arXiv:2303.08010v1 [cs.LG] 14 Mar 2023

Window-Based Early-Exit Cascades for Uncertainty Estimation: When Deep Ensembles are More Efficient than Single Models

Guoxuan Xia
Imperial College London
g.xia21@imperial.ac.uk

Christos-Savvas Bouganis
Imperial College London
christos-savvas.bouganis@imperial.ac.uk

Abstract

Deep Ensembles are a simple, reliable, and effective method of improving both the predictive performance and uncertainty estimates of deep learning approaches. However, they are widely criticised as being computationally expensive, due to the need to deploy multiple independent models. Recent work has challenged this view, showing that for predictive accuracy, ensembles can be more computationally efficient (at inference) than scaling single models within an architecture family. This is achieved by cascading *ensemble members via an* early-exit *approach. In this work, we investigate extending these efficiency gains to tasks related to uncertainty estimation. As many such tasks,* e.g. *selective classification, are binary classification, our key novel insight is to only pass samples within a* window *close to the binary decision boundary to later cascade stages. Experiments on ImageNet-scale data across a number of network architectures and uncertainty tasks show that the proposed window-based early-exit approach is able to achieve a superior uncertainty-computation trade-off compared to scaling single models. For example, a cascaded EfficientNet-B2 ensemble is able to achieve similar coverage at 5% risk as a single EfficientNet-B4 with <30% the number of MACs. We also find that cascades/ensembles give more* reliable *improvements on OOD data vs scaling models up. Code for this work is available at:* https://github.com/Guoxoug/window-early-exit.

1. Introduction

It is important for deep learning systems to provide reliable estimates of predictive uncertainty in scenarios where it is costly to make mistakes, *e.g.* healthcare [2] and autonomous driving [32]. A model should know when to defer a prediction to a human expert or simply discard it as too uncertain. Deep Ensembles [37], where multiple networks of the same architecture are trained independently and combined at inference, are a simple approach to improving both the predictive and uncertainty-related performance of deep neural networks. Moreover, as it is an easy-to-implement baseline, the approach has accumulated a large volume of empirical evidence in support of its efficacy across a wide range of data types and uncertainty-related tasks [3, 13–15, 17, 21, 33, 44–46, 49, 75, 77].

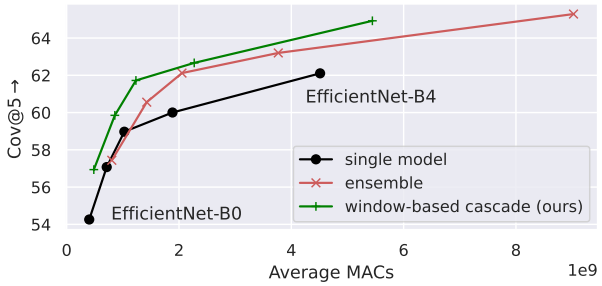

Figure 1. Coverage at risk=5% (selective classification performance) against average computation per sample (MACs). Ensembles are able to outperform scaling single models at higher levels of computation. **Our window-based cascade approach is able to achieve a better uncertainty-computation trade-off compared to single models** via adaptive inference. The models are EfficientNets, ensembles are size $M = 2$ and the dataset is ImageNet-1k.

A common criticism of Deep Ensembles is that they are expensive to deploy [13, 23, 46], as the cost scales linearly with the ensemble size. However, recent work [6, 34, 43, 71, 81], has shown that *ensembling can be more efficient than scaling a single model* when evaluating for predictive accuracy. In particular, Wang et al. [71] show that cascading ensemble members via adaptive early-exiting [38, 67, 72] enables ensembles to obtain consistently better accuracy-computation trade-offs vs scaling single models within an architecture family. This approach involves sequentially running inference on each member and exiting computation early if the ensemble *up to that point* is sufficiently confident. Thus computation is saved by only passing the most difficult samples to the more powerful later cascade stages. Although Wang et al. [71] have shown promising results for accuracy, the

uncertainty-computation trade-off for ensembles remains unexplored. The **key contributions** of this work are:

1. We investigate the unexplored trade-off between *uncertainty-related* performance and computational cost for Deep Ensembles vs scaling single models. We evaluate over multiple downstream tasks: selective classification (SC) [16], out-of-distribution (OOD) detection [78], and selective classification with OOD data (SCOD) [33, 76]. Experiments are performed over a range of ImageNet-scale datasets and convolutional neural network (CNN) architectures. Compared to single-model scaling, we find that Deep Ensembles are more efficient for high levels of compute and give more *reliable* improvements on OOD data.

2. We propose a novel *window-based* early-exit policy, which enables cascaded Deep Ensembles to achieve a *superior uncertainty-computation trade-off* compared to model scaling. Our key insight is that many downstream tasks for uncertainty estimation can be formulated as binary classification. Thus, only samples in a *window* near the binary decision boundary should be evaluated by the more costly but better-performing later cascade stages. Results can be previewed in Fig. 1. We also validate our exit policy on MSDNet [27], a dedicated multi-exit architecture.

2. Preliminaries

2.1. Uncertainty estimation: downstream tasks

In this work, we take a task-oriented view, that motivates the estimation of uncertainties from the perspective of wanting to detect costly or failed predictions [30, 76]. Fig. 2 illustrates the tasks discussed in this section.

Selective classification (SC). Consider a neural network classifier, with parameters $\boldsymbol{\theta}$, that models the conditional distribution $P(y|\boldsymbol{x};\boldsymbol{\theta})$ over K labels $y \in \mathcal{Y} = \{\omega_k\}_{k=1}^{K}$ given inputs $\boldsymbol{x} \in \mathcal{X} = \mathbb{R}^D$. It is trained on dataset $\mathcal{D}_{\text{tr}} = \{y^{(n)}, \boldsymbol{x}^{(n)}\}_{n=1}^{N}$ drawn from the distribution $p_{\text{ID}}(y,\boldsymbol{x})$. This is typically achieved with a categorical softmax output. The classification function f for test input $\boldsymbol{x}^*$ is defined as

$$f(\boldsymbol{x}^*) = \hat{y} = \underset{\omega}{\operatorname{argmax}} P(\omega|\boldsymbol{x}^*;\boldsymbol{\theta}) \ . \tag{1}$$

A *selective classifier* [9] is a function pair (f, g) that includes the aforementioned classifier $f(\boldsymbol{x})$, which yields prediction $\hat{y}$, and a *binary* rejection function,

$$g(\boldsymbol{x};\tau) = \begin{cases} 0 \text{ (reject prediction)}, & \text{if } U(\boldsymbol{x}) > \tau \\ 1 \text{ (accept prediction)}, & \text{if } U(\boldsymbol{x}) \leq \tau \ , \end{cases} \tag{2}$$

where $U(\boldsymbol{x})$ is a measure of predictive uncertainty and τ is an operating threshold. U is typically calculated using the outputs of the neural network. Intuitively, uncertain samples are rejected. The selective risk [16] is then defined as

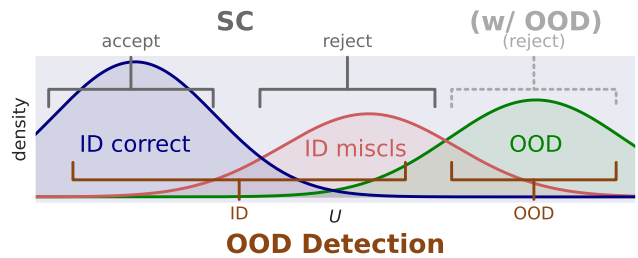

Figure 2. Illustration of uncertainty-related tasks evaluated in this work. **SC** aims to accept correct predictions and reject misclassifications. **OOD detection** aims to separate ID data from OOD data. **SCOD** is **SC** where we additionally want to reject OOD data.

$$\text{Risk}(f,g;\tau) = \frac{\mathbb{E}_{p_{\text{ID}}(\boldsymbol{x})}[g(\boldsymbol{x};\tau)\mathcal{L}_{\text{SC}}(f(\boldsymbol{x}))]}{\mathbb{E}_{p_{\text{ID}}(\boldsymbol{x})}[g(\boldsymbol{x};\tau)]} \ , \tag{3}$$

where

$$\mathcal{L}_{\text{SC}}(f(\boldsymbol{x})) = \begin{cases} 0, & \text{if } f(\boldsymbol{x}) = y \quad \text{(correct)} \\ 1, & \text{if } f(\boldsymbol{x}) \neq y \quad \text{(misclassified)} \ , \end{cases} \tag{4}$$

is the 0/1 error. Intuitively, selective risk measures the average loss over the *accepted* samples. The coverage is defined as $\text{Cov} = \mathbb{E}_{p_{\text{ID}}(\boldsymbol{x})}[g(\boldsymbol{x};\tau)]$, i.e. the proportion of predictions accepted. It is desirable to maximise coverage (accept as many as possible) and minimise risk (reduce the error rate of accepted samples). This is achieved by either improving g, i.e. better rejection of misclassifications, or by improving f, i.e. better accuracy so fewer misclassifications to reject.

SC performance can be either measured using the threshold-free metric, Area Under the Risk-Coverage curve (AURC↓)[1] [16], or be evaluated at an appropriate threshold τ based on a desired criterion [74], *e.g.* coverage at 5% risk (Cov@5↑). In practice, τ can be determined by matching the desired criterion on a held-out validation dataset.

Out-of-distribution (OOD) detection. Regular classification assumes that at deployment, samples originate from the same distribution as the one the classifier was trained on, $(y^*, \boldsymbol{x}^*) \sim p_{\text{ID}}(y,\boldsymbol{x})$. This may not be the case in the real world. OOD detection aims to filter out samples that originate from some distribution $p_{\text{OOD}}(\boldsymbol{x})$ *with labels that are disjoint from* $\mathcal{Y}$ [78].[2] We now assume a mix of in-distribution (ID) and OOD data occurs at deployment,

$$\begin{aligned} &\boldsymbol{x}^* \sim p_{\text{mix}}(\boldsymbol{x}) \\ &p_{\text{mix}}(\boldsymbol{x}) = \alpha p_{\text{ID}}(\boldsymbol{x}) + (1-\alpha) p_{\text{OOD}}(\boldsymbol{x}), \quad \alpha \in [0,1] \ . \end{aligned} \tag{5}$$

[1] Arrows by a metric indicate whether higher (↑) or lower (↓) is better.
[2] Note this is different to *covariate shift* where $\mathcal{Y}$ is preserved [78]

Predictions on OOD data may be costly to pass downstream, *e.g.* a self-driving car classifying a bear as a car. OOD detection is again performed using a *binary* detection function,

$$g(\boldsymbol{x};\tau) = \begin{cases} 0 \text{ (OOD)}, & \text{if } U(\boldsymbol{x}) > \tau \\ 1 \text{ (ID)}, & \text{if } U(\boldsymbol{x}) \leq \tau \ . \end{cases} \tag{6}$$

We intuitively want high uncertainty on OOD data. OOD detection may be evaluated using the threshold-free Area Under the Receiver Operating Characteristic curve (AUROC↑), or at a threshold τ determined on ID data *e.g.* false positive rate at 95% true positive rate (FPR@95↓). We assume no access to OOD data before deployment, although other work [25, 61] relaxes this constraint.

Selective classification with OOD Data (SCOD). As OOD detection does not consider the quality of predictions classified as "ID" by the detector (there may be many ID misclassifications), recent work [30, 33, 76] has argued for a task that combines SC with OOD detection. We adopt the problem definition and naming from [76].[3] In SCOD, the objective is to reject *both* ID misclassifications *and* OOD samples, in order to minimise the loss on the accepted predictions. We assume Eq. (5) at deployment and perform binary rejection using Eq. (2). The loss becomes

$$\mathcal{L}_{\text{SCOD}}(f(\boldsymbol{x})) = \begin{cases} 0, & \text{if } f(\boldsymbol{x}) = y, (y,\boldsymbol{x}) \sim p_{\text{ID}} \\ 1, & \text{if } f(\boldsymbol{x}) \neq y, (y,\boldsymbol{x}) \sim p_{\text{ID}} \\ \beta, & \text{if } \boldsymbol{x} \sim p_{\text{OOD}}, \quad (\beta \geq 0) \ , \end{cases} \tag{7}$$

where a cost is incurred if either an ID misclassification or an OOD sample is accepted, and β reflects the relative cost of these two events. The risk is now given by

$$R(f,g;\tau) = \frac{\mathbb{E}_{p_{\text{mix}}(\boldsymbol{x})}[g(\boldsymbol{x};\tau)\mathcal{L}_{\text{SCOD}}(f(\boldsymbol{x}))]}{\mathbb{E}_{p_{\text{mix}}(\boldsymbol{x})}[g(\boldsymbol{x};\tau)]} \ . \tag{8}$$

SCOD can be evaluated similarly to SC with AURC↓. However, as we assume no access to p_{OOD} pre-deployment, we cannot set threshold τ according to risk on p_{mix} (Eq. (8)). We can however set τ according to coverage and then use *e.g.* Risk@(Cov=)80↓ as a threshold-specific measure. This coverage may be only over the ID data (equivalent to TPR for OOD detection), in which case an ID validation set can be used. Alternatively, one may use coverage over all samples. Here statistics of U (not labels) would need to be obtained from p_{mix} during deployment.

2.2. Ensembles and cascades

Deep Ensembles. Training multiple deep neural networks of the same architecture on the same data, but with different random seeds is a well-established approach to improving the quality of uncertainty estimates [37]. With a Deep Ensemble of M networks $\Theta = \{\boldsymbol{\theta}^{(m)}\}_{m=1}^{M}$, the predictive distribution is typically obtained as

$$P(y|\boldsymbol{x};\Theta) = \frac{1}{M}\sum_{m=1}^{M} P(y|\boldsymbol{x};\boldsymbol{\theta}^{(m)}) \ , \tag{9}$$

that is to say the mean of the member softmax outputs. Predictions are then obtained similarly to Eq. (1). To calculate ensemble uncertainty we adopt the simple approaches of directly using the predictive distribution or averaging individual member uncertainties as in [77],

$$U(\boldsymbol{x};\Theta) = U(P(y|\boldsymbol{x};\Theta)) \ , \tag{10}$$

$$U(\boldsymbol{x};\Theta) = \frac{1}{M}\sum_{m=1}^{M} U(\boldsymbol{x};\boldsymbol{\theta}^{(m)}) \ , \tag{11}$$

although there exist more advanced methods [7, 45, 77].

Adaptive inference with early-exit cascades. One approach to improving the efficiency of deep learning approaches is through *adaptive inference*, where computation is allocated conditionally based on the input sample [22, 38]. Early-exit approaches [38] involve terminating computation early depending on the input when producing a prediction. This may be achieved by designing architectures with intermediate predictors [27, 67], or by cascading full networks in series [71, 72]. By placing more powerful, but also more expensive predictors later on, computation is intuitively saved on "easy" samples that exit early. Although there is work on learnt exit policies [5, 60], a simple and widely adopted approach is to exit if intermediate estimates of uncertainty are below a threshold t [27, 35, 67, 72]. If there are multiple exits, then there may be multiple thresholds. Thresholds can be chosen arbitrarily or optimised on an ID validation set subject to an accuracy or computational constraint [35, 65, 71]. Algorithm 1 applies this to Deep Ensembles (for classification) [71].

In their work, Wang et al. [71] show that the approach in Algorithm 1 enables ensembles to achieve a better

Algorithm 1 Cascaded Deep Ensemble Classifier

Require: Ensemble $\{\boldsymbol{\theta}^{(m)}\}_{m=1}^{M}$ Thresholds $\{t^{(m)}\}_{m=1}^{M-1}$
Test Input $\boldsymbol{x}^*$
for $l = 1,2,\ldots,M$ **do**
  Do inference on $\boldsymbol{x}^*$ using lth ensemble member $\boldsymbol{\theta}^{(l)}$
  Cache outputs for lth member
  Calculate $U(\boldsymbol{x}^*;\{\boldsymbol{\theta}^{(m)}\}_{m=1}^{l})$
  ▷ *Eqs. (10) and (11) using model outputs only up to l* ◁
  if $l = M$ or $U < t^{(l)}$ **then**
    ▷ *exit early using l members if confident, or full ens.* ◁
    return $\hat{y} = \text{argmax}_{\omega} P(\omega|\boldsymbol{x}^*;\{\boldsymbol{\theta}^{(m)}\}_{m=1}^{l})$

[3] The research community has yet to converge on a single name for this task, so other than "SCOD" readers may encounter "failure detection" [30], "unknown detection" [33] or "unified open-set recognition" [4].

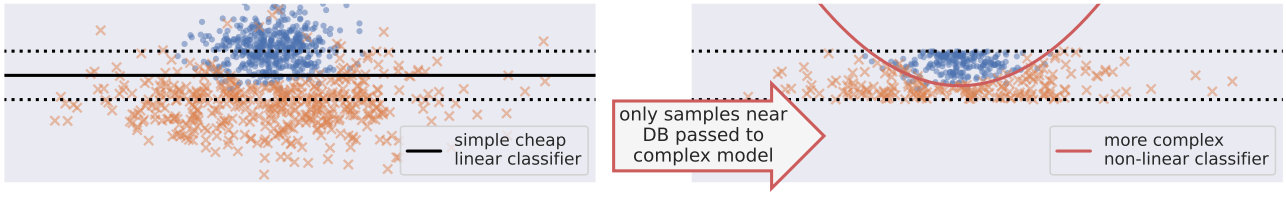

Figure 3. Illustration of intuition for early-exit approaches with a 2D binary classification problem. The most efficient gains in accuracy are obtained when samples near the decision boundary of the first, simple classifier are passed to the second, more powerful (and more costly) classifier. We apply this intuition to the uncertainty-related binary classification tasks in Sec. 2.1 for uncertainty-related early exiting.

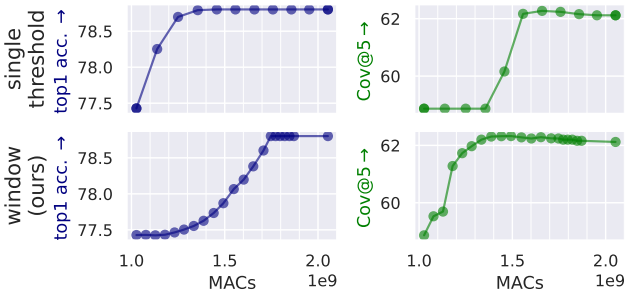

Figure 4. The existing single-threshold policy (top) vs our window-based approach (bottom), as thresholds vary and computation (average MACs) increases. **Although a single threshold is efficient for accuracy, it is not for selective classification** (Cov@5↑). **Our window-based policy shows efficient improvements on the uncertainty-related task**, by targeting the accept/reject decision boundary. Note that when performing SC, top 1 accuracy is not directly relevant, as we are concerned with the risk on only the *accepted* samples, not all samples. We use an EfficientNet-B2 ensemble with $M=2$. The data is ImageNet-1k.

accuracy-computation trade-off compared to scaling single models. We will now extend this to uncertainty estimation.

3. Window-Based Cascades: Targeting Uncertainty-Related Tasks

Directly applying Algorithm 1 results in underwhelming uncertainty performance. Fig. 4 shows that although a rapid improvement in predictive accuracy can be achieved with a small increase in average multiply–accumulate (MAC) operations, uncertainty-related Cov@5↑ only improves after a much larger increase in computation. To better understand why we consider one intuition for early exiting.

3.1. A decision boundary perspective

The typical intuition behind explaining early-exit approaches for multi-class classification is that we can save computation by exiting if an intermediate classifier is confident (low uncertainty) in its prediction. However, it is non-obvious as to how to extend this intuition to uncertainty estimation. Do we need a "meta-uncertainty" to measure the uncertainty of uncertainty estimates? Our key insight is to view early-exiting from a *decision boundary* perspective, and to recall that many downstream tasks using estimates of predictive uncertainty are in fact *binary classification* (see Sec. 2.1, Eqs. (2) and (6)).

Common measures of uncertainty used for early-exiting, such as the maximum softmax probability (MSP), intuitively approximate the distance from the classifier decision boundary [51]. Passing samples from *near the decision boundary* to a more powerful model will result in the most efficient improvement in classification performance. Samples far away are both less likely to be incorrect in the first place and less likely to change sides relative to the decision boundary and be corrected if they are incorrect. This is illustrated in Fig. 3. Applying this intuition to uncertainty-related tasks leads us to our proposed approach.

3.2. Approach: window-based early-exiting

In contrast to existing early-exit approaches that target the decision boundary of the multi-class classification problem using a single threshold on uncertainty, we propose to early-exit only on samples outside of a *window* $[t_1, t_2]$ around the decision boundary τ (Eqs. (2) and (6)) corresponding to the uncertainty-related task of interest, as in Fig. 3. Our approach is detailed in Algorithm 2.

Algorithm 2 Cascaded Deep Ensemble for Uncertainty

Require: Ensemble $\{\boldsymbol{\theta}^{(m)}\}_{m=1}^{M}$ Windows $\{[t_1,t_2]^{(m)}\}_{m=1}^{M-1}$
Test Input $\boldsymbol{x}^*$
for $l=1,2,\ldots,M$ **do**
  Do inference on $\boldsymbol{x}^*$ using lth ensemble member $\boldsymbol{\theta}^{(l)}$
  Cache outputs for lth member
  Calculate $U(\boldsymbol{x}^*;\{\boldsymbol{\theta}^{(m)}\}_{m=1}^{l})$
  ▷ *Eqs. (10) and (11) using model outputs only up to l* ◁
  if $l=M$ or $U \notin [t_1,t_2]^{(l)}$ **then**
    ▷ *exit if U outside window, or full ens.* ◁
    return U, $\hat{y}=\text{argmax}_{\omega} P(\omega|\boldsymbol{x}^*;\{\boldsymbol{\theta}^{(m)}\}_{m=1}^{l})$

In our case, to find $[t_1, t_2]$, we find τ for each exit independently on an ID validation dataset according to the desired criterion, e.g. 80% coverage (Sec. 2.1), and draw a window to capture a certain symmetric percentile of

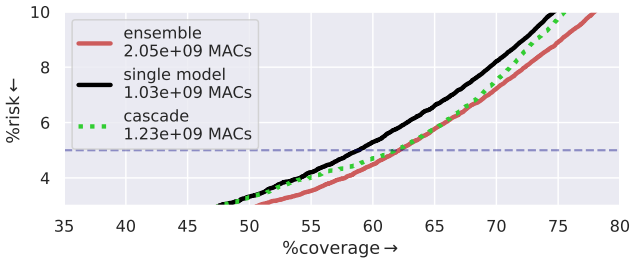

Figure 5. Risk-coverage curves for single model, ensemble, and a window-based cascade. The cascade aims to target risk=5%. **Near the desired operating point the cascade is able to achieve ensemble-like coverage performance**, for a small increase in inference MACs. Further away, at *irrelevant* operating points, the cascade behaves closer to the single model. We use an EfficientNet-B2 ensemble with $M=2$. The data is ImageNet-1k

samples either side of τ. For example, setting $[t_1, t_2]$ at ± 15 percentiles around τ would result in 70% of samples exiting early. Final binary classification for the uncertainty-related task is then performed using a new τ which is found using the final exited uncertainties of the overall adaptive system.

Fig. 4 shows that this window-based approach efficiently improves Cov@5↑. We note that gains in accuracy (over all samples) are worse, as the multi-class decision boundary is no longer targeted. However, during SC, we are concerned with the accuracy of *only the accepted samples*. To explain the SC behaviour of the single threshold in Fig. 4 imagine sweeping a single dashed line over Fig. 3 – the step-like boost occurs when the single t passes over the decision boundary τ meaning samples near τ are now passed on.

The intuition so far has only been linked to the idea that later stages of the cascade have a better binary classifier g, however for SC, it is also the case that they may have a better multi-class classifier f. In this case, after progressing to the next stage, samples may find themselves changing from misclassifications to correct (multi-class) predictions.

Our approach targets a *specific threshold*, which is the *actual deployment setting*. The efficiency gains arise from the fact that *only* samples within a window around τ are passed to the next cascade stage (Fig. 3). Fig. 5 shows that a window-based cascade is able to achieve the full ensemble's performance around the desired 5% risk, however, its behaviour tends to the single model further away (outside the window). As such, threshold-free evaluation metrics (*e.g.* AURC↓) are not suitable for a given window.

We note that it is possible to optimise the window parameters $[t_1, t_2]$ on the ID validation dataset for further improvements, just as regular early-exit thresholds can be optimised [35, 65, 71]. This becomes more relevant in systems with multiple exits. It is desirable to automatically determine $[t_1, t_2]$ when the number of thresholds to find is higher. However, we leave this to future work, as our aim is to simply show the efficacy of our window-based approach.

4. Experimental Results and Discussion

4.1. Main results

Setup. We train two families of computationally efficient image classification CNNs on ImageNet-1k [57]: EfficientNet [64] and MobileNet-V2 [59]. For EfficientNet, we scale from B0→B4. For MobileNet-V2 we use the scalings[4] in Tab. 1. For each model we train a Deep Ensemble size $M = 2$ using random seeds $\{1, 2\}$. We train for 250 epochs, holding out a random subset of 50,000 images for validation, and then evaluate on the original validation set.

Table 1. MobileNet-V2 scaling used in experiments.

input resolution	160	192	224	224	224
width factor	1.0	1.0	1.0	1.3	1.4

All models are trained using Pytorch [50] and Lightning [11]. Full training details can be found in Appendix A. We measure inference computation in multiply-accumulate (MAC) operations[5] using ptflops.[6]

We cascade each ensemble in both directions over seeds ($1 \rightarrow \{1, 2\}$ and $2 \rightarrow \{1, 2\}$) and report the mean values. Note that from this point onwards we refer to cascaded Deep Ensembles as "cascades" and non-adaptive Deep Ensembles as simply "ensembles" for the sake of clarity.

Scaling vs architecture design. We note that the main comparison in this section is between ensembling and *simple scaling*, *e.g.* increasing network width, given a model architecture. Designing a new architecture can also lead to improved computational efficiency. However, the development cost of architecture design, via expert knowledge or neural architecture search [10, 55], makes direct comparison to scaling or ensembling difficult.

4.1.1 Selective classification.

We evaluate SC performance at two different operating points on the risk-coverage curve: Cov@5↑ and Risk@80↓. These represent operating at a much lower risk tolerance vs without selection (∼20-30%) and operating at high coverage. Following Sec. 3.2, we find windows by expanding $[t_1, t_2]$ about τ of the first model in the cascade, capturing symmetric percentiles on either side. We find on the validation set that passing $\pm 10\%$ (20% in total) of ID data around τ to the second stage is sufficient to recover most of the ensemble's performance. We use this window size for the remainder of our experiments. As mentioned previously, we leave further optimisation of $[t_1, t_2]$ to future work.

[4] https://github.com/keras-team/keras/blob/master/keras/applications/mobilenet_v2.py.

[5] Other work may refer to MACs as FLOPs or MADs.

[6] https://github.com/sovrasov/flops-counter.pytorch

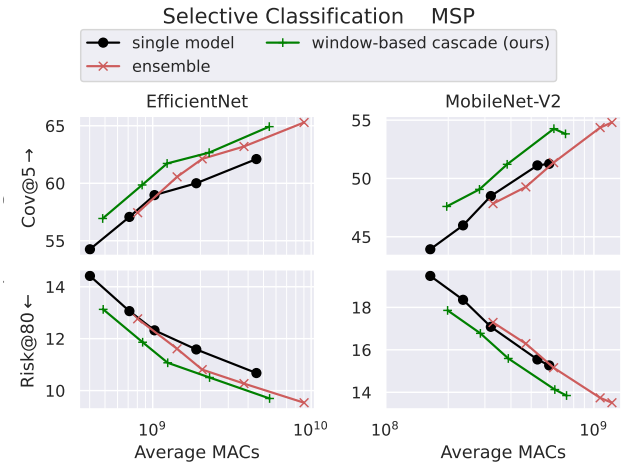

Figure 6. Selective classification performance against average computation (log-scale MACs) for EfficientNet B0→B4 and scaled MobileNet-V2. Ensembles are quite competitive compared to single-model scaling, with an uncertainty-computation trade-off that is slightly worse for lower levels of compute and superior for higher computation. **Our window-based cascades are able to achieve better uncertainty-computation trade-offs in all cases compared to scaling single models**. The data is ImageNet-1k.

We use the standard (negative) Maximum Softmax Probability (MSP) [16, 24] score, $U(\boldsymbol{x}) = -\max_{\omega} P(\omega|\boldsymbol{x})$, to measure uncertainty for SC. For the ensemble, we use Eq. (10) to calculate U. Note that this leads to better misclassification rejection compared to Eq. (11), as we are using Eq. (9) for classification with the ensemble.

Cascades are best and ensembles are competitive. Fig. 6 shows that ensembles are slightly weaker than scaling single models in the lower computation region, and can outperform them at higher compute costs. This is in line with previous work on accuracy [34, 43, 71]. Moreover, it is clear, that by cascading ensembles using our window-based early-exit policy, a superior uncertainty-computation trade-off can be achieved, similar to accuracy in [71].

4.1.2 OOD detection.

We use two high-resolution OOD datasets designed for ImageNet-scale experiments. Openimage-O [70] and iNaturalist [29] are subsets of larger datasets [36, 69] where images were selected to be disjoint to ImageNet-1k labels. The former covers a wide range of classes (like ImageNet); the latter contains botanical images. Additional dataset details can be found in Appendix E. We evaluate at two operating points, FPR@80↓ and FPR@95↓, which represent high and very high retention of ID data. Initially, we set $[t_1, t_2]$ in the same fashion as for SC (Sec. 4.1.1). To measure average computation, we evaluate on a simple scenario

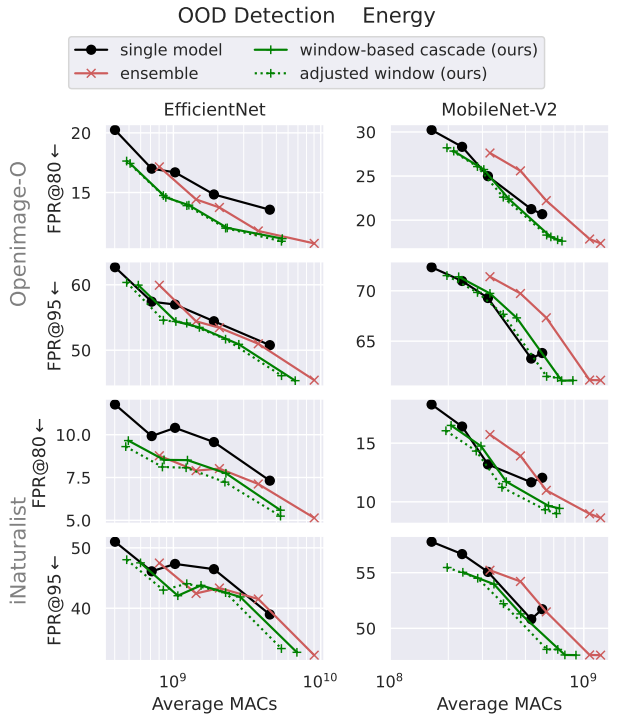

Figure 7. OOD detection performance vs computation. Window-based cascades are competitive overall but suffer from slowdown due to distributional shift in some cases. **Adjusting the window at deployment recovers cascade efficiency**, allowing for more efficient uncertainties compared to scaling single models. **Scaling single models does not reliably reduce FPR↓, but ensembling does**, suggesting that ensembles/cascades are the *safer* option for improving OOD detection. We assume a 1:1 ratio of ID:OOD data.

where $\alpha = 0.5$ (Eq. (5)), i.e. the number of OOD samples is equal to ID samples. The OOD datasets are smaller, so we randomly subsample the ImageNet evaluation set.

We use the simple and popular Energy [42] score, $U(\boldsymbol{x}) = -\log \sum_k \exp v_k$, where $\boldsymbol{v}$ are the pre-softmax logits. For the ensemble, we use Eq. (11) to calculate U as in [77].

Beware of distribution shift when early exiting. Fig. 7 shows that in most scenarios, window-based cascades are better or competitive with single models. However, cascades are weaker in some cases. We hypothesise that this is due to there being more OOD data between $[t_1, t_2]$ compared to ID data, slowing the cascade down, as in this case τ is on the lower tail of the ID data. We remark that slowdown under distributional shift is a general problem with early-exit approaches, and is an interesting direction for further research. We also note that there will be less slowdown in scenarios where OOD data is rarer than ID data ($\alpha > 0.5$).

To correct for this slowdown, we propose *adjusting* the window such that $[t_1, t_2]$ capture $\pm 10\%$ around τ of p_{mix} rather than of p_{ID} (Eq. (5)). To achieve this, percentiles

need to be calculated over the Us of the *test* data. As this *does not require labels*, it could be practically achieved by collecting statistics during deployment and adjusting $[t_1, t_2]$ accordingly. For online processing this may be done in a running manner, whereas for offline data processing, all samples could be processed by the first cascade stage, percentiles could be then calculated, and the corresponding subset of samples passed on to the next stage. Fig. 7 shows that adjusted windows recover cascade efficiency, allowing them to outperform or match single models in all cases. Larger speedups are observed for FPR@95↓, as this τ is on the tail of the ID distribution of U and is likely to be closer to the mode of the OOD distribution.

Scaling doesn't reliably improve OOD detection but ensembling does. Fig. 7 shows that although model scaling has a general trend of reducing FPR↓, it is inconsistent between different models, OOD datasets, and operating thresholds. In fact, *scaling up does not always improve OOD detection*, *e.g.* EfficientNet on iNaturalist. On the other hand, cascades/ensembles *reliably* reduce FPR↓ relative to their corresponding single members. This suggests that cascades are the safer choice for efficiently boosting OOD detection performance, given that model scaling may not even improve detection at all.

4.1.3 SCOD.

We keep the same OOD datasets as the previous section, as well as the 1:1 ratio of ID:OOD data. We set $\beta = 1$ (Eq. (7)), i.e. accepting OOD data and misclassifications is equally costly. We evaluate Risk@80↓, and Risk@50↓, where the coverage is over the ID data (see Sec. 2.1). The first offers higher coverage, whilst the second results in much lower risks. We evaluate with windows set as in Sec. 4.1.1, as well as the adjusted windows from the previous section. We use (negative) MSP for U just as in Sec. 4.1.1.

Overall results (in Fig. 8) are similar to Sec. 4.1.1, with the cascades outperforming scaling single models for all levels of computation. Ensembles perform even better than for in-distribution SC, producing consistently better uncertainty-computation trade-offs vs scaling EfficientNet. This suggests that ensembling leads to even greater gains in being able to distinguish OOD data from *correct* ID predictions, compared to rejecting ID misclassifications.

Some of the OOD detection behaviour from the previous section carries over to SCOD, with ensembling being a *more reliable* way to reduce risk compared to model scaling (see EfficientNet on iNaturalist). Using adjusted windows leads to little change in Risk@50↓, as a stricter τ will have fewer OOD samples near it to cause slowdown. Improvements in Risk@80↓, where τ is closer to the OOD distribution of U, can be observed in some cases, however.

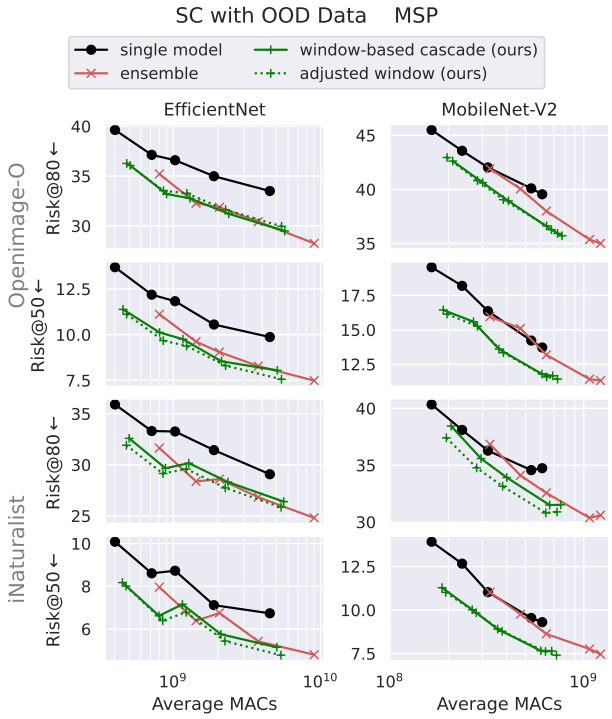

Figure 8. SCOD performance vs computation. Results are in-line with previous Figs. 6 and 7. **Window-based cascades achieve more efficient uncertainty-related performance compared to scaling single models**, and sometimes benefit from deployment-time window adjustment. Ensembles perform better than before, being more efficient than single-model scaling for EfficientNet.

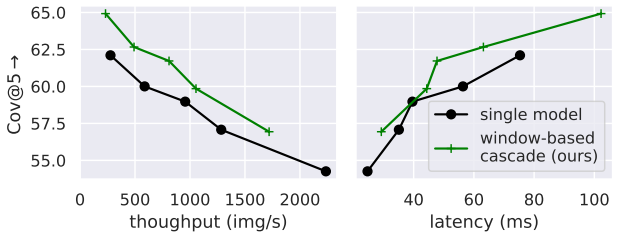

Figure 9. Real-world latency and throughput uncertainty efficiency gains for window-based cascades for EfficientNet B0→B4.

4.2. Real-world latency and throughput

We validate our window-based cascades by measuring average real-world latency (batch size 1) and throughput (batch size 256) for EfficientNets. We use Pytorch and measure latency on an Intel Xeon E5-2698 v4 CPU and throughput on a NVIDIA V100 32GB GPU. For latency, the exit condition is evaluated for each sample, whilst for throughput, it is evaluated over the whole batch, and then the corresponding subset of the batch is passed to the second model. Fig. 9 shows that the theoretical MACs results in Sec. 4.1.1 can indeed translate to real-world deployment.

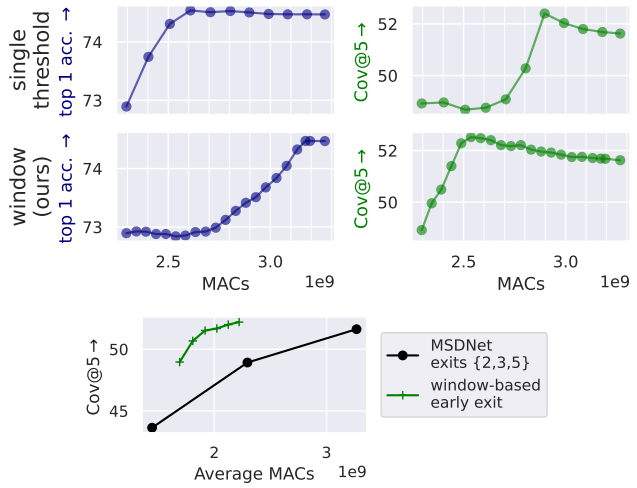

Figure 10. Top: comparison of single-threshold vs window-based early-exit policies on MSDNet exits 3→5. **Our approach generalises to uncertainty estimation on other early-exit architectures.** Bottom: window-based policy applied to 3 exits of MSDNet for SC. **Our approach generalises to multiple early exits.**

4.3. Other early-exit architectures

Although the focus of this work has been on adaptive inference with cascaded Deep Ensembles, the proposed window-based early-exit policy can be applied to *any* early-exit network. To show this, we apply a similar analysis to Fig. 4 between the 3rd and 5th exits of a MSDNet (step=7) [27] pretrained on ImageNet-1k.[7] Fig. 10 shows that our window-based early-exit policy does indeed generalise to other early-exit architectures. Note that MSDNet exits are not ensembled [27]. However, recall that SC can improve with either g or f (Sec. 2.1), so we still expect later exits to be better as they are more accurate (better f).

To show that our approach generalises beyond a single exit, we also apply it to exits $\{2,3,5\}$ of MSDNet. We fix $[t_1,t_2]^{(2)}$ at ± 10 percentiles around $\tau^{(2)}$ and vary the width of the first window $[t_1,t_2]^{(1)}$. Fig. 10 shows that we achieve an efficient uncertainty trade-off vs the individual exits.

5. Related Work

Uncertainty estimation in deep learning. There is a large body of work on improving predictive uncertainty estimates for deep learning approaches [1]. Some focus on improving performance on a specific binary detection task, such as SC [9, 16], OOD detection [24, 78] and now most recently, SCOD [33, 76]. Approaches include: alternative uncertainty scores [39, 42, 47, 70], separate error detectors trained post-hoc [31, 74], modified training approaches [8, 26, 29, 66], feature clipping or sparsification [62, 63, 82] and confidence score combination [4, 28, 70, 76]. Calibration [20] is a separate task where the aim is to produce probabilities that match the true accuracy of a classifier. It is orthogonal to the aforementioned tasks (and the focus of this paper), as in binary classification only the *rank ordering* of scores matters, rather than absolute probabilities [30].

Other work aims to generally improve uncertainties over multiple evaluation metrics. Deep ensembles [37] have stood out as a strong and reliable baseline in this field. However, due to their increased cost, there have been many proposed *single-inference* approaches with the aim of improving uncertainty estimates at a lower cost. Examples include, low-rank weight matrices and weight sharing (BatchEnsemble) [73], implicit subnetworks (MIMO) [23], using spectral normalisation combined with a gaussian process (SNGP) [41] or a gaussian mixture model (DDU) [48], distribution distillation of the ensemble into a single model (EnD^2) [12, 58] and more [15, 19, 52, 53, 68]. We note that our work is orthogonal to these, as any single-inference approach can be ensembled and then cascaded using our window-based adaptive inference approach.

Early-exit networks. Within the wide range of approaches for dynamic or adaptive inference [22], where computation allocation is not fixed during deployment, early-exiting is a simple and popular choice [38, 67]. The core idea is to terminate computation early on "easier" samples in order to make more efficient use of computational resources. This can be performed by cascading entire networks of increasing predictive performance [71, 72], or by attaching intermediate predictors to a backbone [35, 80] or by designing a specialised architecture [27, 79]. Exit policies can be rules-based or learnt [38]. Although the vast majority of work in this field focuses on predictive accuracy, MOOD [40] proposes an approach targeting OOD detection. Unlike our window-based policy, their policy relies on a heuristic based on input image PNG [56] compression ratio to determine the exit and is restricted to only the task of OOD detection. Qendro et al. [54] utilise a multi-exit architecture to improve uncertainty-related performance. However, they ensemble all exits without any adaptive inference.

6. Concluding Remarks

In this work, we investigate the uncertainty-computation trade-off of Deep Ensembles vs scaling single models. Additionally, we propose a window-based early-exit policy and use it to efficiently cascade ensembles for uncertainty-related tasks such as selective classification, OOD detection, and SCOD. Evaluations on ImageNet-scale CNNs and data show that our window-based cascades achieve a superior uncertainty-computation trade-off compared to simply scaling up single models within an architecture family. We hope our work encourages more research into adaptive inference approaches for uncertainty estimation.

[7] https://github.com/kalviny/MSDNet-PyTorch

References

[1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U. Rajendra Acharya, Vladimir Makarenkov, and Saeid Nahavandi. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021. 8

[2] Andrew Beam and Ben Kompa. Second opinion needed: Communicating uncertainty in medical artificial intelligence. *NPJ Digital Medicine*, 4, 2021. 1

[3] William H. Beluch, Tim Genewein, A. Nürnberger, and Jan M. Köhler. The power of ensembles for active learning in image classification. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9368–9377, 2018. 1

[4] Jun CEN, Di Luan, Shiwei Zhang, Yixuan Pei, Yingya Zhang, Deli Zhao, Shaojie Shen, and Qifeng Chen. The devil is in the wrongly-classified samples: Towards unified open-set recognition. In *The Eleventh International Conference on Learning Representations*, 2023. 3, 8

[5] Xinshi Chen, Hanjun Dai, Yu Li, Xin Gao, and Le Song. Learning to stop while learning to predict. In *International Conference on Machine Learning*, pages 1520–1530. PMLR, 2020. 3

[6] Didan Deng and Emil Bertram Shi. Ensembling with a fixed parameter budget: When does it help and why? In Vineeth N. Balasubramanian and Ivor Tsang, editors, *Proceedings of The 13th Asian Conference on Machine Learning*, volume 157 of *Proceedings of Machine Learning Research*, pages 1176–1191. PMLR, 17–19 Nov 2021. 1

[7] Stefan Depeweg, José Miguel Hernández-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *ICML*, 2018. 3

[8] Xuefeng Du, Zhaoning Wang, Mu Cai, and Yixuan Li. Vos: Learning what you don't know by virtual outlier synthesis. *Proceedings of the International Conference on Learning Representations*, 2022. 8

[9] Ran El-Yaniv and Yair Wiener. On the foundations of noise-free selective classification. *J. Mach. Learn. Res.*, 11:1605–1641, 2010. 2, 8

[10] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019. 5

[11] William Falcon and The PyTorch Lightning team. PyTorch Lightning, 3 2019. 5, 13

[12] Yassir Fathullah and Mark J. F. Gales. Self-distribution distillation: efficient uncertainty estimation. In James Cussens and Kun Zhang, editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 663–673. PMLR, 01–05 Aug 2022. 8

[13] Yassir Fathullah, Mark John Francis Gales, and Andrey Malinin. Ensemble distillation approaches for grammatical error correction. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2745–2749, 2020. 1

[14] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *ArXiv*, abs/1912.02757, 2019.

[15] Gianni Franchi, Xuanlong Yu, Andrei Bursuc, Emanuel Aldea, Severine Dubuisson, and David Filliat. Latent discriminant deterministic uncertainty. In *Computer Vision – ECCV 2022*, Cham, 2022. 1, 8

[16] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *NIPS*, 2017. 2, 6, 8

[17] Yonatan Geifman, Guy Uziel, and Ran El-Yaniv. Bias-reduced uncertainty estimation for deep neural classifiers. In *International Conference on Learning Representations*, 2019. 1

[18] Priya Goyal, Piotr Dollár, Ross B. Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *ArXiv*, abs/1706.02677, 2017. 13

[19] Federica Granese, Marco Romanelli, Daniele Gorla, Catuscia Palamidessi, and Pablo Piantanida. Doctor: A simple method for detecting misclassification errors. In *NeurIPS*, 2021. 8

[20] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 06–11 Aug 2017. 8

[21] Fredrik K Gustafsson, Martin Danelljan, and Thomas B Schön. Evaluating scalable bayesian deep learning methods for robust computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2020. 1

[22] Y. Han, G. Huang, S. Song, L. Yang, H. Wang, and Y. Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7436–7456, nov 2022. 3, 8

[23] Marton Havasi, Rodolphe Jenatton, Stanislav Fort, Jeremiah Zhe Liu, Jasper Snoek, Balaji Lakshminarayanan, Andrew Mingbo Dai, and Dustin Tran. Training independent subnetworks for robust prediction. In *International Conference on Learning Representations*, 2021. 1, 8

[24] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *Proceedings of International Conference on Learning Representations*, 2017. 6, 8

[25] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *Proceedings of the International Conference on Learning Representations*, 2019. 3

[26] Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira. Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10948–10957, 2020. 8

[27] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q. Weinberger. Multi-scale dense networks for resource efficient image classification. In *ICLR*, 2018. 2, 3, 8, 13

[28] Rui Huang, Andrew Geng, and Yixuan Li. On the importance of gradients for detecting distributional shifts in the wild. In *NeurIPS*, 2021. 8

[29] Rui Huang and Yixuan Li. Mos: Towards scaling out-of-distribution detection for large semantic space. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8706–8715, 2021. 6, 8, 14

[30] Paul F Jaeger, Carsten Tim Lüth, Lukas Klein, and Till J. Bungert. A call to reflect on evaluation practices for failure detection in image classification. In *International Conference on Learning Representations*, 2023. 2, 3, 8

[31] Amita Kamath, Robin Jia, and Percy Liang. Selective question answering under domain shift. In *ACL*, 2020. 8

[32] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 1

[33] Jihyo Kim, Jiin Koo, and Sangheum Hwang. A unified benchmark for the unknown detection capability of deep neural networks. *ArXiv*, abs/2112.00337, 2021. 1, 2, 3, 8

[34] D. Kondratyuk, Mingxing Tan, Matthew A. Brown, and Boqing Gong. When ensembling smaller models is more efficient than single large models. *ArXiv*, abs/2005.00570, 2020. 1, 6

[35] Alexandros Kouris, Stylianos I. Venieris, Stefanos Laskaridis, and Nicholas Lane. Multi-exit semantic segmentation networks. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 330–349, Cham, 2022. Springer Nature Switzerland. 3, 5, 8

[36] Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*, 2017. 6, 14

[37] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2020. 1, 3, 8

[38] Stefanos Laskaridis, Alexandros Kouris, and Nicholas D. Lane. Adaptive inference through early-exit networks: Design, challenges and directions. In *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning*, EMDL'21, page 1–6, New York, NY, USA, 2021. Association for Computing Machinery. 1, 3, 8

[39] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018. 8

[40] Ziqian Lin, Sreya Dutta Roy, and Yixuan Li. Mood: Multi-level out-of-distribution detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15313–15323, June 2021. 8

[41] Jeremiah Zhe Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax-Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. 8

[42] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 2020. 6, 8

[43] Ekaterina Lobacheva, Nadezhda Chirkova, Maxim Kodryan, and Dmitry P Vetrov. On power laws in deep ensembles. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 2375–2385. Curran Associates, Inc., 2020. 1, 6

[44] Andrey Malinin, Neil Band, Yarin Gal, Mark Gales, Alexander Ganshin, German Chesnokov, Alexey Noskov, Andrey Ploskonosov, Liudmila Prokhorenkova, Ivan Provilkov, Vatsal Raina, Vyas Raina, Denis Roginskiy, Mariya Shmatova, Panagiotis Tigas, and Boris Yangel. Shifts: A dataset of real distributional shift across multiple large-scale tasks. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 1

[45] Andrey Malinin and Mark John Francis Gales. Uncertainty estimation in autoregressive structured prediction. In *ICLR*, 2021. 3

[46] Andrey Malinin, Bruno Mlodozeniec, and Mark John Francis Gales. Ensemble distribution distillation. *Proceedings of the International Conference on Learning Representations*, 2020. 1

[47] Yifei Ming, Ying Fan, and Yixuan Li. POEM: Out-of-distribution detection with posterior sampling. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15650–15665. PMLR, 17–23 Jul 2022. 8

[48] Jishnu Mukhoti, Andreas Kirsch, Joost R. van Amersfoort, Philip H. S. Torr, and Yarin Gal. Deterministic neural networks with appropriate inductive biases capture epistemic and aleatoric uncertainty. *ArXiv*, abs/2102.11582, 2021. 8

[49] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019. 1

[50] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 5, 13

[51] Tim Pearce, Alexandra Brintrup, and Jun Zhu. Understanding softmax confidence and uncertainty. *ArXiv*, abs/2106.04972, 2021. 4

[52] Francesco Pinto, Harry Yang, Ser-Nam Lim, Philip Torr, and Puneet K. Dokania. Mix-maxent: Improving accuracy and uncertainty estimates of deterministic neural networks. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021. 8

[53] Francesco Pinto, Harry Yang, Ser-Nam Lim, Philip Torr, and Puneet K. Dokania. Using mixup as a regularizer can surprisingly improve accuracy & out-of-distribution robustness. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. 8

[54] Lorena Qendro, Alexander Campbell, Pietro Lio, and Cecilia Mascolo. Early exit ensembles for uncertainty quantification. In Subhrajit Roy, Stephen Pfohl, Emma Rocheteau, Girmaw Abebe Tadesse, Luis Oala, Fabian Falck, Yuyin Zhou, Liyue Shen, Ghada Zamzmi, Purity Mugambi, Ayah Zirikly, Matthew B. A. McDermott, and Emily Alsentzer, editors, *Proceedings of Machine Learning for Health*, volume 158 of *Proceedings of Machine Learning Research*, pages 181–195. PMLR, 04 Dec 2021. 8

[55] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A comprehensive survey of neural architecture search. *ACM Computing Surveys (CSUR)*, 54:1 – 34, 2021. 5

[56] Greg Roelofs and Richard Koman. *PNG: The Definitive Guide*. O'Reilly and Associates, Inc., USA, 1999. 8

[57] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015. 5, 13, 14

[58] Max Ryabinin, Andrey Malinin, and Mark Gales. Scaling ensemble distribution distillation to many classes with proxy targets. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 6023–6035. Curran Associates, Inc., 2021. 8

[59] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 5, 13

[60] Simone Scardapane, Danilo Comminiello, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. Differentiable branching in deep networks for fast inference. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4167–4171, 2020. 3

[61] Vikash Sehwag, Mung Chiang, and Prateek Mittal. {SSD}: A unified framework for self-supervised outlier detection. In *International Conference on Learning Representations*, 2021. 3

[62] Yiyou Sun, Chuan Guo, and Yixuan Li. React: Out-of-distribution detection with rectified activations. In *NeurIPS*, 2021. 8

[63] Yiyou Sun and Yixuan Li. Dice: Leveraging sparsification for out-of-distribution detection. In *European Conference on Computer Vision*, 2022. 8

[64] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019. 5, 13

[65] Xinrui Tan, Hongjia Li, Liming Wang, Xueqing Huang, and Zhen Xu. Empowering adaptive early-exit inference with latency awareness. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(11):9825–9833, May 2021. 3, 5

[66] Engkarat Techapanurak, Masanori Suganuma, and Takayuki Okatani. Hyperparameter-free out-of-distribution detection using cosine similarity. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020. 8

[67] Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. Branchynet: Fast inference via early exiting from deep neural networks. *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2464–2469, 2016. 1, 3, 8

[68] Joost van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. 2020. 8

[69] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset, 2017. 6, 14

[70] Haoqi Wang, Zhizhong Li, Litong Feng, and Wayne Zhang. Vim: Out-of-distribution with virtual-logit matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 6, 8, 14

[71] Xiaofang Wang, Dan Kondratyuk, Eric Christiansen, Kris M. Kitani, Yair Movshovitz-Attias, and Elad Eban. Wisdom of committees: An overlooked approach to faster and more accurate models. In *International Conference on Learning Representations*, 2022. 1, 3, 5, 6, 8, 14

[72] Xin Wang, Yujia Luo, Daniel Crankshaw, Alexey Tumanov, and Joseph Gonzalez. Idk cascades: Fast deep learning by learning not to overthink. 2018. 1, 3, 8

[73] Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2020. 8

[74] Spencer Whitehead, Suzanne Petryk, Vedaad Shakib, Joseph Gonzalez, Trevor Darrell, Anna Rohrbach, and Marcus Rohrbach. Reliable visual question answering: Abstain rather than answer incorrectly. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 148–166, Cham, 2022. Springer Nature Switzerland. 2, 8

[75] Xixin Wu, Kate Knill, Mark John Francis Gales, and Andrey Malinin. Ensemble approaches for uncertainty in spoken language assessment. In *Interspeech*, 2020. 1

[76] Guoxuan Xia and Christos-Savvas Bouganis. Augmenting softmax information for selective classification with out-of-distribution data. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 1995–2012, December 2022. 2, 3, 8

[77] Guoxuan Xia and Christos-Savvas Bouganis. On the usefulness of deep ensemble diversity for out-of-distribution detection. *ArXiv*, abs/2207.07517, 2022. 1, 3, 6

[78] Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021. 2, 8

[79] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution adaptive networks for efficient inference. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2366–2375, 2020. 8

[80] Yigitcan Kaya and Sanghyun Hong and Tudor Dumitras. Shallow-Deep Networks: Understanding and mitigating network overthinking. In *Proceedings of the 2019 International Conference on Machine Learning (ICML)*, Long Beach, CA, Jun 2019. 8

[81] Shuai Zhao, Liguang Zhou, Wenxiao Wang, D. Cai, Tin Lun Lam, and Yangsheng Xu. Toward better accuracy-efficiency trade-offs: Divide and co-training. *IEEE Transactions on Image Processing*, 31:5869–5880, 2020. 1

[82] Yao Zhu, YueFeng Chen, Chuanlong Xie, Xiaodan Li, Rong Zhang, Hui Xue', Xiang Tian, bolun zheng, and Yaowu Chen. Boosting out-of-distribution detection with typical features. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. 8

A. Additional Setup Details

Training. We train two families of computationally efficient CNNs for image classification on ImageNet-1k [57]: EfficientNet [64] and MobileNet-V2 [59]. For EfficientNet, we scale width, depth and resolution as the original authors [64] from B0→B4. For MobileNet-V2 we use the scalings in Tab. 2, which are taken from the Keras github repository.[8] For each model we train a Deep Ensemble size $M = 2$ using random seeds $\{1,2\}$ (everything is the same between ensemble members other than the random seed).

Table 2. MobileNet-V2 scaling used in experiments.

input resolution	160	192	224	224	224
width factor	1.0	1.0	1.0	1.3	1.4

We train all models for 250 epochs on ImageNet-1k, and hold out a random subset of 50,000 images from the training set for validation (we then evaluate on the original validation set). We train using standard cross entropy. We use stochastic gradient descent with a learning rate of 0.2, weight decay of 4e-5 and a batch size of 1024 for all models other than EfficientNet-B4, for which we use a learning rate of 0.1 and batch size of 512 due to GPU memory constraints (following the scaling recommendations in [18]). We use cosine learning rate decay with a 5 epoch linear warmup.[9] We use default random resize-cropping and random horizontal flipping for data augmentation, and images are scaled using bicubic interpolation.

We train all of our models using PyTorch [50] and Lightning [11] distributed over 8 NVIDIA V100 32GB GPUs using Automatic Mixed Precision.[10] Training and evaluation code can be found at `https://github.com/Guoxoug/window-early-exit`. Please follow the instructions in the `README.md` file in order to reproduce our results and plots.

Setting windows. In general, we find τ on the validation set and then vary $[t_1,t_2]$ by placing them at increasing symmetric percentiles on either side of τ. If either side of the window hits either the zeroth or 100th percentile, then the expansion will only apply to the other side, *e.g.* if τ is set at TPR=95% then there is only room for 5% (of ID data) on the side more uncertain than τ. As mentioned previously, we leave further optimisation of $[t_1,t_2]$ to future work.

[8] `https://github.com/keras-team/keras/blob/master/keras/applications/mobilenet_v2.py`.

[9] This is a combination of the hyperparameters from `https://github.com/d-li14/mobilenetv2.pytorch` and the scaling approaches recommended in [18].

[10] `https://developer.nvidia.com/automatic-mixed-precision`

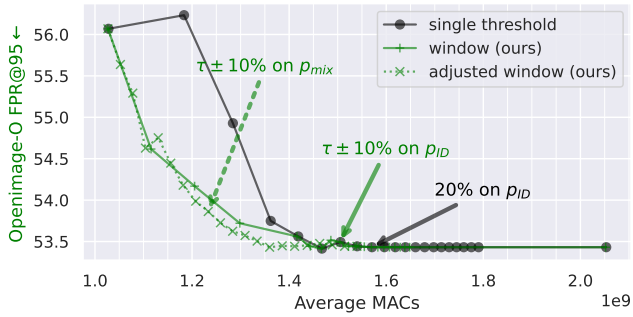

Figure 11. Comparison of OOD detection efficiency for different exit policies. The ratio of ID:OOD data is 1:1 ($\alpha = 0.5$). Using a single threshold is inefficient compared to using a window. Adjusting $[t_1, t_2]$ using statistics from p_{mix} significantly reduces the slowdown caused by distributional shift (if the window is set on p_{ID}). We use an EfficientNet-B2 ensemble with $M = 2$.

MSDNet. For MSDNet [27] we use publically available pretrained weights[11]. We use the version of the model designed for ImageNet with step=7, and experiment on a subset $\{2,3,5\}$ of the five available exits. For the three-exit scenario we fix the percentiles of the second exit $[t_1,t_2]^{(2)}$ at ± 10 around $\tau^{(2)}$ and increase the percentiles of the first exit $[t_1,t_2]^{(1)}$ from $\pm 10 \rightarrow \pm 35$ in steps of 5% around $\tau^{(1)}$.

B. Additional Exit Policy Comparisons

We show comparisons between the single-threshold policy, and our window-based policy for OOD detection (FPR@95↓, Openimage-O, $\alpha = 0.5$). For the single-threshold and non-adjusted window approach we set t and $[t_1, t_2]$ based on the percentage of p_{ID} passed on to the next cascade stage. For the *adjusted* window we set $[t_1,t_2]$ according to percentiles measured on p_{mix} instead.

Similarly to SC, Fig. 11 shows that the single-threshold approach only improves after t (the exit threshold) passes over τ (the detection threshold). However, this happens earlier as the operating point TPR=95% happens to be closer to the starting point of the single-threshold sweep (it starts from *most* uncertain). Our window-based approach more efficiently improves OOD detection.

Different specific exit policies are also marked. It can be seen that setting the window $[t_1, t_2]$ according to p_{mix} rather than p_{ID} significantly reduces slowdown caused by distribution shift. Setting $[t_1,t_2]$ to $\pm 10\%$ around τ on p_{ID} leads to $\sim 50\%$ of samples from p_{mix} passing through to the second stage. Note that setting $[t_1,t_2]$ at ± 10 percentiles around TPR=95% on ID data only allows 15% of ID data through as the window caps out on one side.

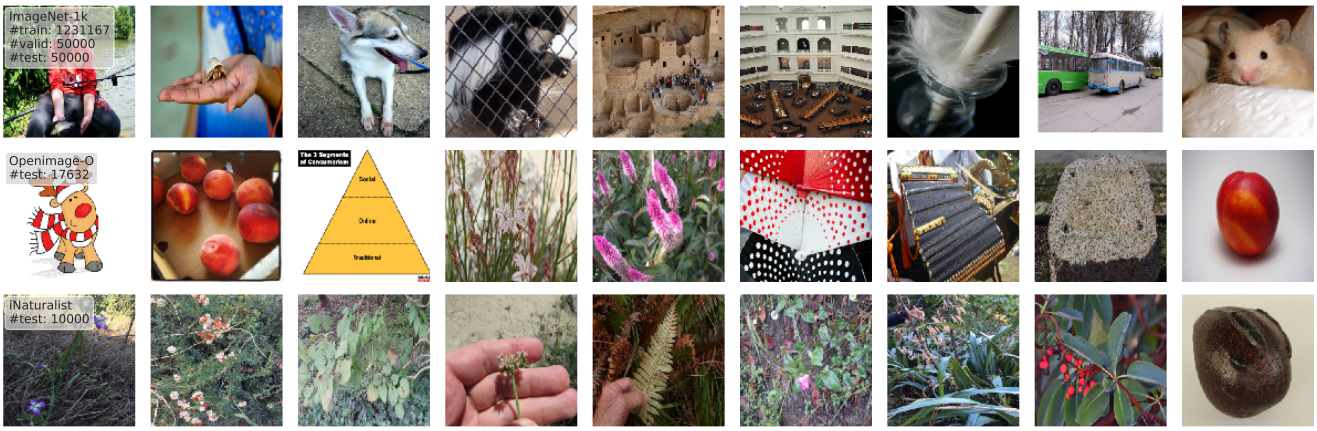

Figure 12. Example images from each image dataset used, with #samples in each split.

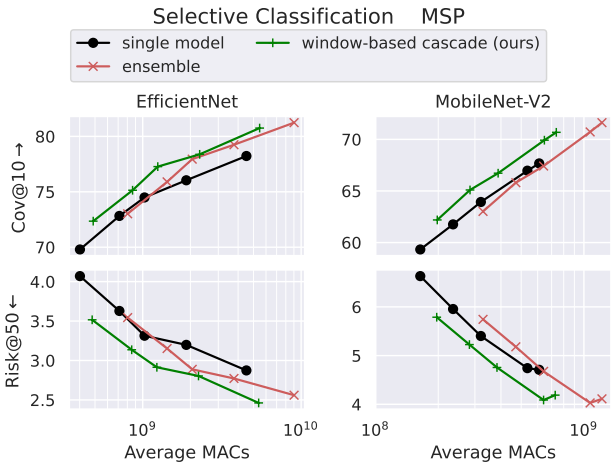

Figure 13. SC–computation comparison for single models, ensembles and window-based cascades, on additional operating thresholds. Results tell the same story as the main paper – cascades are able to achieve the best uncertainty-computation trade-off.

C. Additional Selective Classification Results

We include additional SC results at two more operating points, Risk@50↓ and Cov@10↑, which represent, compared to the main results, a lower coverage requirement and a higher risk tolerance respectively. The results are similar to those in the main paper, showing that cascades are able to achieve efficient uncertainty estimation compared to model scaling over a range of different operating thresholds.

D. Accuracy-Computation Results

Fig. 14 shows the accuracy-computation trade-off using single-threshold cascades for EfficientNet and MobileNet-V2. We pass the most uncertain 20% of samples from the 1st model to the second cascade stage. The results are unsurprisingly similar to those in [71], i.e. ensembles are less efficient than single models in the low-compute region, but outperform them for higher computation levels. Cascades then allow ensembles to become more efficient for all levels of compute. We note the baseline accuracy of our EfficientNets is lower than in [71] as we train them for fewer epochs with a simpler recipe and larger validation set, however, we do not believe this affects the takeaways from our results.

[11] https://github.com/kalviny/MSDNet-PyTorch

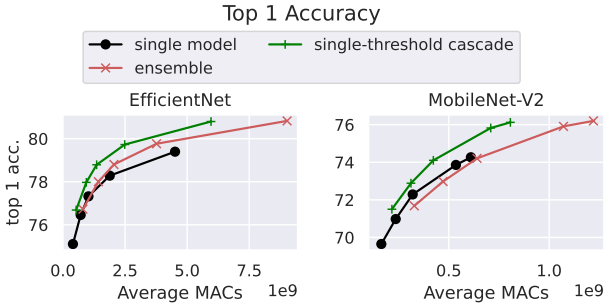

Figure 14. Accuracy–computation comparison for single models, ensembles and single-threshold cascades.

E. Additional Dataset Information

We include randomly sampled example images from the datasets used in this work: ImageNet-1k [57], Openimage-O [36, 70], and iNaturalist [29, 69]. We also show information about the number of samples in each dataset split (Fig. 12). The OOD datasets are recently released high-resolution benchmarking datasets. They aim to move vision-based OOD detection evaluation beyond CIFAR-scale images into more *realistic* image-classification scenarios. The samples in each dataset are carefully chosen to be semantically disjoint from the label space of ImageNet-1k. Openimage-O contains a wide range of classes like ImageNet, whilst iNaturalist contains botanical images.