayscale. The OOD test aimed to demonstrate the robustness of SAE to OOD data where naive ensembles are known to outperform many state-of-the-art methods [27].

We chose feed-forward, residual, convolutional, fully connected, or transformer architectures of various capacities, a combination of classification and regression tasks, and balanced and unbalanced datasets in RGB or grayscale, with many or few data samples across ID and OOD data to demonstrate the versatility of our framework in different applications.

D.2 Algorithmic Metrics

We used the following metrics to evaluate the performance of the methods:

Accuracy Accuracy is the percentage of correctly classified samples defined in Equation 4. We want to maximise accuracy for classification tasks.

$$\text{Accuracy} = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\mathbb{1}_{\hat{y}_i = y_i} \tag{4}$$

where $\mathcal{D}$ is the dataset, $\hat{y}_i$ is the predicted label for the i-th sample and y_i is the ground truth label for the i-th sample.

F1 Score F1 score is the harmonic mean of precision and recall defined in Equation 5. It is more suitable for imbalanced datasets than accuracy. We want to maximise the F1 score for classification tasks.

$$\text{F1 Score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

where $\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}$ and $\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}$. TP, FP and FN are the number of true positives, false positives and false negatives, respectively. We measured the macro F1 score, the average of the F1 scores for each class, in a one-vs-all fashion.

Expected Calibration Error Expected calibration error (ECE) [13] measures a model's confidence calibration. A model is well-calibrated if the confidence of the model's predictions matches the accuracy of the predictions. We want to minimise ECE for classification tasks. ECE is defined in Equation 6.

$$\text{ECE} = \sum_{m=1}^{M}\frac{|B_m|}{|\mathcal{D}|}|\text{acc}(B_m) - \text{conf}(B_m)| \tag{6}$$

where $\mathcal{D}$ is the dataset, M is the number of bins, B_m is the M-th bin, $|B_m|$ is the number of samples in the M-th bin, $\text{acc}(B_m)$ is the accuracy of the M-th bin and $\text{conf}(B_m)$ is the confidence of the M-th bin. We used 15 bins for all datasets.

Class-Conditional Expected Calibration Error The class-conditional expected calibration error (CC-ECE) [13] measures the calibration of a model for each class. It is better suited for imbalanced datasets than ECE. We want to minimise CC-ECE for classification tasks. CC-ECE is defined in Equation 7.

$$\text{CC-ECE} = \sum_{c=1}^{C}\frac{|\mathcal{D}_c|}{|\mathcal{D}|}\text{ECE}(\mathcal{D}_c) \tag{7}$$

where $\mathcal{D}_c$ is the set of samples with the highest probability of class c and C is the number of classes.

Negative Log-Likelihood Negative log-likelihood (NLL) is a proper scoring rule for probabilistic models. It measures the average negative log probability of the ground truth labels for classification and the average negative log-likelihood of the ground truth values for regression. We want to minimise NLL for classification and regression tasks. NLL for classification is defined in Equation 8, and NLL for regression is described in Equation 9.

$$\text{NLL} = -\frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\mathbb{1}_{\hat{y}_i=y_i}\log\hat{y}_i \tag{8}$$

where $\hat{y}_i$ is the predicted probability of the ground truth label for the i-th sample.

$$\text{NLL} = -\frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\log\mathcal{N}(y_i|\hat{\mu}_i,\hat{\sigma}_i^2) \tag{9}$$

where $\mathcal{N}$ is the normal distribution, y_i is the ground truth value for the i-th sample, $\hat{\mu}_i$ is the predicted mean for the i-th sample and $\hat{\sigma}_i$ is the predicted standard deviation for the i-th sample.

Mean Squared Error Mean squared error (MSE) is a regression metric measuring the average squared difference between the predicted and ground truth values. We want to minimise MSE for regression tasks. MSE is defined in Equation 10.

$$\text{MSE} = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}(\hat{\mu}_i - y_i)^2 \tag{10}$$

where $\hat{\mu}_i$ is the predicted value for the i-th sample and y_i is the ground truth value for the i-th sample.

We use the F1 score and CC-ECE to measure the performance under unbalanced settings instead of accuracy or ECE insensitive to class imbalance.

E Practical Implementation and Hardware Cost

In this Section, we analyse the hardware cost of the SAE approach in detail, covering everything from modifying the input layer to adding the early exits.

E.1 Input Layer

The input layer is modified to accept a set of N inputs, one for each input in N. In practice, considering image data of size $C \times H \times W$, the input layer is modified to accept a set of N inputs of size $N \times C \times H \times W$. That is done simply by concatenating the N inputs along the channel dimension.

FC The input layer is, by default, a fully connected layer for the residual FC network. The difference from standard NN is simply increasing the input dimension from C to $N \times C$ while leaving the output dimension F unchanged.

The number of parameters for the input layer is:

- Fully Connected: $F \times N \times C + F$.

This is in comparison to the standard NN where the number of parameters for the input layer is: $F \times C + F$ or when $N = 1$.

The number of FLOPs for the input layer is:

- Fully Connected: $F \times N \times C + F$.

This is in comparison to the standard NN where the number of FLOPs for the input layer is: $F \times C + F$ or when $N = 1$.

ResNet & VGG For ResNet and VGG, the input layer is, by default, a convolutional layer. The difference to standard NN is simply increasing the input channels in the convolution from C to $N \times C$ while leaving the output channel count F unchanged.

The number of parameters for the input layer is:

- Convolution: $F \times N \times C \times K \times K + F$, K is the kernel size.

This is compared to the standard NN where the input layer parameters are $F \times C \times K \times K + F$ or when $N = 1$.

The number of FLOPs for the input layer is:

- Convolution: $F \times N \times C \times K \times K \times H \times W + F \times H \times W$ and assuming stride 1.

This is in comparison to the standard NN where the number of FLOPs for the input layer is: $F \times C \times K \times K \times H \times W + F \times H \times W$, again assuming stride 1 or when $N = 1$.

ViT For ViT, the input layer creating the S embeddings from input patches is, by default, a fully connected layer. The difference to standard NN is simply increasing the input dimension from $P \times P \times C$ to $P \times P \times N \times C$ where P is the patch size.

The number of parameters for the input layer is:

- Fully Connected: $F \times P \times P \times N \times C + F$.

This is in comparison to the standard NN where the number of parameters for the input layer is: $F \times P \times P \times C + F$ or when $N = 1$.

The number of FLOPs for the input layer is:

- Fully Connected: $F \times P \times P \times N \times C \times S + F \times S$.

This is in comparison to the standard NN where the number of FLOPs for the input layer is: $F \times P \times P \times C \times S + F \times S$ or when $N = 1$.

E.2 Early Exits

The hardware cost of adding the early exits to the FC, ResNet, VGG and ViT are as follows. We denote $\{n^i\}_{i=1}^{D}$ as the number of inputs using the exit at depth i for all depths D and F^{last} and $\{F^i\}_{i=1}^{D}$ as the feature size at the last layer and the depth i respectively. We define the F^{last} as the last hidden dimension of the backbone, e.g. at the position of global average pooling for ResNet for the VGG or as the embedding dimension for ViT.

FC We add an early exit after a residual block for the residual FC network. The early exit is composed of a fully connected layer with F^{last} output nodes, batch normalisation layer, ReLU activation function and a prediction head with F^{last} input nodes and $n^i \times O$ output nodes for the i-th exit. Therefore, the number of parameters for the early exit, conditioned on the number of inputs n^i using the exit at depth i, is:

- Fully Connected: $F^i \times F^{last} + F^{last}$.

- Batch Normalisation: $2 \times F^{last}$.
- Prediction head: $F^{last} \times n^i \times O + n^i \times O$.

The number of FLOPs for the early exit, conditioned on the number of inputs n^i using the exit at depth i, is:

- Fully Connected: $F^i \times F^{last} + F^{last}$.
- Batch Normalisation: $2 \times F^{last}$.
- ReLU: F^{last}.
- Prediction head: $F^{last} \times n^i \times O + n^i \times O$.

ResNet & VGG For ResNet and VGG, we add an early exit after a set of residual or feed-forward blocks. The early exits are composed of a reshaping 1×1 convolutional layer with stride 1 with F^{last} output channels, a batch normalisation layer, ReLU activation function, global average pooling followed by the prediction head with F^{last} input nodes and $n^i \times O$ output nodes for the i-th exit. Therefore, the number of parameters for the early exit, conditioned on the number of inputs n^i using the exit at depth i, is:

- Convolution: $F^i \times F^{last} \times 1 \times 1 + F^{last}$.
- Batch Normalisation: $2 \times F^{last}$.
- Prediction head: $F^{last} \times n^i \times O + n^i \times O$, where O is the number of output nodes.

The number of FLOPs for the early exit, conditioned on the number of inputs n^i using the exit at depth i, is:

- Convolution: $F^i \times F^{last} \times 1 \times 1 \times H^i \times W^i + F^{last} \times H^i \times W^i$, H^i and W^i are the height and width of the feature map at the given depth i.
- Batch Normalisation: $2 \times F^{last} \times H^i \times W^i$.
- ReLU: $F^{last} \times H^i \times W^i$.
- Global Average Pooling: $F^{last} \times H^i \times W^i$.
- Prediction head: $F^{last} \times n^i \times O + n^i \times O$.

ViT For ViT, the early exits are composed of a fully connected layer with F input and output channels, where F is the embedding dimension, followed by a GeLU activation, layer normalisation, reduction via average pooling across the sequence dimension, and a prediction head which is a linear layer with F input nodes and $n^i \times O$ output nodes. Therefore, the number of parameters for the early exit, conditioned on the number of inputs n^i using the exit at depth i, is:

- Fully connected layer: $F \times F + F$.
- Layer normalisation: $2 \times F$.
- Prediction head: $F \times n^i \times O + n^i \times O$.

The FLOPs for the early exit, conditioned on the number of inputs n^i using the exit at depth i, are:

- Fully connected layer: $F \times F \times (S+N) + F \times (S+N)$, where S is the sequence length.
- GeLU: $F \times (S+N)$.
- Layer normalisation: $2 \times F \times (S+N)$.
- Reduction: $F \times (S+N)$.
- Prediction head: $F \times n^i \times O + n^i \times O$.

The early exits are added during inference only for the top K chosen early exits decided by the learnt variational parameters θ for each input in N.

In summary, increasing N is simpler than increasing K. If N is increased, only the input and exit layers need to be modified, whereas if K is increased, the early exits need to be added which favours increasing N over K. The overall training time of the model is marginally increased by N or K in comparison to an SE NN, however, the early exits introduce branching in the architecture and if not implemented efficiently, the training time can increase. Likewise, the evaluation time is only marginally increased by N or K and caution should be taken to implement the early exits efficiently to avoid a significant increase in evaluation time.

F Supplementary for Baseline Performance

F.1 NLL and OOD Data

In Figures 5, 6, 7, and 8 we provide the additional results for the baselines across all datasets with respect to NLL on ID test data and OOD test data.

For TinyImageNet and the ID test set in Figure 5a, the I/B ✚ configuration is better in NLL in comparison to the naive ensemble ■ with 3.2× fewer FLOPs and 1.5× fewer parameters. On OOD data in Figures 5b, 5c, and 5d, the I/B ✚ configuration is within the standard deviation of accuracy and with better mean ECE and NLL than the best ensemble ■ while conserving the same FLOPs and parameters.

For BloodMNIST and the ID test set in Figure 6a, the I/B ✚ configuration is comparable to the naive ensemble ■ in NLL with 1.5× fewer FLOPs and a comparable number of parameters. On OOD data, seen in Figures 6b, 6c, and 6d, the SAE configurations all achieve high F1 score in comparison to the baselines, and a I/B ✚ configuration is within 2% of CC-ECE or F1 of the best ensemble ■ with 2.7× fewer FLOPs and 3.7× fewer parameters.

For PneumoniaMNIST and the ID test set in Figure 7a, the EE ⬢ configuration achieves one of the lowest NLLs with approximately half the FLOPs and parameters of the best BE ◀. On OOD data, in Figures 7b, 7c, and 7d, the MIMO ⬟ configuration is within 1% of the best BE ◀ with similar FLOPs and parameters.

Lastly, for RetinaMNIST and the OOD test set in Figures 8a and 8b, the I/B ✚ configuration is the best-performing method with approximately 2× fewer FLOPs than the second best BE ◀.

F.2 Overall Baseline Performance Ranks

In Tables 2, 3, 4 and 5 we collected all the Pareto candidates for a method and a dataset, calculated the mean performance of the candidates across all metrics and then ranked the methods with respect to each other and their mean per-metric performance. Then we averaged the ranks across all the datasets per classification and regression. The best mean ranking, where lower ranks are better, is highlighted in bold. As it can be seen in the Tables, the methods generalised under SAE rank top or second across all datasets and metrics, demonstrating the robustness of SAE to different datasets and environments whether considering classification or regression tasks or ID or OOD data. In case of FLOPs and Params, one needs to be careful with interpreting the results, as the methods with the lowest FLOPs and Params are not necessarily the best-performing methods in terms of the algorithmic metrics.

F.3 Hyperparameter Optimisation

In Figure 9, we provide the hyperparameter optimisation (HPO) results for the baselines across all datasets. The plots contain all the configurations tried by HPO, with high-opacity configurations indicating the configurations used for the final evaluation over repeated seeds. The configurations were decided by looking at their empirical Pareto optimality across all the methods and then for each method individually across all the datasets. These results demonstrate the HPO trends and algorithmic performance range that the baselines and SAE can achieve. As the Figure shows, SAE tends to cluster around the Pareto front, demonstrating the adaptability of SAE to different search spaces and the robustness of the HPO process. At the same time, this enables the practitioner to choose multiple configurations for SAE, which can be used to trade-off between the number of FLOPs, the number of parameters, and the algorithmic performance.

F.4 Numerical Results

In Tables 6, 7, 8, 9, 10, 11, 12, and 13, we provide the numerical results for the baselines across all datasets. As in the main paper, we divide the SAE approach into the following categories: I/B, EE, MIMMO, MIMO, and SE NN. We specify how many configurations were selected for the repeated seed evaluation. We specify the configurations for each category where the first row of each method denotes the best performance on the first metric, the second row indicates the best performance on the second metric, and the third row represents the best performance on the third metric. The **bold** font denotes the best performance across all the methods, and the *italic* and <u>underlined</u> font denote the configurations compared in the text.

G Supplementary for Changing K and N

In Figures 10, 11, 12 and 13 we show the Varying N, K on TinyImageNet, RetinaMNIST, PneumoniaMNIST and BloodMNIST datasets concerning different metrics. As discussed in the main part of the paper, there might be different suitable configurations for various metrics. We would point the reader's attention to the OOD results across all datasets and the ID results where we have shown that the N and K are different. This further indicates the necessity of SAE and the proposed search space and optimisation strategy, since the best configurations are not consistent across the same metric, dataset and backbone architecture.

H Supplementary for Depth Preference

In Figures 14, 15, 16 and 17, we show the depth preference for different N and K on TinyImageNet, RetinaMNIST, BloodMNIST and PneumoniaMNIST, respectively. We categorise the results according to N and K and show the average depth preference across all K or N, respectively. The results show that the depth preference is consistent across different N and K. Depending on the complexity of the problem and the capacity of the backbone architecture, the depth preference can vary, but the general trend is that the deeper exits are preferred in comparison to the shallower ones. As K is increased, the networks also utilise the shallower exits more, however, the preference for the deeper exits is still present. Interestingly, as N is increased, the depth preference changes and we see that, for example, the network might prefer the deepest exit for processing one input and a shallower exit for processing another input. This is best seen in the TinyImageNet dataset in Figure 14. We suspect that this is the network's way of adapting to the complexity of the input data and the capacity of the backbone architecture.

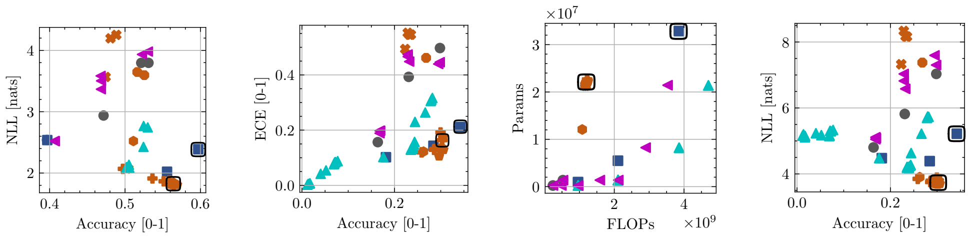
I Supplementary for Hyperparameter Influence

In Tables 14, 15, 16, and 17 we benchmark the influence of the hyperparameters on the performance of the SAE across all datasets and metrics. The hyperparameters are the number of processed inputs N, the maximum number of early exits K, the width multiplier W, the start and end input repetition probability i_{start} and i_{end}, the start and end temperature T_{start} and T_{end} and the start and end values of the trade-off regulariser α_{start} and α_{end}. We measured the correlation between the hyperparameters and the performance metrics via the Spearman rank correlation coefficient [41] when the hyperparameters are continuous such as $i(\cdot)$, $T(\cdot)$, and $\alpha(\cdot)$. For categorical hyperparameters such as N, K, and W, we computed the ANOVA F-value [12]. Note that ANOVA is not a direct measure of correlation, but it is used to determine whether there are statistically significant differences between the means of two or more groups. Hence it does not provide a direct measure of the strength of the relationship between the hyperparameter and the performance metric but it can be used to determine the importance of the hyperparameter's value on the performance metric.

The influence of input repetition probability i varies across datasets in its impact on metrics such as NLL and Accuracy, suggesting its optimisation should be tailored per dataset characteristics. The temperature T demonstrates minimal but consistent influence across datasets, especially towards the end values T_{end}, indicating its role in the stability and fine-tuning of the SAE NN. Correlations with performance metrics such as NLL and Accuracy fluctuate for α, which points towards its role in controlling the trade-off between different data fitting and regularisation. The variance of N and K show that they strongly influence predictive performance and computational overhead across datasets. The width multiplier W exerts substantial influence on the computational demand, FLOPs and number of parameters, particularly evident in the RetinaMNIST dataset. Changes in W also affect model accuracy and other performance metrics, which suggests that increasing the network width is crucial for capturing adequate feature representations enabling the fitting of multiple predictors in the same architecture.

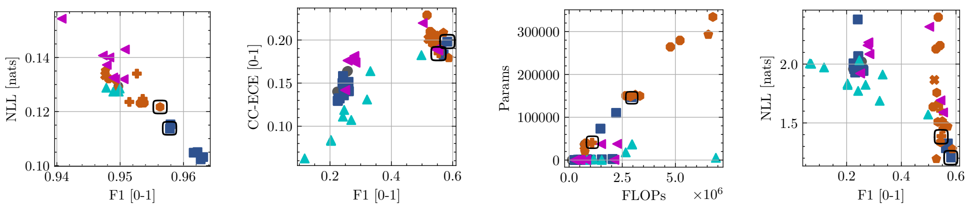
The N and K hyperparameters need to be observed in the context of $\alpha(t)$, $T(t)$ and $i(t)$, which balance the data fitting and regularisation. If $\alpha(t)$, is high the distribution of the exit weights will be more spread out across exits, showing that the network's representations are already useful for predictions at earlier depths. If the $T(t)$ is low, the exit weights will be sharper, showing focus only on the most confident depths. Lastly, if the $i(t)$ is high, the subnetworks will reuse more features between each other, as discussed in MIMO. In detail, we noticed that to balance high N,K, the HPO would decrease $\alpha(t)$ and $T(t)$ and increase

$i(t)$ and vice versa. We want to stress that HPO chose different start and end points of the $\alpha(t)$, $T(t)$ and $i(t)$ demonstrating that for the best performance, the hyperparameters need to be scheduled to allow for the phase transitions between data fitting and regularisation.



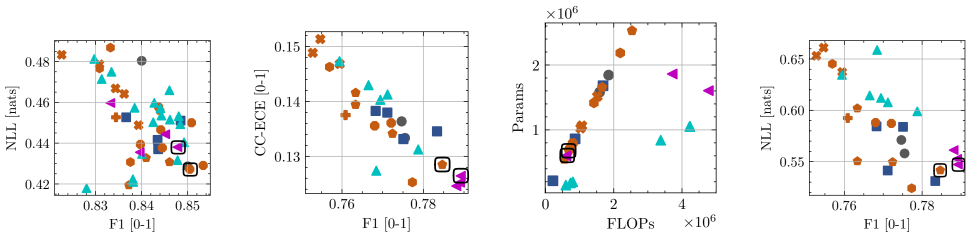
(a) ID ACC, NLL (b) OOD ACC, CAL (c) OOD FLOPs, Params (d) OOD ACC, NLL

Figure 5: Comparison on ID and OOD test sets for TinyImageNet, with respect to **Standard NN** ●, **NN Ensemble** ■, **SAE: I/B:** $N \geq 2, 2 \leq K < D$ ✚ , **EE:** $N = 1, K \geq 2$ ⬢ , **MIMMO:** $N \geq 2, K = D$ ⬟ , **MIMO:** $N \geq 2, K = 1$ ⬣ , **SE NN:** $N = 1, K = 1$ ✖ , **MCD** ▲, **BE** ◀. The black outlines denote the configurations compared in the text.



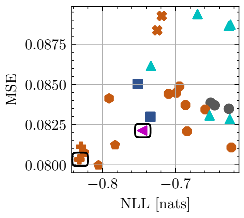
(a) ID ACC, NLL (b) OOD ACC, CAL (c) OOD FLOPs, Params (d) OOD ACC, NLL

Figure 6: Comparison on ID and OOD test sets for BloodMNIST, with respect to **Standard NN** ●, **NN Ensemble** ■, **SAE: I/B:** $N \geq 2, 2 \leq K < D$ ✚ , **EE:** $N = 1, K \geq 2$ ⬢ , **MIMMO:** $N \geq 2, K = D$ ⬟ , **MIMO:** $N \geq 2, K = 1$ ⬣ , **SE NN:** $N = 1, K = 1$ ✖ , **MCD** ▲, **BE** ◀. The black outlines denote the configurations compared in the text.

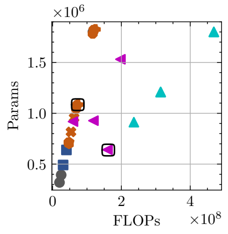


(a) ID ACC, NLL (b) OOD ACC, CAL (c) OOD FLOPs, Params (d) OOD ACC, NLL

Figure 7: Comparison on ID and OOD test sets for PneumoniaMNIST, with respect to **Standard NN** ●, **NN Ensemble** ■, **SAE: I/B:** $N \geq 2, 2 \leq K < D$ ✚ , **EE:** $N = 1, K \geq 2$ ⬢ , **MIMMO:** $N \geq 2, K = D$ ⬟ , **MIMO:** $N \geq 2, K = 1$ ⬣ , **SE NN:** $N = 1, K = 1$ ✖ , **MCD** ▲, **BE** ◀. The black outlines denote the configurations compared in the text.



(a) OOD NLL, MSE



(b) OOD FLOPs, Params

Figure 8: Comparison on OOD test sets for RetinaMNIST, with respect to **Standard NN** ●, **NN Ensemble** ■, **SAE: I/B:** $N \geq 2, 2 \leq K < D$ ✚ , **EE:** $N = 1, K \geq 2$ ⯃ , **MIMMO:** $N \geq 2, K = D$ ⬟ , **MIMO:** $N \geq 2, K = 1$ ⬢ , **SE NN:** $N = 1, K = 1$ ✖ , **MCD** ▲, **BE** ◀. The black outlines denote the configurations compared in the text.

METHOD	NLL	ACCURACY/F1	ECE/CC-ECE	FLOPS	PARAMS
STANDARD NN ●	5.67 ± 0.47	4.67 ± 2.05	6.00 ± 1.63	**1.00 ± 0.00**	3.00 ± 0.00
NN ENSEMBLE ■	**3.00 ± 1.41**	**2.67 ± 1.25**	3.67 ± 2.49	7.00 ± 2.16	6.33 ± 2.05
MCD ▲	7.67 ± 1.89	9.00 ± 0.00	4.00 ± 2.83	8.00 ± 0.82	**1.00 ± 0.00**
BE ◄	6.00 ± 2.16	6.33 ± 1.25	6.33 ± 0.47	7.33 ± 0.47	2.67 ± 0.94
MIMO; $N \geq 2, K = 1$ ⬢	4.00 ± 2.16	4.67 ± 2.05	4.33 ± 0.94	4.33 ± 1.70	6.67 ± 1.25
MIMMO; $N \geq 2, K = D$ ⬟	**3.00 ± 2.83**	4.00 ± 2.45	5.67± 2.36	3.67 ± 1.70	5.33 ± 2.87
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	3.33 ± 1.25	3.67 ± 1.70	**3.00 ± 2.83**	6.67 ± 1.70	8.33 ± 0.47
SE NN; $N = 1, K = 1$ ✖	7.67 ± 1.25	7.33 ± 0.94	6.33 ± 3.09	3.00 ± 0.82	5.00 ± 0.82
EE; $N = 1, K \geq 2$ ⬣	4.67 ± 2.49	**2.67 ± 1.25**	5.67 ± 2.49	4.00 ± 0.82	6.67 ± 0.47

Table 2: Average ranking across all classification ID test datasets. We took all the Pareto candidates for a method and a dataset, calculated the mean performance of the candidates across all metrics and then ranked the methods with respect to each other and their mean per-metric performance. Then we averaged the ranks across all the datasets. Lower rank is better.

METHOD	NLL	ACCURACY/F1	ECE/CC-ECE	FLOPS	PARAMS
STANDARD NN ●	5.67 ± 1.25	5.00 ± 2.16	4.00 ± 1.41	**2.00 ± 1.41**	3.33 ± 2.05
NN ENSEMBLE ■	4.00 ± 1.63	4.67 ± 2.36	4.67 ± 2.05	6.00 ± 2.45	5.00 ± 2.45
MCD ▲	5.67 ± 1.70	9.00 ± 0.00	**1.33 ± 0.47**	8.00 ± 0.82	**2.33 ± 1.89**
BE ◄	5.67 ± 3.40	5.00 ± 2.94	4.33 ± 2.05	6.33 ± 1.70	3.33 ± 1.25
MIMO; $N \geq 2, K = 1$ ⬢	3.67 ± 0.94	4.67 ± 1.25	3.67 ± 2.05	5.00 ± 2.45	6.67 ± 2.05
MIMMO; $N \geq 2, K = max_K$ ⬟	**1.67 ± 0.94**	**2.00 ± 1.41**	5.33 ± 1.70	5.00 ± 1.41	6.33 ± 2.05
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	3.67 ± 2.36	3.67 ± 2.36	5.67 ± 1.70	7.00 ± 1.63	8.33 ± 0.47
SE NN; $N = 1, K = 1$ ✖	7.67 ± 1.89	6.67 ± 1.25	8.67 ± 0.47	2.33 ± 0.47	4.33 ± 1.25
EE; $N = 1, K \geq 2$ ⬣	7.33 ± 0.94	4.33 ± 0.47	7.33 ± 1.70	3.33 ± 1.70	5.33 ± 2.36

Table 3: Average ranking across all classification OOD test datasets. We took all the Pareto candidates for a method and a dataset, calculated the mean performance of the candidates across all metrics and then ranked the methods with respect to each other and their mean per-metric performance. Then we averaged the ranks across all the datasets. Lower rank is better.

METHOD	NLL	MSE	FLOPS	PARAMS
STANDARD NN ●	6	5	**1**	**1**
NN ENSEMBLE ■	7	2	8	9
MCD ▲	4	6	9	6
BE ◄	9	7	5	4
MIMO; $N \geq 2, K = 1$ ⬢	2	8	2	2
MIMMO; $N \geq 2, K = D$ ⬟	5	4	6	7
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	**1**	3	7	8
SE NN; $N = 1, K = 1$ ✖	3	9	3	3
EE; $N = 1, K \geq 2$ ⬣	8	**1**	4	5

Table 4: Ranking across RetinaMNIST ID test dataset. We took all the Pareto candidates for a method, calculated the mean performance of the candidates across all metrics and then ranked the methods with respect to each other and their mean per-metric performance. Lower rank is better.

METHOD	NLL	MSE	FLOPs	PARAMS
STANDARD NN ●	8	5	3	3
NN ENSEMBLE ■	4	6	**1**	**1**
MCD ▲	7	8	9	8
BE ◀	9	3	8	5
MIMO; $N \geq 2, K = 1$ ⬢	3	7	2	2
MIMMO; $N \geq 2, K = D$ ⬟	2	**1**	5	6
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	**1**	2	7	9
SE NN; $N = 1, K = 1$ ✖	5	9	4	4
EE; $N = 1, K \geq 2$ ⯃	6	4	6	7

Table 5: Ranking across RetinaMNIST OOD test dataset. We took all the Pareto candidates for a method, calculated the mean performance of the candidates across all metrics and then ranked the methods with respect to each other and their mean per-metric performance. Lower rank is better.

METHOD	COUNT	NLL [NATS]	ACCURACY [0-1]	ECE [0-1]	FLOPs [M]	PARAMS [M]
STANDARD NN ●	4	2.540 ± 0.006	0.396 ± 0.001	0.010 ± 0.002	**240.137 ± 0.000**	**0.241 ± 0.000**
		3.796 ± 0.025	0.531 ± 0.001	0.330 ± 0.002	963.251 ± 0.000	8.218 ± 0.000
		2.540 ± 0.006	0.396 ± 0.001	0.010 ± 0.002	240.137 ± 0.000	0.241 ± 0.000
NN ENSEMBLE ■	3	2.022 ± 0.008	0.556 ± 0.001	0.037 ± 0.002	2105.251 ± 0.000	5.481 ± 0.000
		2.388 ± 0.006	**0.597 ± 0.002**	0.083 ± 0.003	3853.002 ± 0.000	32.873 ± 0.000
		2.538 ± 0.015	0.397 ± 0.003	0.009 ± 0.002	240.137 ± 0.000	0.241 ± 0.000
MCD ▲	19	2.086 ± 0.009	0.505 ± 0.003	0.015 ± 0.003	2110.102 ± 0.000	1.370 ± 0.000
		2.744 ± 0.020	0.530 ± 0.002	0.172 ± 0.004	3858.836 ± 0.000	8.218 ± 0.000
		5.139 ± 0.002	0.012 ± 0.001	**0.003 ± 0.001**	3858.836 ± 0.000	8.218 ± 0.000
BE ◀	7	2.520 ± 0.005	0.407 ± 0.003	0.032 ± 0.005	483.445 ± 0.000	0.244 ± 0.000
		3.980 ± 0.020	0.530 ± 0.004	0.338 ± 0.002	3863.801 ± 0.000	8.259 ± 0.000
		2.523 ± 0.011	0.406 ± 0.002	0.027 ± 0.003	966.889 ± 0.000	0.247 ± 0.000
MIMO; $N \geq 2, K = 1$ ⬢	2	2.071 ± 0.048	0.505 ± 0.009	0.051 ± 0.014	1081.378 ± 81.796	12.078 ± 5.669
		2.522 ± 0.048	0.511 ± 0.004	0.210 ± 0.003	1180.377 ± 0.000	21.488 ± 0.000
		2.071 ± 0.048	0.505 ± 0.009	0.051 ± 0.014	1081.378 ± 81.796	12.078 ± 5.669
MIMMO; $N \geq 2, K = D$ ⬟	4	1.807 ± 0.013	0.563 ± 0.005	0.063 ± 0.003	1243.154 ± 0.000	22.338 ± 0.000
		1.808 ± 0.003	0.569 ± 0.001	0.046 ± 0.002	1243.154 ± 0.000	22.338 ± 0.000
		1.815 ± 0.016	0.561 ± 0.004	0.026 ± 0.003	1243.051 ± 0.000	22.235 ± 0.000
SE NN; $N = 1, K = 1$ ✖	3	3.566 ± 0.056	0.474 ± 0.004	0.319 ± 0.006	1173.197 ± 0.000	21.384 ± 0.000
		4.252 ± 0.011	0.488 ± 0.002	0.363 ± 0.002	1173.197 ± 0.000	21.384 ± 0.000
		3.566 ± 0.056	0.474 ± 0.004	0.319 ± 0.006	1173.197 ± 0.000	21.384 ± 0.000
EE; $N = 1, K \geq 2$ ⯃	2	3.596 ± 0.020	0.526 ± 0.005	0.337 ± 0.092	1181.852 ± 0.000	21.619 ± 0.000
		3.596 ± 0.020	0.526 ± 0.005	0.337 ± 0.092	1181.852 ± 0.000	21.619 ± 0.000
		3.596 ± 0.020	0.526 ± 0.005	0.337 ± 0.092	1181.852 ± 0.000	21.619 ± 0.000
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	5	***1.801 ± 0.026***	*0.564 ± 0.009*	*0.040 ± 0.006*	*1206.773 ± 0.000*	*22.098 ± 0.000*
		1.801 ± 0.026	0.564 ± 0.009	0.040 ± 0.006	1206.773 ± 0.000	22.098 ± 0.000
		2.068 ± 0.010	0.497 ± 0.003	0.018 ± 0.001	1196.418 ± 0.000	22.033 ± 0.000

Table 6: Best results for empirically Pareto optimal configurations on TinyImageNet ID dataset. Each row represents the best configuration for each method for the first metric, the second metric, etc. The count represents how many configurations were in the Pareto front. The *italic* and underlined values represent the compared configurations in the main text, respectively. The **bold** values represent the best value across all methods for each metric.

METHOD	COUNT	NLL [NATS]	ACCURACY [0-1]	ECE [0-1]	FLOPS [M]	PARAMS [M]
STANDARD NN ●	3	4.803 ± 1.586	0.164 ± 0.104	0.157 ± 0.094	$\mathbf{240.137} \pm \mathbf{0.000}$	$\mathbf{0.241} \pm \mathbf{0.000}$
		7.033 ± 2.485	0.298 ± 0.135	0.497 ± 0.105	1173.197 ± 0.000	21.384 ± 0.000
		4.803 ± 1.586	0.164 ± 0.104	0.157 ± 0.094	240.137 ± 0.000	0.241 ± 0.000
NN ENSEMBLE ■	3	4.387 ± 1.809	0.284 ± 0.151	0.143 ± 0.090	2105.251 ± 0.000	5.481 ± 0.000
		$\underline{5.219 \pm 2.330}$	$\underline{\mathbf{0.342} \pm \mathbf{0.158}}$	$\underline{0.211 \pm 0.100}$	$\underline{3853.002 \pm 0.000}$	$\underline{32.873 \pm 0.000}$
		4.477 ± 1.460	0.182 ± 0.115	0.102 ± 0.070	960.548 ± 0.000	0.964 ± 0.000
MCD ▲	26	4.180 ± 1.517	0.240 ± 0.136	0.134 ± 0.097	2110.102 ± 0.000	1.370 ± 0.000
		5.731 ± 2.304	0.282 ± 0.143	0.317 ± 0.099	3858.836 ± 0.000	8.218 ± 0.000
		5.158 ± 0.019	0.012 ± 0.001	$\mathbf{0.002} \pm \mathbf{0.001}$	3858.836 ± 0.000	8.218 ± 0.000
BE ◀	8	5.050 ± 1.891	0.168 ± 0.107	0.189 ± 0.112	966.889 ± 0.000	0.247 ± 0.000
		7.302 ± 2.542	0.298 ± 0.136	0.446 ± 0.069	3528.084 ± 0.000	21.435 ± 0.000
		5.050 ± 1.891	0.168 ± 0.107	0.189 ± 0.112	966.889 ± 0.000	0.247 ± 0.000
MIMO; $N \geq 2, K = 1$ ⬢	1	3.902 ± 1.371	0.263 ± 0.136	0.122 ± 0.078	1081.378 ± 81.796	12.078 ± 5.669
		3.902 ± 1.371	0.263 ± 0.136	0.122 ± 0.078	1081.378 ± 81.796	12.078 ± 5.669
		3.902 ± 1.371	0.263 ± 0.136	0.122 ± 0.078	1081.378 ± 81.796	12.078 ± 5.669
MIMMO; $N \geq 2, K = D$ ⬟	5	$\mathbf{3.686} \pm \mathbf{1.425}$	0.301 ± 0.151	0.138 ± 0.087	1243.051 ± 0.000	22.235 ± 0.000
		3.704 ± 1.445	0.306 ± 0.153	0.129 ± 0.086	1243.154 ± 0.000	22.338 ± 0.000
		3.697 ± 1.420	0.296 ± 0.152	0.106 ± 0.078	1243.154 ± 0.000	22.338 ± 0.000
SE NN; $N = 1, K = 1$ ✖	4	7.324 ± 2.499	0.223 ± 0.127	0.494 ± 0.091	1173.197 ± 0.000	21.384 ± 0.000
		8.162 ± 2.635	0.235 ± 0.129	0.546 ± 0.098	1173.197 ± 0.000	21.384 ± 0.000
		7.324 ± 2.499	0.223 ± 0.127	0.494 ± 0.091	1173.197 ± 0.000	21.384 ± 0.000
EE; $N = 1, K \geq 2$ ⯃	1	7.375 ± 2.668	0.268 ± 0.143	0.462 ± 0.089	1181.852 ± 0.000	21.619 ± 0.000
		7.375 ± 2.668	0.268 ± 0.143	0.462 ± 0.089	1181.852 ± 0.000	21.619 ± 0.000
		7.375 ± 2.668	0.268 ± 0.143	0.462 ± 0.089	1181.852 ± 0.000	21.619 ± 0.000
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	5	*3.723 ± 1.443*	*0.303 ± 0.151*	*0.164 ± 0.088*	*1206.773 ± 0.000*	*22.098 ± 0.000*
		3.723 ± 1.443	0.303 ± 0.151	0.164 ± 0.088	1206.773 ± 0.000	22.098 ± 0.000
		3.849 ± 1.334	0.259 ± 0.133	0.120 ± 0.092	1221.467 ± 0.044	22.640 ± 0.044

Table 7: Best results for empirically Pareto optimal configurations on TinyImageNet OOD dataset. Each row represents the best configuration for each method for the first metric, the second metric, etc. The count represents how many configurations were in the Pareto front. The *italic* and underlined values represent the compared configurations in the main text, respectively. The **bold** values represent the best value across all methods for each metric.

METHOD	COUNT	NLL [NATS]	F1 [0-1]	CC-ECE [0-1]	FLOPS [M]	PARAMS [M]
STANDARD NN ●	1	0.129 ± 0.004	0.950 ± 0.003	0.023 ± 0.003	**4.697 ± 0.000**	0.246 ± 0.000
		0.129 ± 0.004	0.950 ± 0.003	0.023 ± 0.003	4.697 ± 0.000	0.246 ± 0.000
		0.129 ± 0.004	0.950 ± 0.003	0.023 ± 0.003	4.697 ± 0.000	0.246 ± 0.000
NN ENSEMBLE ■	6	**0.102 ± 0.002**	**0.963 ± 0.001**	0.023 ± 0.002	40.102 ± 0.000	2.315 ± 0.000
		0.103 ± 0.003	0.963 ± 0.001	0.023 ± 0.002	35.371 ± 0.000	1.134 ± 0.000
		0.115 ± 0.003	0.958 ± 0.001	**0.020 ± 0.001**	17.686 ± 0.000	0.567 ± 0.000
MCD ▲	5	0.127 ± 0.002	0.949 ± 0.001	0.023 ± 0.001	18.820 ± 0.000	0.246 ± 0.000
		0.129 ± 0.002	0.950 ± 0.003	0.020 ± 0.001	18.820 ± 0.000	0.246 ± 0.000
		0.129 ± 0.002	0.950 ± 0.003	0.020 ± 0.001	18.820 ± 0.000	0.246 ± 0.000
BE ◀	8	0.132 ± 0.002	0.951 ± 0.004	0.022 ± 0.002	14.218 ± 0.000	0.249 ± 0.000
		0.143 ± 0.004	0.951 ± 0.004	0.028 ± 0.001	30.243 ± 0.000	0.583 ± 0.000
		0.132 ± 0.002	0.951 ± 0.004	0.022 ± 0.002	14.218 ± 0.000	0.249 ± 0.000
MIMO; $N \geq 2, K = 1$ ⬢	2	*0.122 ± 0.005*	*0.956 ± 0.004*	*0.022 ± 0.001*	*11.424 ± 0.000*	*0.600 ± 0.000*
		0.122 ± 0.005	0.956 ± 0.004	0.022 ± 0.001	11.424 ± 0.000	0.600 ± 0.000
		0.122 ± 0.005	0.956 ± 0.004	0.022 ± 0.001	11.424 ± 0.000	0.600 ± 0.000
MIMMO; $N \geq 2, K = D$ ⬟	1	0.135 ± 0.006	0.948 ± 0.003	0.029 ± 0.001	5.621 ± 0.289	**0.212 ± 0.072**
		0.135 ± 0.006	0.948 ± 0.003	0.029 ± 0.001	5.621 ± 0.289	0.212 ± 0.072
		0.135 ± 0.006	0.948 ± 0.003	0.029 ± 0.001	5.621 ± 0.289	0.212 ± 0.072
SE NN; $N = 1, K = 1$ ✖	1	0.133 ± 0.006	0.948 ± 0.004	0.022 ± 0.001	9.468 ± 0.558	0.440 ± 0.139
		0.133 ± 0.006	0.948 ± 0.004	0.022 ± 0.001	9.468 ± 0.558	0.440 ± 0.139
		0.133 ± 0.006	0.948 ± 0.004	0.022 ± 0.001	9.468 ± 0.558	0.440 ± 0.139
EE; $N = 1, K \geq 2$ ⯃	4	0.123 ± 0.006	0.953 ± 0.004	0.025 ± 0.004	10.465 ± 0.132	0.610 ± 0.002
		0.123 ± 0.005	0.954 ± 0.004	0.024 ± 0.002	10.237 ± 0.000	0.606 ± 0.000
		0.132 ± 0.002	0.948 ± 0.001	0.021 ± 0.001	5.789 ± 0.000	0.336 ± 0.000
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	3	0.124 ± 0.003	0.951 ± 0.002	0.024 ± 0.002	10.539 ± 0.000	0.599 ± 0.000
		0.125 ± 0.006	0.954 ± 0.003	0.026 ± 0.003	11.499 ± 0.071	0.608 ± 0.004
		0.124 ± 0.003	0.951 ± 0.002	0.024 ± 0.002	10.539 ± 0.000	0.599 ± 0.000

Table 8: Best results for empirically Pareto optimal configurations on BloodMNIST ID dataset. Each row represents the best configuration for each method for the first metric, the second metric, etc. The count represents how many configurations were in the Pareto front. The *italic* and underlined values represent the compared configurations in the main text, respectively. The **bold** values represent the best value across all methods for each metric.

METHOD	COUNT	NLL [NATS]	F1 [0-1]	CC-ECE [0-1]	FLOPS [M]	PARAMS [M]
		1.478 ± 1.255	0.560 ± 0.258	0.187 ± 0.092	0.737 ± 0.000	0.037 ± 0.000
STANDARD NN ●	5	1.478 ± 1.255	0.560 ± 0.258	0.187 ± 0.092	0.737 ± 0.000	0.037 ± 0.000
		1.959 ± 0.990	0.223 ± 0.080	0.140 ± 0.025	**0.128** ± **0.000**	**0.000** ± **0.000**
		1.206 ± 1.040	0.582 ± 0.269	0.198 ± 0.078	2.948 ± 0.000	0.147 ± 0.000
NN ENSEMBLE ■	17	1.206 ± 1.040	0.582 ± 0.269	0.198 ± 0.078	2.948 ± 0.000	0.147 ± 0.000
		2.006 ± 1.530	0.225 ± 0.109	0.129 ± 0.030	0.256 ± 0.000	0.000 ± 0.000
		1.571 ± 1.203	0.498 ± 0.221	0.182 ± 0.065	2.962 ± 0.000	0.037 ± 0.000
MCD ▲	11	1.571 ± 1.203	0.498 ± 0.221	0.182 ± 0.065	2.962 ± 0.000	0.037 ± 0.000
		2.005 ± 0.001	0.065 ± 0.004	**0.011** ± **0.005**	0.842 ± 0.000	0.000 ± 0.000
		1.589 ± 1.425	0.553 ± 0.262	0.188 ± 0.095	2.259 ± 0.000	0.038 ± 0.000
BE ◀	13	1.589 ± 1.425	0.553 ± 0.262	0.188 ± 0.095	2.259 ± 0.000	0.038 ± 0.000
		1.921 ± 1.127	0.249 ± 0.077	0.142 ± 0.026	0.270 ± 0.000	0.000 ± 0.000
		1.335 ± 1.217	0.543 ± 0.265	0.201 ± 0.074	3.327 ± 0.009	0.150 ± 0.002
MIMO; $N \geq 2, K = 1$ ⬢	4	1.485 ± 1.445	0.556 ± 0.289	0.190 ± 0.091	6.767 ± 0.019	0.334 ± 0.005
		1.485 ± 1.445	0.556 ± 0.289	0.190 ± 0.091	6.767 ± 0.019	0.334 ± 0.005
		1.194 ± **0.849**	0.528 ± 0.233	0.208 ± 0.044	0.741 ± 0.005	0.019 ± 0.000
MIMMO; $N \geq 2, K = D$ ⬟	4	1.277 ± 1.422	**0.587** ± **0.281**	0.179 ± 0.094	3.116 ± 0.000	0.153 ± 0.000
		1.277 ± 1.422	0.587 ± 0.281	0.179 ± 0.094	3.116 ± 0.000	0.153 ± 0.000
		1.866 ± 1.869	0.521 ± 0.267	0.201 ± 0.099	0.719 ± 0.030	0.032 ± 0.008
SE NN; $N = 1, K = 1$ ✖	1	1.866 ± 1.869	0.521 ± 0.267	0.201 ± 0.099	0.719 ± 0.030	0.032 ± 0.008
		1.866 ± 1.869	0.521 ± 0.267	0.201 ± 0.099	0.719 ± 0.030	0.032 ± 0.008
		1.636 ± 1.412	0.516 ± 0.272	0.229 ± 0.055	0.792 ± 0.020	0.040 ± 0.000
EE; $N = 1, K \geq 2$ ⬣	5	2.158 ± 2.128	0.541 ± 0.290	0.201 ± 0.095	5.213 ± 0.000	0.279 ± 0.000
		1.664 ± 1.409	0.537 ± 0.283	0.197 ± 0.094	2.672 ± 0.000	0.150 ± 0.000
		1.387 ± 1.373	0.547 ± 0.266	0.198 ± 0.069	1.083 ± 0.005	0.040 ± 0.000
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	3	*1.451 ± 1.529*	*0.554 ± 0.265*	*0.185 ± 0.083*	*1.075 ± 0.000*	*0.040 ± 0.000*
		1.451 ± 1.529	0.554 ± 0.265	0.185 ± 0.083	1.075 ± 0.000	0.040 ± 0.000

Table 9: Best results for empirically Pareto optimal configurations on BloodMNIST OOD dataset. Each row represents the best configuration for each method for the first metric, the second metric, etc. The count represents how many configurations were in the Pareto front. The *italic* and underlined values represent the compared configurations in the main text, respectively. The **bold** values represent the best value across all methods for each metric.

METHOD	COUNT	NLL [NATS]	F1 [0-1]	CC-ECE [0-1]	FLOPS [M]	PARAMS [M]
		0.429 ± 0.015	0.850 ± 0.007	0.107 ± 0.006	1.850 ± 0.000	1.844 ± 0.000
STANDARD NN ●	2	0.429 ± 0.015	0.850 ± 0.007	0.107 ± 0.006	1.850 ± 0.000	1.844 ± 0.000
		0.429 ± 0.015	0.850 ± 0.007	0.107 ± 0.006	1.850 ± 0.000	1.844 ± 0.000
		0.437 ± 0.018	0.844 ± 0.006	0.106 ± 0.002	**0.217 ± 0.000**	0.215 ± 0.000
NN ENSEMBLE ■	4	0.451 ± 0.024	0.848 ± 0.011	0.112 ± 0.003	2.111 ± 0.000	2.106 ± 0.000
		0.437 ± 0.018	0.844 ± 0.006	0.106 ± 0.002	0.217 ± 0.000	0.215 ± 0.000
		0.418 ± 0.017	0.828 ± 0.003	0.112 ± 0.008	0.595 ± 0.000	**0.148 ± 0.000**
MCD ▲	20	0.440 ± 0.030	0.849 ± 0.008	0.108 ± 0.004	4.229 ± 0.000	1.053 ± 0.000
		0.678 ± 0.002	0.385 ± 0.000	**0.042 ± 0.002**	0.595 ± 0.000	0.148 ± 0.000
		0.436 ± 0.011	0.840 ± 0.010	0.109 ± 0.009	0.450 ± 0.000	0.153 ± 0.000
BE ◀	4	0.438 ± 0.018	0.848 ± 0.007	0.106 ± 0.010	3.388 ± 0.000	0.861 ± 0.000
		$\underline{0.444 \pm 0.026}$	$\underline{0.845 \pm 0.014}$	$\underline{0.103 \pm 0.010}$	$\underline{4.774 \pm 0.000}$	$\underline{1.604 \pm 0.000}$
		0.431 ± 0.010	0.838 ± 0.005	0.105 ± 0.010	1.418 ± 0.033	1.416 ± 0.033
MIMO; $N \geq 2, K = 1$ ⬢	3	0.431 ± 0.010	0.838 ± 0.005	0.105 ± 0.010	1.418 ± 0.033	1.416 ± 0.033
		0.431 ± 0.010	0.838 ± 0.005	0.105 ± 0.010	1.418 ± 0.033	1.416 ± 0.033
		0.419 ± 0.025	0.837 ± 0.008	0.106 ± 0.009	1.660 ± 0.000	1.656 ± 0.000
MIMMO; $N \geq 2, K = D$ ⬟	4	0.431 ± 0.017	0.846 ± 0.005	0.104 ± 0.009	2.062 ± 0.075	2.057 ± 0.075
		0.431 ± 0.017	0.846 ± 0.005	0.104 ± 0.009	2.062 ± 0.075	2.057 ± 0.075
		0.449 ± 0.029	0.839 ± 0.016	0.109 ± 0.009	1.519 ± 0.220	1.515 ± 0.219
SE NN; $N = 1, K = 1$ ✖	5	0.449 ± 0.029	0.839 ± 0.016	0.109 ± 0.009	1.519 ± 0.220	1.515 ± 0.219
		0.449 ± 0.029	0.839 ± 0.016	0.109 ± 0.009	1.519 ± 0.220	1.515 ± 0.219
		0.427 ± 0.029	*0.850 ± 0.013*	*0.097 ± 0.009*	*2.115 ± 0.000*	*2.109 ± 0.000*
EE; $N = 1, K \geq 2$ ⯃	7	0.429 ± 0.028	**0.853 ± 0.008**	0.105 ± 0.008	2.645 ± 0.000	2.638 ± 0.000
		0.427 ± 0.029	0.850 ± 0.013	0.097 ± 0.009	2.115 ± 0.000	2.109 ± 0.000
		0.453 ± 0.058	0.834 ± 0.018	0.104 ± 0.017	4.094 ± 0.132	4.088 ± 0.132
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	1	0.453 ± 0.058	0.834 ± 0.018	0.104 ± 0.017	4.094 ± 0.132	4.088 ± 0.132
		0.453 ± 0.058	0.834 ± 0.018	0.104 ± 0.017	4.094 ± 0.132	4.088 ± 0.132

Table 10: Best results for empirically Pareto optimal configurations on PneumoniaMNIST ID dataset. Each row represents the best configuration for each method for the first metric, the second metric, etc. The count represents how many configurations were in the Pareto front. The *italic* and $\underline{\text{underlined}}$ values represent the compared configurations in the main text, respectively. The **bold** values represent the best value across all methods for each metric.

Method	Count	NLL [nats]	F1 [0-1]	CC-ECE [0-1]	FLOPs [M]	Params [M]
Standard NN ●	2	0.558 ± 0.184	0.775 ± 0.100	0.133 ± 0.041	1.585 ± 0.000	1.581 ± 0.000
		0.558 ± 0.184	0.775 ± 0.100	0.133 ± 0.041	1.585 ± 0.000	1.581 ± 0.000
		0.558 ± 0.184	0.775 ± 0.100	0.133 ± 0.041	1.585 ± 0.000	1.581 ± 0.000
NN Ensemble ■	4	0.531 ± 0.195	0.783 ± 0.096	0.135 ± 0.042	0.867 ± 0.000	0.861 ± 0.000
		0.531 ± 0.195	0.783 ± 0.096	0.135 ± 0.042	0.867 ± 0.000	0.861 ± 0.000
		0.584 ± 0.247	0.775 ± 0.100	0.133 ± 0.044	$\mathbf{0.217 \pm 0.000}$	0.215 ± 0.000
MCD ▲	8	0.599 ± 0.415	0.779 ± 0.104	0.131 ± 0.062	0.733 ± 0.000	0.182 ± 0.000
		0.599 ± 0.415	0.779 ± 0.104	0.131 ± 0.062	0.733 ± 0.000	0.182 ± 0.000
		0.678 ± 0.005	0.385 ± 0.000	$\mathbf{0.049 \pm 0.021}$	0.595 ± 0.000	$\mathbf{0.148 \pm 0.000}$
BE ◀	3	0.547 ± 0.171	$\mathbf{0.789 \pm 0.082}$	0.126 ± 0.042	0.601 ± 0.000	0.599 ± 0.000
		0.547 ± 0.171	0.789 ± 0.082	0.126 ± 0.042	0.601 ± 0.000	0.599 ± 0.000
		0.561 ± 0.210	0.788 ± 0.089	0.125 ± 0.045	4.774 ± 0.000	1.604 ± 0.000
MIMO; $N \geq 2, K = 1$ ⬢	2	$\mathbf{0.524 \pm 0.223}$	0.777 ± 0.098	0.125 ± 0.050	1.418 ± 0.033	1.416 ± 0.033
		0.524 ± 0.223	0.777 ± 0.098	0.125 ± 0.050	1.418 ± 0.033	1.416 ± 0.033
		0.524 ± 0.223	0.777 ± 0.098	0.125 ± 0.050	1.418 ± 0.033	1.416 ± 0.033
MIMMO; $N \geq 2, K = D$ ⬟	4	*0.542 ± 0.299*	*0.785 ± 0.108*	*0.128 ± 0.063*	*0.682 ± 0.000*	*0.680 ± 0.000*
		0.542 ± 0.299	0.785 ± 0.108	0.128 ± 0.063	0.682 ± 0.000	0.680 ± 0.000
		0.542 ± 0.299	0.785 ± 0.108	0.128 ± 0.063	0.682 ± 0.000	0.680 ± 0.000
SE NN; $N = 1, K = 1$ ✖	3	0.637 ± 0.280	0.759 ± 0.107	0.147 ± 0.051	1.519 ± 0.220	1.515 ± 0.219
		0.637 ± 0.280	0.759 ± 0.107	0.147 ± 0.051	1.519 ± 0.220	1.515 ± 0.219
		0.637 ± 0.280	0.759 ± 0.107	0.147 ± 0.051	1.519 ± 0.220	1.515 ± 0.219
EE; $N = 1, K \geq 2$ ⯃	2	0.587 ± 0.238	0.772 ± 0.110	0.136 ± 0.053	0.715 ± 0.029	0.711 ± 0.029
		0.587 ± 0.238	0.772 ± 0.110	0.136 ± 0.053	0.715 ± 0.029	0.711 ± 0.029
		0.588 ± 0.257	0.768 ± 0.108	0.136 ± 0.045	0.798 ± 0.000	0.795 ± 0.000
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	1	0.592 ± 0.248	0.761 ± 0.110	0.138 ± 0.051	4.094 ± 0.132	4.088 ± 0.132
		0.592 ± 0.248	0.761 ± 0.110	0.138 ± 0.051	4.094 ± 0.132	4.088 ± 0.132
		0.592 ± 0.248	0.761 ± 0.110	0.138 ± 0.051	4.094 ± 0.132	4.088 ± 0.132

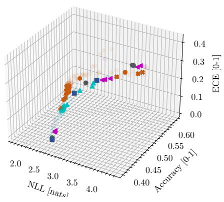
Table 11: Best results for empirically Pareto optimal configurations on PneumoniaMNIST OOD dataset. Each row represents the best configuration for each method for the first metric, the second metric, etc. The count represents how many configurations were in the Pareto front. The *italic* and underlined values represent the compared configurations in the main text, respectively. The **bold** values represent the best value across all methods for each metric.

Method	Count	NLL [nats]	MSE	FLOPs [M]	Params [M]
Standard NN ●	3	-0.891 ± 0.008	0.069 ± 0.001	$\mathbf{15.315 \pm 0.000}$	$\mathbf{0.246 \pm 0.000}$
		-0.843 ± 0.032	0.068 ± 0.001	58.925 ± 0.000	0.912 ± 0.000
NN Ensemble ■	9	-0.901 ± 0.015	0.068 ± 0.000	49.949 ± 0.000	0.787 ± 0.000
		-0.858 ± 0.017	$\mathbf{0.067 \pm 0.000}$	220.672 ± 0.000	3.403 ± 0.000
MCD ▲	6	-0.898 ± 0.015	0.068 ± 0.000	488.560 ± 0.000	1.875 ± 0.000
		-0.898 ± 0.015	0.068 ± 0.000	488.560 ± 0.000	1.875 ± 0.000
BE ◀	3	-0.880 ± 0.027	0.069 ± 0.001	44.969 ± 0.000	0.697 ± 0.000
		-0.880 ± 0.027	0.069 ± 0.001	44.969 ± 0.000	0.697 ± 0.000
MIMO; $N \geq 2, K = 1$ ⬢	1	*-0.896 ± 0.007*	*0.070 ± 0.001*	*47.839 ± 11.736*	*0.699 ± 0.171*
		-0.896 ± 0.007	0.070 ± 0.001	47.839 ± 11.736	0.699 ± 0.171
MIMMO; $N \geq 2, K = D$ ⬟	4	-0.890 ± 0.042	0.068 ± 0.001	122.704 ± 0.000	1.824 ± 0.000
		-0.868 ± 0.065	0.068 ± 0.001	122.704 ± 0.000	1.824 ± 0.000
SE NN; $N = 1, K = 1$ ✖	1	-0.886 ± 0.004	0.070 ± 0.000	53.502 ± 12.028	0.817 ± 0.184
		-0.886 ± 0.004	0.070 ± 0.000	53.502 ± 12.028	0.817 ± 0.184
EE; $N = 1, K \geq 2$ ⯃	3	-0.856 ± 0.011	0.067 ± 0.000	88.794 ± 6.317	1.357 ± 0.096
		-0.856 ± 0.011	0.067 ± 0.000	88.794 ± 6.317	1.357 ± 0.096
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	2	$\mathbf{-0.910 \pm 0.044}$	0.068 ± 0.001	122.702 ± 0.000	1.823 ± 0.000
		-0.902 ± 0.049	0.068 ± 0.001	122.702 ± 0.000	1.823 ± 0.000

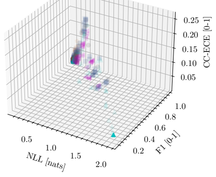
Table 12: Best results for empirically Pareto optimal configurations on RetinaMNIST ID dataset. Each row represents the best configuration for each method for the first metric, the second metric, etc. The count represents how many configurations were in the Pareto front. The *italic* and underlined values represent the compared configurations in the main text, respectively. The **bold** values represent the best value across all methods for each metric.

METHOD	COUNT	NLL [NATS]	MSE	FLOPS [M]	PARAMS [M]
STANDARD NN ●	3	-0.652 ± 0.742	0.084 ± 0.027	**20.145 ± 0.000**	**0.320 ± 0.000**
		-0.627 ± 0.602	0.083 ± 0.030	117.167 ± 0.000	1.800 ± 0.000
NN ENSEMBLE ■	2	-0.752 ± 0.404	0.085 ± 0.028	30.630 ± 0.000	0.492 ± 0.000
		-0.735 ± 0.468	0.083 ± 0.024	40.290 ± 0.000	0.640 ± 0.000
MCD ▲	6	-0.734 ± 0.345	0.086 ± 0.032	236.272 ± 0.000	0.913 ± 0.000
		-0.626 ± 0.542	0.083 ± 0.030	468.671 ± 0.000	1.800 ± 0.000
BE ◀	4	$\underline{-0.746 \pm 0.423}$	$\underline{0.082 \pm 0.023}$	$\underline{159.686 \pm 0.000}$	$\underline{0.642 \pm 0.000}$
		-0.746 ± 0.423	0.082 ± 0.023	159.686 ± 0.000	0.642 ± 0.000
MIMO; $N \geq 2, K = 1$ ⬢	1	-0.791 ± 0.202	0.084 ± 0.028	47.839 ± 11.736	0.699 ± 0.171
		-0.791 ± 0.202	0.084 ± 0.028	47.839 ± 11.736	0.699 ± 0.171
MIMMO; $N \geq 2, K = D$ ⬟	3	-0.825 ± 0.182	0.081 ± 0.026	48.747 ± 0.000	0.718 ± 0.000
		-0.806 ± 0.212	0.080 ± 0.025	122.704 ± 0.000	1.824 ± 0.000
SE NN; $N = 1, K = 1$ ✖	2	-0.725 ± 0.371	0.088 ± 0.033	62.532 ± 28.371	0.955 ± 0.433
		-0.725 ± 0.371	0.088 ± 0.033	62.532 ± 28.371	0.955 ± 0.433
EE; $N = 1, K \geq 2$ ⯃	7	-0.710 ± 0.305	0.084 ± 0.035	68.293 ± 0.000	1.044 ± 0.000
		-0.624 ± 0.362	0.081 ± 0.028	119.037 ± 0.000	1.821 ± 0.000
I/B; $N \geq 2, 2 \leq K \leq D$ ✚	2	***-0.832 ± 0.175***	***0.080 ± 0.025***	*74.009 ± 0.000*	*1.086 ± 0.000*
		-0.832 ± 0.175	0.080 ± 0.025	74.009 ± 0.000	1.086 ± 0.000

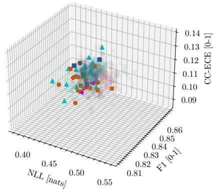
Table 13: Best results for empirically Pareto optimal configurations on RetinaMNIST OOD dataset. Each row represents the best configuration for each method for the first metric, the second metric, etc. The count represents how many configurations were in the Pareto front. The *italic* and $\underline{\text{underlined}}$ values represent the compared configurations in the main text, respectively. The **bold** values represent the best value across all methods for each metric.



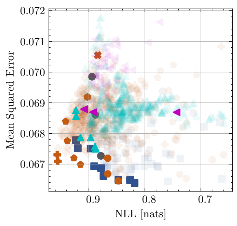
(a) TinyImageNet ID.



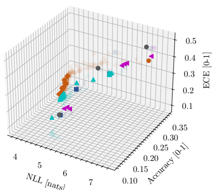
(b) BloodMNIST ID.



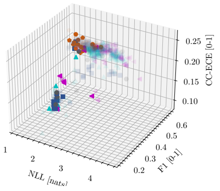
(c) PneumoniaMNIST ID.



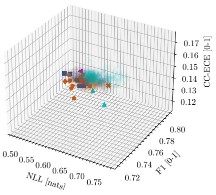
(d) RetinaMNIST ID.



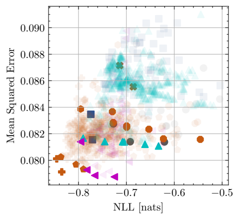
(e) TinyImageNet OOD.



(f) BloodMNIST OOD.

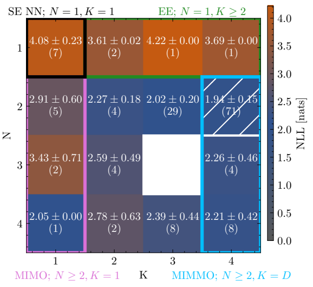


(g) PneumoniaMNIST OOD.

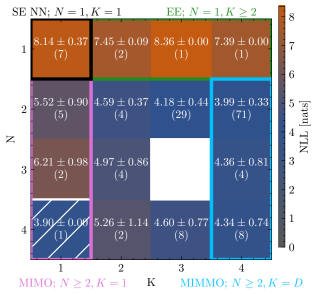


(h) RetinaMNIST OOD.

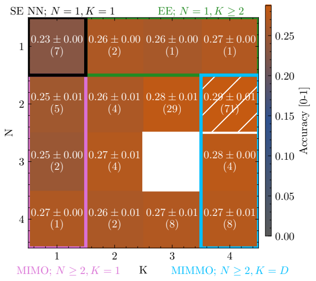
Figure 9: Comparison on TinyImageNet, BloodMNIST, PneumoniaMNIST, and RetinaMNIST across ID (upper row) and OOD (lower row) datasets, with respect to **Standard NN** ●, **NN Ensemble** ■, **SAE: I/B:** $N \geq 2, 2 \leq K < D$ ✚ , **EE:** $N = 1, K \geq 2$ ⬣ , **MIMMO:** $N \geq 2, K = D$ ⬟ , **MIMO:** $N \geq 2, K = 1$ ⬢ , **SE NN:** $N = 1, K = 1$ ✖ , **MCD** ▲, **BE** ◀. The plots contain all the configurations tried by hyperparameter optimisation. The high opacity configurations indicate the configurations used for the final evaluation over repeated seeds.



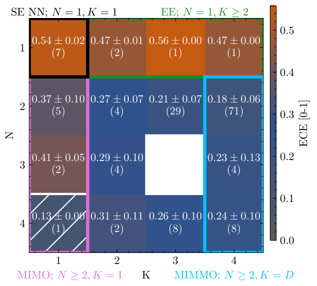
(a) NLL on TinyImageNet ID.



(b) NLL on TinyImageNet OOD.

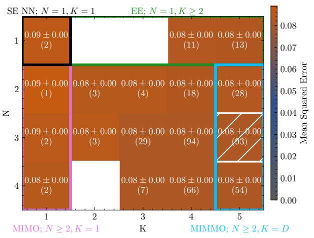


(c) Accuracy on TinyImageNet OOD.

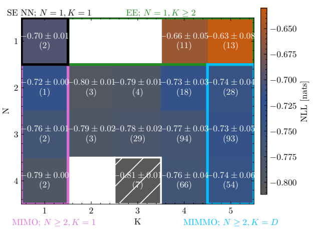


(d) ECE on TinyImageNet OOD.

Figure 10: Varying N, K on TinyImageNet ID and OOD test sets. The upper number is the average performance over N, K combinations. The number in brackets is the number of sampled configurations by HPO. White box means no configurations sampled for that N, K. Pattern signals best average performance. The coloured outlines signal the special cases for the generalised methods.

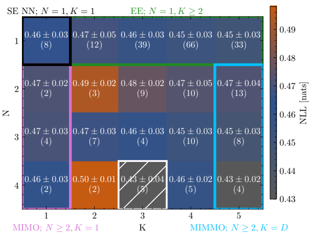


(a) MSE on RetinaMNIST OOD.

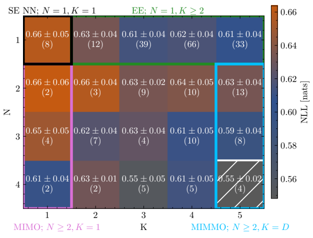


(b) NLL on RetinaMNIST OOD.

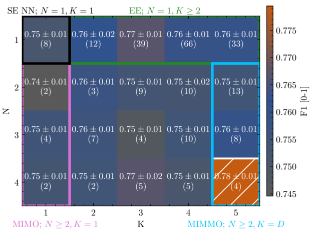
Figure 11: Varying N, K on RetinaMNIST OOD test sets. The upper number is the average performance over N, K combinations. The number in brackets is the number of sampled configurations by HPO. White box means no configurations sampled for that N, K. Pattern signals best average performance. The coloured outlines signal the special cases for the generalised methods.



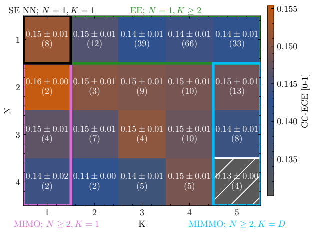
(a) NLL on PneumoniaMNIST ID.



(b) NLL on PneumoniaMNIST OOD.

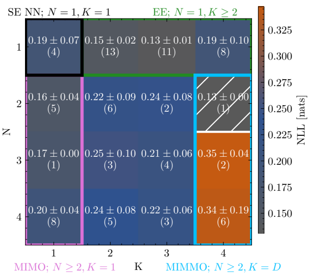


(c) F1 on PneumoniaMNIST OOD.

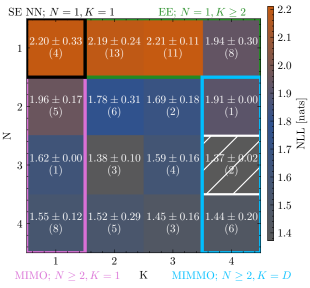
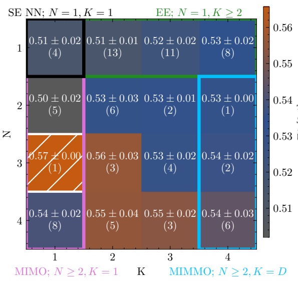
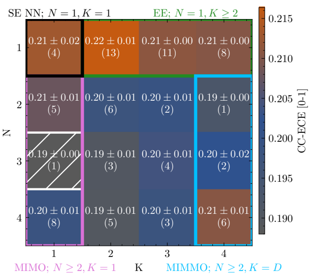


(d) ECE on PneumoniaMNIST OOD.

Figure 12: Varying N, K on PneumoniaMNIST ID and OOD test sets. The upper number is the average performance over N, K combinations. The number in brackets is the number of sampled configurations by HPO. White box means no configurations sampled for that N, K. Pattern signals best average performance. The coloured outlines signal the special cases for the generalised methods.

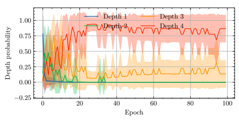
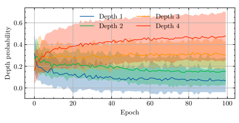
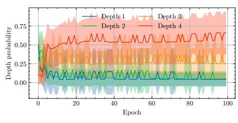
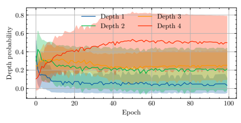


(a) NLL on BloodMNIST ID.

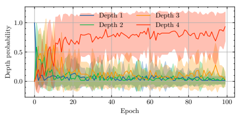
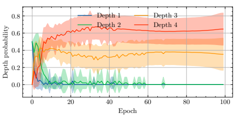
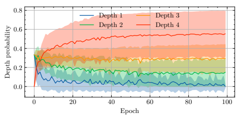
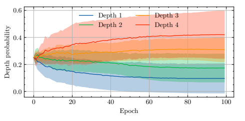




(b) NLL on BloodMNIST OOD. (c) F1 on BloodMNIST OOD. (d) ECE on BloodMNIST OOD.

Figure 13: Varying N, K on BloodMNIST ID and OOD test sets. The upper number is the average performance over N, K combinations. The number in brackets is the number of sampled configurations by HPO. White box means no configurations sampled for that N, K. Pattern signals best average performance. The coloured outlines signal the special cases for the generalised methods.

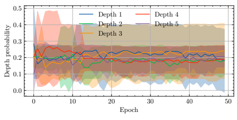





(a) $N = 1$. (b) $N = 2$. (c) $N = 3$. (d) $N = 4$.

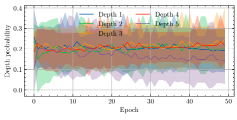





(e) $K = 1$. (f) $K = 2$. (g) $K = 3$. (h) $K = D$.

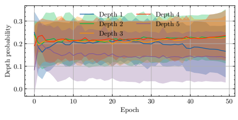
Figure 14: Depth preference during training when averaging over different N and K for TinyImageNet. The lines denote the mean trend, and the shaded regions denote the standard deviation across configurations.



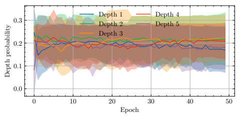
(a) $N = 1$.



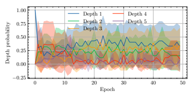
(b) $N = 2$.



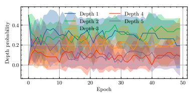
(c) $N = 3$.



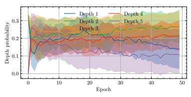
(d) $N = 4$.



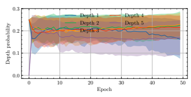
(e) $K = 1$.



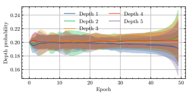
(f) $K = 2$.



(g) $K = 3$.

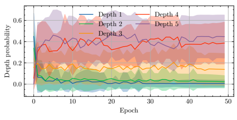


(h) $K = 4$.

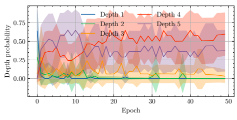


(i) $K = D$.

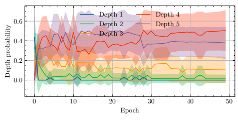
Figure 15: Depth preference during training when averaging over different N and K for RetinaMNIST. The lines denote the mean trend, and the shaded regions denote the standard deviation across configurations.



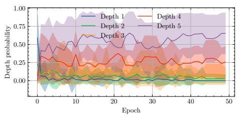
(a) $N = 1$.



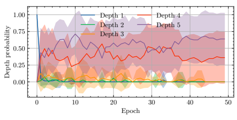
(b) $N = 2$.



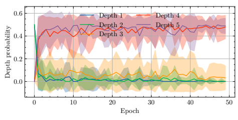
(c) $N = 3$.



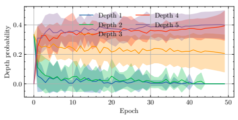
(d) $N = 4$.



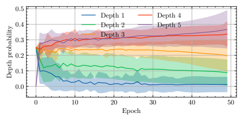
(e) $K = 1$.



(f) $K = 2$.

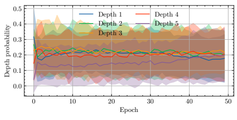


(g) $K = 3$.

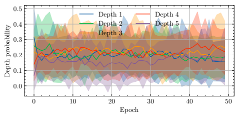


(h) $K = D$.

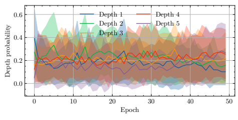
Figure 16: Depth preference during training when averaging over different N and K for BloodMNIST. The lines denote the mean trend, and the shaded regions denote the standard deviation across configurations.



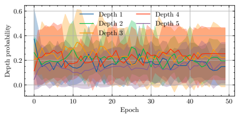
(a) $N = 1$.



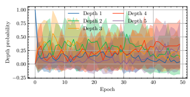
(b) $N = 2$.



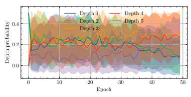
(c) $N = 3$.



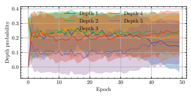
(d) $N = 4$.



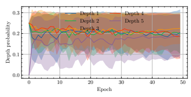
(e) $K = 1$.



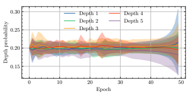
(f) $K = 2$.



(g) $K = 3$.



(h) $K = 4$.



(i) $K = D$.

Figure 17: Depth preference during training when averaging over different N and K for PneumoniaMNIST. The lines denote the mean trend, and the shaded regions denote the standard deviation across configurations.

METRIC	$i_{\text{START}}(\cdot)$	$i_{\text{END}}(\cdot)$	$T_{\text{START}}(\cdot)$	$T_{\text{END}}(\cdot)$	$\alpha_{\text{START}}(\cdot)$	$\alpha_{\text{END}}(\cdot)$	N^c	K^c
NLL	-0.03	0.32	0.27	0.03	-0.26	-0.29	100.09	49.48
ACCURACY	0.24	-0.16	-0.28	-0.07	0.15	0.3	38.78	61.01
ECE	0.27	0.35	0.16	0.06	-0.28	-0.3	46.72	31.64
FLOPS	0.02	-0.12	-0.25	-0.03	0.28	0.45	8.54	42.09
PARAMS	0.07	-0.1	-0.23	-0.01	0.26	0.4	1.19	10.72

Table 14: Correlation of hyperparameters with metrics on the TinyImageNet ID test set. The default correlation measure is Spearman correlation, for the columns with categorical hyperparameters ANOVA is used. i denotes the input repetition probability, T the temperature, α the alpha parameter, N^c the number of members and K^c the maximum number of exits. c stands for categorical variables. The start and end, denote the start and end value of the hyperparameter, respectively.

METRIC	$i_{\text{START}}(\cdot)$	$i_{\text{END}}(\cdot)$	$T_{\text{START}}(\cdot)$	$T_{\text{END}}(\cdot)$	$\alpha_{\text{START}}(\cdot)$	$\alpha_{\text{END}}(\cdot)$	N^c	K^c	W^c
F1	0.31	0.33	-0.05	0.03	0.13	0.06	5.01	1.49	43.7
CC-ECE	–0.3	-0.33	-0.1	-0.1	-0.05	-0.11	5.39	9.75	20.12
NLL	-0.29	-0.35	-0.03	-0.01	-0.06	-0.08	5.81	2.97	47.86
FLOPS	0.2	0.1	-0.08	-0.09	0.06	0.03	0.26	1.51	68.27
PARAMS	0.18	0.16	-0.1	-0.09	0.1	0.04	1.07	0.89	50.71

Table 15: Correlation of hyperparameters with metrics on the BloodMNIST ID test set. The default correlation measure is Spearman correlation, for the columns with categorical hyperparameters ANOVA is used. i denotes the input repetition probability, T the temperature, α the alpha parameter, N^c the number of members, K^c the maximum number of exits and W^c the width multiplier. c stands for categorical variables. The start and end, denote the start and end value of the hyperparameter, respectively.

METRIC	$i_{\text{START}}(\cdot)$	$i_{\text{END}}(\cdot)$	$T_{\text{START}}(\cdot)$	$T_{\text{END}}(\cdot)$	$\alpha_{\text{START}}(\cdot)$	$\alpha_{\text{END}}(\cdot)$	N^c	K^c	W^c
F1	0.01	-0.04	-0.04	0.01	0.01	0.07	8	2.62	3.18
CC-ECE	0.11	0.01	0.01	-0.02	0	-0.02	2.15	0.44	0.5
NLL	0.1	0.02	-0.01	0	0.01	-0.01	4.02	2.11	2.55
FLOPS	-0.01	-0.12	-0.05	-0.1	-0.11	0.13	3.95	0.59	219.72
PARAMS	-0.01	-0.12	-0.04	-0.1	-0.12	0.12	3.97	0.59	219.34

Table 16: Correlation of hyperparameters with metrics on the PneumoniaMNIST ID test set. The default correlation measure is Spearman correlation, for the columns with categorical hyperparameters ANOVA is used. i denotes the input repetition probability, T the temperature, α the alpha parameter, N^c the number of members, K^c the maximum number of exits and W^c the width multiplier. c stands for categorical variables. The start and end, denote the start and end value of the hyperparameter, respectively.

METRIC	$i_{\text{START}}(\cdot)$	$i_{\text{END}}(\cdot)$	$T_{\text{START}}(\cdot)$	$T_{\text{END}}(\cdot)$	$\alpha_{\text{START}}(\cdot)$	$\alpha_{\text{END}}(\cdot)$	N^c	K^c	W^c
NLL	0.33	0.12	-0.12	-0.07	-0.05	0.04	53.86	11.14	5.61
MSE	0.08	0.08	-0.03	0	0.02	-0.06	48.99	18.59	9.39
FLOPS	-0.02	-0.01	0.13	0.05	-0.11	-0.12	69.89	9.45	1268.52
PARAMS	-0.03	-0.02	0.13	0.05	-0.11	-0.12	70.62	9.39	1303.6

Table 17: Correlation of hyperparameters with metrics on the RetinaMNIST ID test set. The default correlation measure is Spearman correlation, for the columns with categorical hyperparameters ANOVA is used. i denotes the input repetition probability, T the temperature, α the alpha parameter, N^c the number of members, K^c the maximum number of exits and W^c the width multiplier. c stands for categorical variables. The start and end, denote the start and end value of the hyperparameter, respectively.