Fixing Overconfidence in Dynamic Neural Networks

Lassi Meronen[1,2] Martin Trapp[1] Andrea Pilzer[3] Le Yang[4] Arno Solin[1]
[1]Aalto University [2]Saab Finland Oy [3]NVIDIA [4]Xi'an Jiaotong University
{lassi.meronen, martin.trapp, arno.solin}@aalto.fi, apilzer@nvidia.com, yangle15@xjtu.edu.cn

Abstract

Dynamic neural networks are a recent technique that promises a remedy for the increasing size of modern deep learning models by dynamically adapting their computational cost to the difficulty of the inputs. In this way, the model can adjust to a limited computational budget. However, the poor quality of uncertainty estimates in deep learning models makes it difficult to distinguish between hard and easy samples. To address this challenge, we present a computationally efficient approach for post-hoc uncertainty quantification in dynamic neural networks. We show that adequately quantifying and accounting for both aleatoric and epistemic uncertainty through a probabilistic treatment of the last layers improves the predictive performance and aids decision-making when determining the computational budget. In the experiments, we show improvements on CIFAR-100, ImageNet, and Caltech-256 in terms of accuracy, capturing uncertainty, and calibration error.

1. Introduction

The ability to scale deep neural networks up to millions of parameters (*e.g.*, [29, 44, 50]) on massive data sets (*e.g.*, [8, 28]) has been a crucial part of achieving impressive performance, sometimes exceeding human experts on many natural-language processing and computer vision tasks. However, learning and deploying such models entails high computational costs and becomes increasingly difficult [44, 50], especially as inference costs arise for every deployed instance and can heavily add up [43].

Henceforth, there has been an increased interest in techniques for energy-efficient inference in deep learning [2, 13]. A particularly promising direction is to *dynamically* select the most cost-efficient sub-model based on the *difficulty* of the test sample. Dynamic neural networks (DNNs, [15, 18]) leverage a multi-exit architecture and *dynamically* route test samples based on their difficulty. For example, an image of a sunflower might be easy to classify, requiring less compute, while a tiger in a snowy environment might be more ambiguous and can only be correctly identified after

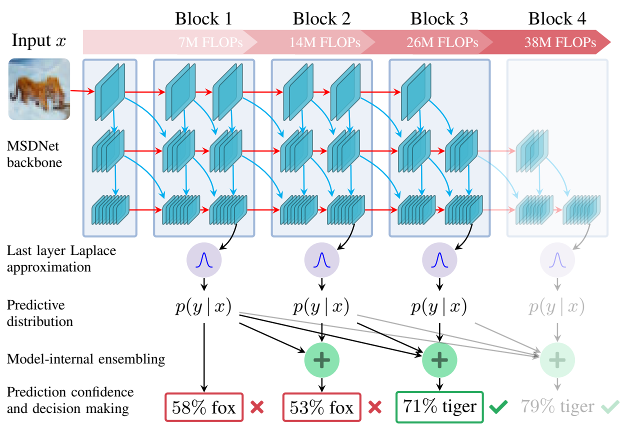


Figure 1. Increasing depth/scale of a dynamic neural network. Our probabilistic decision-making captures a calibrated confidence estimate, allowing better decisions from the intermediate classifiers on when to stop the evaluation (decision on whether to output the prediction is shown in red and green outline).

several stages (*cf*. Fig. 1). The adaptive early exiting is typically based on the confidence scores at the individual exits or learned gating functions [15]. Thus, it is crucial that the predictive densities or gating functions are robust and allow for trustworthy decision-making. However, current approaches are problematic as typical neural architectures are: *(i)* **miscalibrated** [14], *(ii)* **overconfident** [16], and *(iii)* their predictions do not capture **epistemic** uncertainties (uncertainty about the true model, see [22]).

Uncertainty quantification is the interest of probabilistic (or *Bayesian*) methods in deep learning. Bayesian deep learning [48] has recently gained increasing attention in the machine learning community as a means for uncertainty quantification (*e.g.*, [22, 49]) and model selection (*e.g.*, [3, 19]), compromising, among others, advancements on prior specification (*e.g.*, [11, 34–36]) and efficient approximate inference schemes (*e.g.*, [7, 25, 30]). Even though some of these advancements have recently found application in computer vision (*e.g.*, [41, 45, 47]), they have not found adoption for decision-making in DNNs.

In this work, we propose a new probabilistic treatment of the multiple exits of DNNs by applying a Bayesian formulation combined with efficient post-hoc approximate inference using multiple last-layer Laplace approximations [24]. Our probabilistic treatment reduces *overconfidence*, improves *calibration*, and captures *epistemic* uncertainties arising through the uncertainty about the true model without increasing the computational burden significantly. Moreover, we propose aggregating predictions across the exits to utilise uncertainties arising at earlier stages and show that accounting for uncertainties in decision-making (bottom in Fig. 1) outperforms the standard dynamic neural network (MSDNet [18]) on CIFAR-100, ImageNet, and Caltech-256.

Our contributions can be summarized as follows: *(i)* We introduce a probabilistic treatment for early exit dynamic neural networks (DNNs) utilising a Bayesian formulation. *(ii)* We present a computationally efficient post-hoc approach for uncertainty-aware decision-making by leveraging our efficient last-layer Laplace approximation implementation combined with model-internal ensembling, which works out of the box without retraining. *(iii)* Finally, we show on CIFAR-100, ImageNet, and Caltech-256 that our probabilistic treatment improves over a conventional approach in accuracy, capturing uncertainties, and calibration error.

1.1. Related work

Dynamic neural networks (DNNs, [15, 26, 42]) that utilise intermediate classifiers, allow early exit of easy samples during inference. This tailors the computation depth of each input sample at runtime and offers complementary performance gains to other efficiency optimisations. The early works developing the chain-structured models show limited performance due to the interference among different classifiers [21, 46], which is addressed by the proposed Multi-Scale DenseNet (MSDNet, [18]) via dense connections and a multi-scale structure. Moreover, the MSDNet is further improved from the aspects of reducing resolution redundancy [51] and training process [27, 39]. However, the confidence-based early exiting criterion applied in these aforementioned works can be problematic, and the generated confidence does not necessarily reflect the complexity of the input. Furthermore, the poor estimation of uncertainty in these models makes it difficult to decide which samples are easy and which are hard, and further vulnerable to the slow-down attacks in [17]. Despite these shortcomings, multi-exit models have been successfully applied for image segmentation [23], image caption prediction [10], and to natural language processing by implementing early exits into a BERT model [9, 43].

Bayesian deep learning [22, 37] allows including prior knowledge and assumptions into deep learning models, and formulates their predictions through a probabilistic treatment. Calculating the posterior distribution of a Bayesian neural network is usually intractable, and approximate inference techniques need to be used, such as variational inference [4], deep ensembles [25], MC dropout [12], or Laplace approximation [20, 24, 40]—each having strengths and weaknesses. A key guiding principle in this work is constraining the computational budget, which leads us to propose an approach that utilizes computationally light Laplace approximations and re-uses predictions of DNN intermediate classifiers in a model-internal ensemble. Prior work does not consider the overconfidence or calibration of confidence estimates that DNNs use to make decisions on which samples require more computational budget.

2. Background

We are concerned with image classification under computational budget restrictions subject to a labelled training data set, $\mathcal{D}_{\text{train}} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{n_{\text{train}}}$, where $\mathbf{x}_i$ is d-dimensional input (*e.g.*, RGB image with $d = 3 \times N_{\text{pixels}}$). Labels $\mathbf{y}_i$ are c-dimensional one-hot encoded vectors indicating the correct class label in the classification task. The computational budget restrictions are applied in the form of a budgeted batch classification setup, in which a fixed computational budget B—measured in average floating point operations per input sample (FLOPs)—must be distributed across a batch of test samples to achieve the highest possible accuracy. Here, a model is trained on the training set $\mathcal{D}_{\text{train}}$ with an 'unlimited' computational budget and tested on a set of test samples $\mathcal{D}_{\text{test}} = \{\mathbf{x}_j, \mathbf{y}_j\}_{j=1}^{n_{\text{test}}}$ using a *limited* average budget B per test sample.

For a DNN model having n_{block} intermediate classifiers (see Fig. 1 for a sketch), we refer to the predictive distribution of each of these classifiers as $p_k(\hat{\mathbf{y}}_i \mid \mathbf{x}_i), k = 1, 2, \ldots, n_{\text{block}}$. The feature representation before the last linear layer of each classifier is referred to as $\boldsymbol{\phi}_{i,k} = f_k(\mathbf{x}_i)$, and the parameters of the last linear layer are $\boldsymbol{\theta}_k = \{\mathbf{W}_k, \mathbf{b}_k\}$. The prediction $p_k(\hat{\mathbf{y}}_i \mid \mathbf{x}_i)$ is obtained from the feature representation $\boldsymbol{\phi}_{i,k}$ as follows: $p_k(\hat{\mathbf{y}}_i \mid \mathbf{x}_i) = \text{softmax}(\hat{\mathbf{z}}_{i,k})$, where $\hat{\mathbf{z}}_{i,k} = \mathbf{W}_k \boldsymbol{\phi}_{i,k} + \mathbf{b}_k$.

Our focus is on fixing the overconfidence of DNN architectures that dynamically adapt the network depth utilising early exiting by intermediate classifiers, each with increasing computational requirements. Implementing classifiers as early exits into a single network allows reusing computation and reduces the overall inference cost compared to using independent models of varying sizes. Early versions of such DNNs suffer from interference between classifiers during training [21, 46]. This problem can be alleviated by using dense connectivity between intermediate classifiers, and by utilising a multi-scale structure having fine and coarse-scale features throughout the network. This architecture is referred to as a Multi-Scale DenseNet (MSDNet, [18]), and we use it as a backbone to demonstrate our methods of improving uncertainty estimation, as at the time of writing it clearly outperforms other image classification DNNs that use intermediate classifiers. However, our methods are applicable to any DNN that utilises early exiting. Fig. 1 visualizes the structure

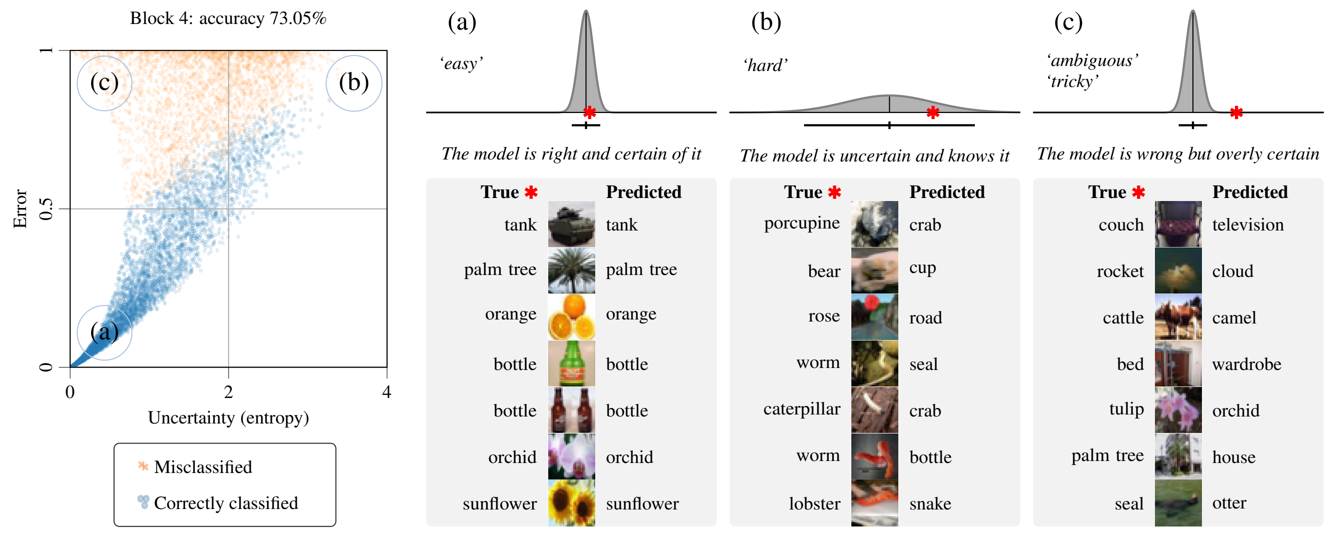


Figure 2. **Knowing what the model does not know.** An uncertainty–error scatter plot showing CIFAR-100 test set predictions for the last classifier of an MSDNet with Laplace and model-internal ensembling, and example images of three types corresponding to three areas in the uncertainty–error scatter plot. The error is defined as $1 - p(\hat{\mathbf{y}} = \mathbf{y})$, where $p(\hat{\mathbf{y}} = \mathbf{y})$ is the predicted probability on the correct label.

of the DNN backbone and the applied uncertainty quantification methods. The visualized FLOPs numbers are for the 'small' MSDNet model used for CIFAR-100 (see Sec. 4)

The challenge of efficiently using a DNN backbone with intermediate classifiers is the decision-making problem of when to exit the model. The goal is to use less capacity for clear 'easy' samples while unleashing more capacity for more difficult or tricky cases. To achieve this goal, the model must be well-calibrated and know when it is uncertain about its predictions. Many DNN models (*e.g.* [18, 46]) use predicted confidences from intermediate classifiers to make decisions on early exiting, but the calibration of these predictions is not controlled.

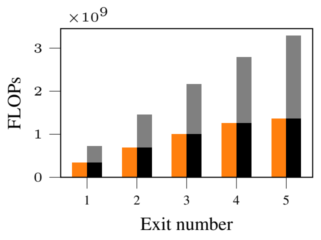
Uncertainty estimation in deep learning is often divided into estimating two different types of uncertainty [22]: *aleatoric* and *epistemic*. Aleatoric uncertainty is related to randomness intrinsic to the task at hand and cannot be reduced. Epistemic uncertainty is related to our knowledge of the task and can be reduced by learning more about the task, *e.g.*, by obtaining more data. In our problem setting of image classification, epistemic uncertainty is related to the model parameters. In Fig. 2, epistemic uncertainty is present in the predictions for images of type (b): the model has not learned the task well enough to classify these difficult sample images correctly. On the other hand, an example of aleatoric uncertainty is seen in some of the sample images of type (c): an image may contain objects from multiple classes, and the most prominent object is not necessarily labelled as the correct class ('ambiguous'), or some images may be completely mislabelled. We refer to this kind of samples as 'tricky'.

A Bayesian treatment to uncertainty estimation means that instead of obtaining a single point estimate of the model parameters $\boldsymbol{\theta}$ as the result of neural network training, Bayesian inference is used to obtain a *posterior distribution* over the model parameters given the training data $\mathcal{D}_{\text{train}}$:

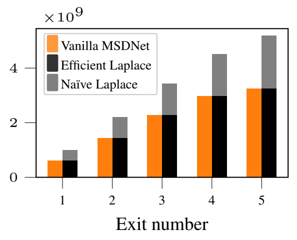
$$p(\boldsymbol{\theta} \mid \mathcal{D}_{\text{train}}) = \frac{p(\mathcal{D}_{\text{train}} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{\int_{\boldsymbol{\theta}} p(\mathcal{D}_{\text{train}}, \boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta}} = \frac{[\text{likelihood}] \times [\text{prior}]}{[\text{model evidence}]}. \tag{1}$$

Usually, calculating the exact posterior for a deep learning model is intractable. This means that the posterior of the model parameters must be approximated. One approach is to consider the $L-1$ first layers of a deep neural network with L layers as a fixed feature extractor and limit the Bayesian treatment to the last (L^{th}) layer. This drastically decreases the number of parameters for which the posterior distribution needs to be estimated. However, computations are usually still infeasible as no analytic solution exists, and further approximations are needed.

An efficient approximation to the posterior of the model parameters is the Laplace approximation, which performs a second-order Taylor expansion of Eq. (1) around the maximum *a posteriori* (MAP) estimate of the target distribution, resulting in a Gaussian distribution. The model parameters representing the MAP estimate can be found by maximising the unnormalised posterior: $p(\boldsymbol{\theta} \mid \mathcal{D}) \propto p(\mathcal{D}_{\text{train}} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta}) = p(\boldsymbol{\theta}, \mathcal{D})$, which is commonly assumed in log-space for numerical stability: $\log p(\boldsymbol{\theta}, \mathcal{D}_{\text{train}}) = \log p(\mathcal{D}_{\text{train}} \mid \boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$. In classification tasks, we typically minimise the cross-entropy loss, which is equivalent to maximising the log-likelihood. Moreover, commonly used regularisation methods such as weight decay can be interpreted as a log-prior. Hence, deep learning models



(a) ImageNet small model



(b) ImageNet large model

Figure 3. Test time computational cost of ▒ naïve and ■ efficient methods of sampling the Laplace predictive distribution (50 MC samples) compared against the 🟧 vanilla MSDNet, on ImageNet data. Results shown for each intermediate exit. Details in App. C.

learned with conventional training methods for classification tasks can be seen as maximising the unnormalised log-posterior and we, therefore, **directly obtain the MAP estimate** ($\boldsymbol{\theta}_{\text{MAP}}$) required for Laplace approximations through standard training. The Laplace approximation of the posterior is then formed using a multivariate Gaussian centred at the MAP estimate with covariance given by the inverse of the Hessian $\mathbf{H}$ of the negative log-posterior, *i.e.*, $\mathrm{N}(\boldsymbol{\theta}_{\text{MAP}} \,|\, \mathbf{H}^{-1})$ and $\mathbf{H} \coloneqq -\nabla^2_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta} \,|\, \mathcal{D}) \,|\,_{\boldsymbol{\theta}_{\text{MAP}}}$. The Hessian can be efficiently approximated using the generalised Gauss-Newton algorithm [5] or by Kronecker factorisation [33, 40].

3. Methods

Our aim is to couple an early exit DNN backbone with an uncertainty-aware decision-making process. For uncertainty quantification, we leverage the early exit structure of the DNN architecture and propose an approach that uses Laplace approximations and model-internal ensembling. The motivation for improving uncertainty estimation is that if the intermediate classifiers in the DNN can more accurately estimate the uncertainty in their predictions, they can better recognise which samples are hard and require further computation, and for which samples the prediction is already confident enough to be exited early.

3.1. Laplace approximation of a dynamic NN

To approximate the predictive posterior we propose an efficient implementation of a last-layer Laplace approximation for each intermediate classifier of the DNN. Using our computationally cheap last-layer approach enables us to stay resource-efficient while at the same time improving the decision-making within DNNs. Note that in this section we have dropped the index k for the intermediate exit for simplicity, as all operations are done independently for each exit.

To save on computational costs of sampling from a larger dimensional Gaussian on the last-layer parameters $\mathrm{N}(\boldsymbol{\theta}_{\text{MAP}}, \mathbf{H}^{-1})$, where $\boldsymbol{\theta}_{\text{MAP}} = \{\mathbf{W}_{\text{MAP}}, \mathbf{b}_{\text{MAP}}\}$ are the last-layer MAP parameters of one exit, we linearly project the Gaussian to a predictive distribution $p(\hat{\mathbf{z}}_i \,|\, \mathbf{x}_i)$. This directly allows us to sample pre-softmax outputs $\hat{\mathbf{z}}_i$ and reduces the dimensionality of the Gaussian to the number of classes c. The final prediction $\hat{\mathbf{y}}_i$ is then calculated by sampling n_{MC} samples from the predictive Gaussian: $\hat{\mathbf{y}}_i = \frac{1}{n_{\text{MC}}} \sum_{l=1}^{n_{\text{MC}}} \operatorname{softmax}(\hat{\mathbf{z}}_i^{(l)})$. If performed naïvely [24], this forces performing all sampling-related computation at test time, resulting in a per sample cost of $\text{FLOPs}_{\text{naïve}} = 2c^2(n_{\text{MC}} + 1) + \frac{1}{3}c^3 + 2p^2 + p - 1$ which grows cubically with the number of classes c (p is the feature space dimensionality: $\boldsymbol{\phi}_i \in \mathbb{R}^p$). Absorbing the last layer biases into the weight matrix allows us to shift most of the computation required for sampling to be performed before test time. The resulting efficient Laplace implementation for DNNs has a cost of $\text{FLOPs}_{\text{efficient}} = 2cn_{\text{MC}} + 2p^2 + 5p + 2$, making Laplace approximation computationally viable in the budget restricted regime. Fig. 3 shows a computational cost comparison for the naïve and efficient Laplace approximations. More detailed analysis is presented in App. C.

Applying a Laplace approximation only to the last linear layer at each intermediate exit now provides us with a Gaussian distribution $p(\hat{\mathbf{z}}_i \,|\, \mathbf{x}_i) = \mathrm{N}(\hat{\mathbf{W}}_{\text{MAP}}^{\top} \hat{\boldsymbol{\phi}}_i, (\hat{\boldsymbol{\phi}}_i^{\top} \mathbf{V} \hat{\boldsymbol{\phi}}_i) \mathbf{U})$ for each classifier. Here $\mathbf{V}^{-1} \otimes \mathbf{U}^{-1} = \mathbf{H}^{-1}$ is an approximate inverse Hessian, being a Kronecker factorisation of the generalised Gauss–Newton matrix and $\hat{\boldsymbol{\phi}}_i = (\boldsymbol{\phi}_i^{\top}, 1)^{\top}$ are the features after augmenting the biases into the weights. Samples from this distributions are calculated as $\hat{\mathbf{z}}_i^{(l)} = \hat{\mathbf{W}}_{\text{MAP}}^{\top} \hat{\boldsymbol{\phi}}_i + (\hat{\boldsymbol{\phi}}_i^{\top} \mathbf{V} \hat{\boldsymbol{\phi}}_i)^{\frac{1}{2}} (\mathbf{L}\mathbf{g}^{(l)})$, where $\mathbf{g}^{(l)} \sim \mathrm{N}(\mathbf{0}, \mathbf{I})$ and $\mathbf{L}$ is the Cholesky factor of $\mathbf{U}$.

To ensure well calibrated predictions from the Laplace approximation, it is useful to utilise temperature scaling [14] on the sampled predictions $\hat{\mathbf{y}}_i$. In practice, this means dividing each sampled pre-softmax output $\hat{\mathbf{z}}_i^{(l)}$ with a temperature scaling parameter T before taking the softmax. Also the Laplace prior variance σ is a hyperparameter affecting the results. We use a different value of T and σ for each classifier in the network. To choose appropriate values for the temperature scaling and prior variance parameters, we perform a grid search over possible pairs of values of T and σ, selecting the pair of values that minimises the negative log-predictive density score on the validation set for the classifier in question. Details on this grid search are in App. A.

3.2. Model-internal ensembling

To improve the predictive uncertainty and robustness of the DNN predictions, we utilise the idea of ensembling multiple predictions together [25]. However, using deep ensembles, where M independent networks are trained, is not feasible in the budget-restricted scenario. Instead, we utilise the predictions of intermediate classifiers in the dynamic neural network to form the ensemble and refer to this as a *model-internal ensemble* (MIE). The predictions from different intermediate classifiers are neither independent nor equal, as they are predictions from different stages of the same computational pipeline, and later classifiers have more

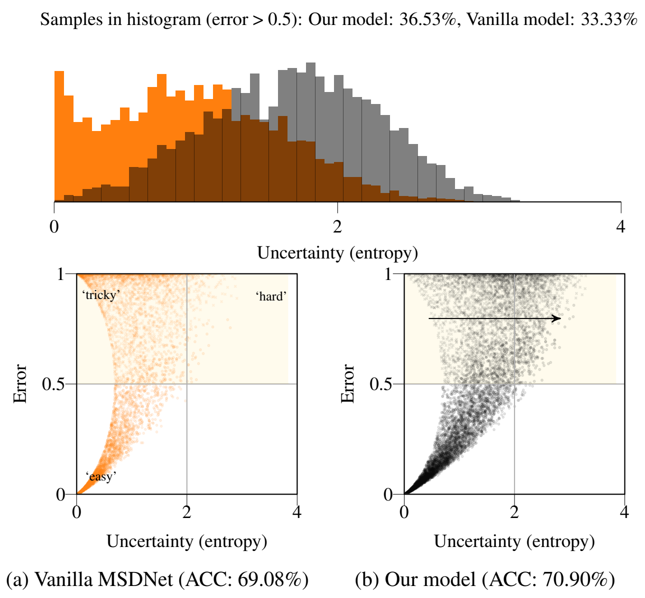


Figure 4. Comparison of uncertainty–error scatter plots from the **second to last classifier** of a vanilla MSDNet and our model with Laplace and model-internal ensembling. The uncertainty histogram on the top shows points with error > 0.5: the uncertainty should be high for the model to be able to recognize these samples as 'hard', and continue their evaluation to the next block. For our model these samples have a high uncertainty, while the vanilla MSDNet is overconfident. See Sec. 3.3 for definition of error.

capacity compared to earlier classifiers. To account for the difference in capacity in the MIE members, we can scale their influence to the final prediction in proportion to their computational complexity. Predictions from intermediate classifiers for forming an MIE are readily available and require no additional computation, as they need to be calculated to make the decision whether to continue computation further in the network. The model-internal ensemble prediction for the k^{th} classifier in an early exit DNN is:

$$p_k^{\text{ens}}(\hat{\mathbf{y}}_i \,|\, \mathbf{x}_i) = \frac{1}{\sum_{l=1}^{k} w_l} \sum_{m=1}^{k} w_m \, p_m(\hat{\mathbf{y}}_i \,|\, \mathbf{x}_i). \quad (2)$$

This is a weighted average of intermediate classifiers up to classifier k, for which we are calculating the MIE prediction. Later classifiers have more predictions in the average to aggregate, as more already calculated intermediate predictions are available. The weights w_m are the computational costs of the DNN in FLOPs up to classifier m. The added computational cost of MIE is marginal (see App. C for details).

3.3. Illustrating the intuition

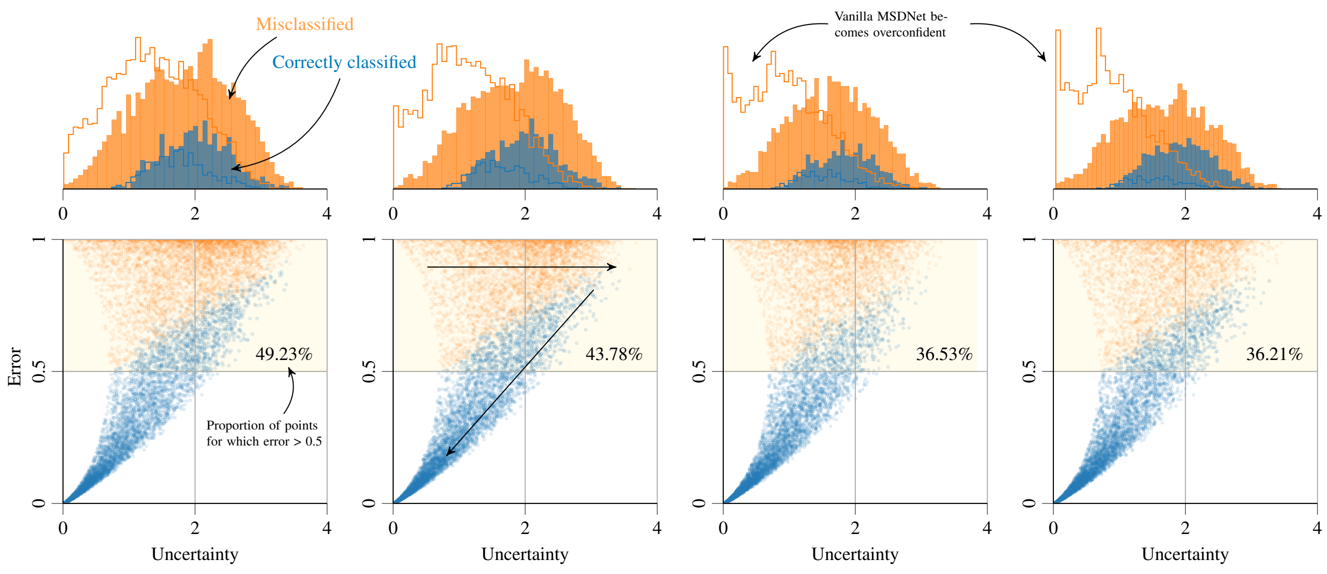
Our model hinges on the realization that properly capturing epistemic uncertainty is key to making informed decisions. This is visible in Figs. 2 and 4 that demonstrate how better uncertainty quantification can improve the decision-making in a DNN. In Fig. 2, samples of type (c) are ones that the model has predicted with high confidence, but the predicted label is incorrect. These can be mislabelled samples or samples that are very ambiguous. However, for a poorly calibrated model, the type (c) samples may also include a large number of samples that the model predicted overconfidently for no intuitive reason, and that should instead be of type (b): hard samples that the current stage of the model can't accurately classify. This can be seen in Fig. 4 where the standard DNN model has a large number of overconfidently predicted samples in the upper left corner of the scatter plot. In this figure, we can see that the improved calibration of our model allows these samples to move to the upper right corner of the scatter plot. This change prevents these predictions from exiting at the current intermediate classifier and instead allows for potentially improving the prediction in the later steps of the DNN. In figures, the error is defined as $1 - p(\hat{\mathbf{y}} = \mathbf{y})$, where $p(\hat{\mathbf{y}} = \mathbf{y})$ is the predicted probability on the correct label.

For the decision-making to be efficient in a DNN, each intermediate classifier should have calibrated uncertainties and not show too many samples in the upper left area of the uncertainty–error scatter plots. Fig. 5 shows that this is true for our model, and the picture samples in Fig. 2 under the label (c) show that the samples remaining in the upper-left corner are mostly ambiguous samples, which even a calibrated model would predict incorrectly but confidently.

4. Experiments

We performed a series of experiments on benchmark image classification tasks to assess the improvements obtained through our probabilistic treatment applied to an early-exit DNN. For each model size for all data sets, a common backbone DNN (MSDNet [18]) was trained minimising the L2 regularised sum of cross-entropy losses computed for all exits on the training set. We trained three different-sized DNN backbone models on each data set to cover a larger range of budgets. Depending on the desired budget, only one of these models would be used at a time. We refer to these models as 'small', 'medium', and 'large'.

After training, we evaluated each model on the test set in a budgeted batch classification setup. Early exiting decisions were based on model predicted confidence, for which thresholds $t_k, k = 1, 2, \ldots, n_{\text{block}}$ were calculated on the validation set. We refer to [18] for details on the calculation of the thresholds. We report the Top-1 and Top-5 accuracies of each model over a range of computational budgets measured in average floating point operations (FLOPs) per test sample. In addition, following the recommendations for better validation metrics in image analysis [31], we compare the negative log-predictive density (NLPD) and the expected calibration error (ECE). Note that NLPD captures both accuracy and uncertainty quantification quality while ECE assesses only calibration, *i.e.*, how consistent the confidence



(a) **Block 1** (ACC: 61.73%) (b) **Block 2** (ACC: 66.56%) (c) **Block 3** (ACC: 70.90%) (d) **Block 4** (ACC: 73.05%)

Figure 5. Model certainty grows with capacity. The bottom scatter plots show ● correctly classified and ● misclassified test points on an uncertainty vs. error axis. The uncertainty of points in the top half is summarized as a histogram above each scatter plot (in scale with each other). Our model consistently adds capacity to calibrate itself to tricky cases ('tricky' $\rightarrow$ 'hard') and improves accuracy ('hard' $\rightarrow$ 'easy') over the blocks, while the vanilla MSDNet model (solid line in histograms, for reference) does not. Fig. 8 in App. B shows corresponding scatter plots for vanilla MSDNet, and uncertainty histograms including all points in the scatter plots for both models. See Sec. 3.3 for definition of error.

Table 1. Table of Top-1/Top-5 accuracy, negative log-predictive density (NLPD), and expected calibration error (ECE) for different models on CIFAR-100 and ImageNet data. All numbers are averages over a range of computational budgets in the budgeted batch classification setup. 'MIE Laplace T_{opt} σ_{opt}'-model corresponds to 'Our model' that is referred to in other figures. MIE stands for model-internal ensembling.

	$(n_{\text{train}}, d, c, n_{\text{batch}})$	CIFAR-100 (50000, 3072, 100, 64)				IMAGENET (1281167, 150528, 1000, 256)			
		Top-1 ACC ↑	Top-5 ACC ↑	NLPD ↓	ECE ↓	Top-1 ACC ↑	Top-5 ACC ↑	NLPD ↓	ECE ↓
Small	MSDNet (vanilla)	69.25	90.48	1.498	0.182	68.15	**88.22**	1.338	0.019
	+ Laplace T_{opt} σ_{opt}	69.06 −0.19	90.58 +0.10	1.208 −0.289	0.073 −0.109	68.10 −0.05	88.18 −0.04	**1.337** −0.001	**0.015** −0.005
	+ MIE	**69.97** +0.72	90.88 +0.40	1.218 −0.280	0.080 −0.102	68.27 +0.12	88.13 −0.10	1.355 +0.017	0.055 +0.036
	+ MIE Laplace T_{opt} σ_{opt}	69.84 +0.59	**91.09** +0.61	**1.133** −0.364	**0.017** −0.165	**68.31** +0.16	88.11 −0.11	1.356 +0.018	0.052 +0.032
Medium	MSDNet (vanilla)	74.12	91.94	1.549	0.190	72.78	91.01	1.123	0.033
	+ Laplace T_{opt} σ_{opt}	73.92 −0.20	92.01 +0.06	1.070 −0.479	0.083 −0.107	72.72 −0.07	91.03 +0.03	**1.118** −0.005	**0.018** −0.015
	+ MIE	**75.03** +0.91	92.97 +1.03	1.011 −0.538	0.050 −0.140	72.98 +0.20	**91.12** +0.11	1.119 −0.004	0.042 +0.009
	+ MIE Laplace T_{opt} σ_{opt}	74.99 +0.86	**93.23** +1.29	**0.944** −0.605	**0.026** −0.164	**73.04** +0.26	90.96 −0.05	1.121 −0.002	0.031 −0.003
Large	MSDNet (vanilla)	75.36	92.78	1.475	0.178	74.33	91.57	1.066	0.050
	+ Laplace T_{opt} σ_{opt}	75.32 −0.05	92.83 +0.05	0.996 −0.479	0.075 −0.103	74.29 −0.04	91.53 −0.04	1.053 −0.013	**0.020** −0.030
	+ MIE	76.32 +0.95	93.50 +0.72	0.949 −0.525	0.061 −0.117	**74.82** +0.49	**91.88** +0.30	**1.029** −0.037	0.028 −0.022
	+ MIE Laplace T_{opt} σ_{opt}	**76.34** +0.98	**93.84** +1.05	**0.885** −0.590	**0.025** −0.152	74.80 +0.47	91.81 +0.24	1.032 −0.034	0.032 −0.019

scores are with the posterior probabilities. See App. A.5 for more details on the metrics. In figures, 'Our model' refers to an MSDNet backbone using Laplace approximation and MIE, while optimizing temperature scales and Laplace prior variances in a grid search. We additionally trained DenseNet and ResNet models as baselines, see App. A.4 for details on them. For more baseline results we refer to [18, Sec. 5.2].

Ablation studies were performed to investigate the individual contribution of the Laplace approximation and model-internal ensembling (MIE) to the model performance, testing models that use either only Laplace or only MIE. The results of this ablation study are included in Tab. 1 and Tab. 2. Results of a more comprehensive ablation study are in Tabs. 6 and 7 in App. B. Note that as Laplace and MIE are applied on the same trained vanilla MSDNet model that is used on its own, the differences in the results are not from randomness between different training runs. Laplace approximation improves the uncertainty quantification properties of the model by lowering NLPD and ECE values, whereas MIE usually improves both accuracy and uncertainty quantification properties. Considering overall performance over all four metrics (Top-1 and Top-5 accuracy, NLPD, and ECE), we can see that Laplace and MIE together give the best performance on CIFAR-100, whereas on ImageNet and Caltech-256 using MIE alone or together with Laplace both give roughly

Table 2. Table of Top-1/Top-5 accuracy, NLPD, and ECE for different models on Caltech-256 data. All numbers are averages over a range of computational budgets in the budgeted batch classification setup. 'MIE Laplace T_{opt} σ_{opt}'-model corresponds to 'Our model' that is referred to in other figures.

	$(n_{\text{train}}, d, c, n_{\text{batch}})$	CALTECH-256 (25607, 150528, 257, 128)			
		Top-1 ACC ↑	Top-5 ACC ↑	NLPD ↓	ECE ↓
Small	MSDNet (vanilla)	61.0	78.2	2.16	0.18
	+ Lap T_{opt} σ_{opt}	60.5 −0.5	78.1 −0.1	1.86 −0.29	**0.05** −0.13
	+ MIE	**61.9** +0.9	78.8 +0.6	1.94 −0.21	0.08 −0.10
	+ MIE Lap T_{opt} σ_{opt}	61.7 +0.6	**79.0** +0.8	**1.81** −0.34	0.09 −0.09
Medium	MSDNet (vanilla)	63.8	80.2	1.98	0.17
	+ Lap T_{opt} σ_{opt}	63.4 −0.4	79.9 −0.3	1.74 −0.24	**0.07** −0.10
	+ MIE	**65.1** +1.3	**81.4** +1.2	1.72 −0.26	0.08 −0.09
	+ MIE Lap T_{opt} σ_{opt}	64.3 +0.5	81.3 +1.1	**1.65** −0.33	0.08 −0.09
Large	MSDNet (vanilla)	64.9	80.7	1.90	0.16
	+ Lap T_{opt} σ_{opt}	64.7 −0.2	80.7 +0.0	1.65 −0.25	**0.04** −0.12
	+ MIE	**65.9** +0.9	82.4 +1.8	1.62 −0.28	0.06 −0.10
	+ MIE Lap T_{opt} σ_{opt}	65.6 +0.7	**82.5** +1.8	**1.58** −0.32	0.09 −0.07

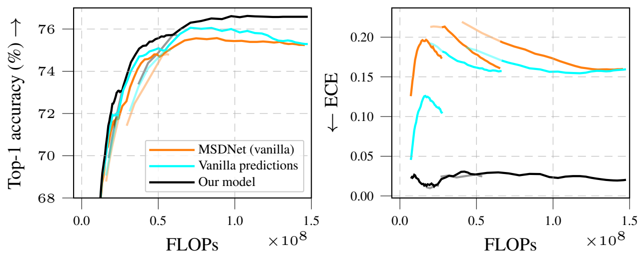


Figure 6. A decision-making experiment, where vanilla MSDNet and 'Our model' are compared to results obtained using our model for decision-making, and taking predictions from vanilla MSDNet (labelled 'Vanilla predictions'). Full results in Fig. 9 in App. B.

equally good results. Despite some inconsistency in the combination of uncertainty quantification methods giving best performance, it is clear that improving uncertainty estimation improves the DNN performance over the vanilla model. The range of budgets over which results were averaged to obtain the numbers in Tabs. 1 and 2 are listed in Tab. 3 in App. A.

CIFAR-100 experiment results are visualized in Fig. 7 and numerical results are presented in Tab. 1. The three model sizes, small, medium, and large, are plotted as separate curves in Fig. 7. From the results in Fig. 7 we see that our model improves Top-1 and Top-5 accuracies over all tested computational budget levels, compared to the vanilla MSDNet model (in-line with results in [18]), and improves uncertainty quantification and calibration properties, which is seen in the decrease of negative log-predictive density (NLPD) and expected calibration error (ECE). From the curves, we can pinpoint at 10^8 FLOPs an improvement of 1.2 %-points in Top-1 accuracy and 1.1 %-points in Top-5 accuracy. We also note that although the vanilla MSDNet has clearly superior Top-1 accuracy compared to baseline ResNet and DenseNet models, it has poor performance in terms of NLPD and ECE in comparison to the baselines. CIFAR-100 experiment details are in App. A.1.

To investigate the contribution of better decision-making on the improved predictive performance, we performed an experiment separating the improvement due to better decision-making from the improvement due to better predictions at individual intermediate exits. In this experiment, we replaced the vanilla model decision-making with the decision-making of our model, while using the vanilla model predictions for calculating the results. Results on CIFAR-100 are in Fig. 6 showing that our approach improves both decision-making (orange vs. light blue) and prediction quality (light blue vs. black). Interestingly, apart from improving accuracy, better decision-making also improves calibration, as seen from the improved ECE (details in App. B).

ImageNet experiment results are visualized in Fig. 7, and numerical results are presented in Tab. 1. Fig. 7 shows that on larger computational budgets, our model achieves improvements over the vanilla MSDNet model on all metrics. We note that the ECE numbers achieved by the vanilla MSDNet on ImageNet data are much better than those achieved by the vanilla MSDNet on CIFAR-100 data, suggesting relatively good calibration especially on lower budgets, and resulting in limited usefulness of uncertainty quantification methods. As the computational budget increases, vanilla MSDNet becomes less calibrated, negatively affecting the decision-making at the intermediate classifiers, which is seen in worse accuracy compared to our model. The good calibration of the vanilla MSDNet on ImageNet can be explained by the relatively small model size preventing overfitting the data. The largest MSDNet model used for ImageNet here has 62 million parameters, while the current state-of-the-art model [6] has 2440 million parameters. From the accuracy curves in Fig. 7 we can pinpoint at $2.5 \cdot 10^9$ FLOPs an improvement of 0.63 %-points in Top-1 accuracy and 0.34 %-points in Top-5 accuracy. ImageNet experiment details are in App. A.2.

Caltech-256 is an image classification data set with similar resolution images as ImageNet, but with a small number of training samples. For Caltech-256 the same backbone DNN models were used as for ImageNet. Results are visualized in Fig. 7 where we see similar trends as for CIFAR-100, with uncertainty quantification techniques improving performance on all metrics. Numerical results are presented in Tab. 2, where different uncertainty quantification methods excel at different metrics, but improve consistently over the vanilla MSDNet. We note that the same backbone DNNs that were used for ImageNet, benefit more from our uncertainty quantification methods when used on Caltech-256, possibly due to the smaller training set size that allows overfitting with these model sizes. We can deduce that uncertainty quantification in DNNs is beneficial especially when the standard model is overfitting the data, which is usually the case. From the accuracy curves in Fig. 7 we can pinpoint at $2.5 \cdot 10^9$ FLOPs an improvement of 1.4 %-points in Top-1 accuracy and 1.8 %-points in Top-5 accuracy. Caltech-256 experiment details are in App. A.3.

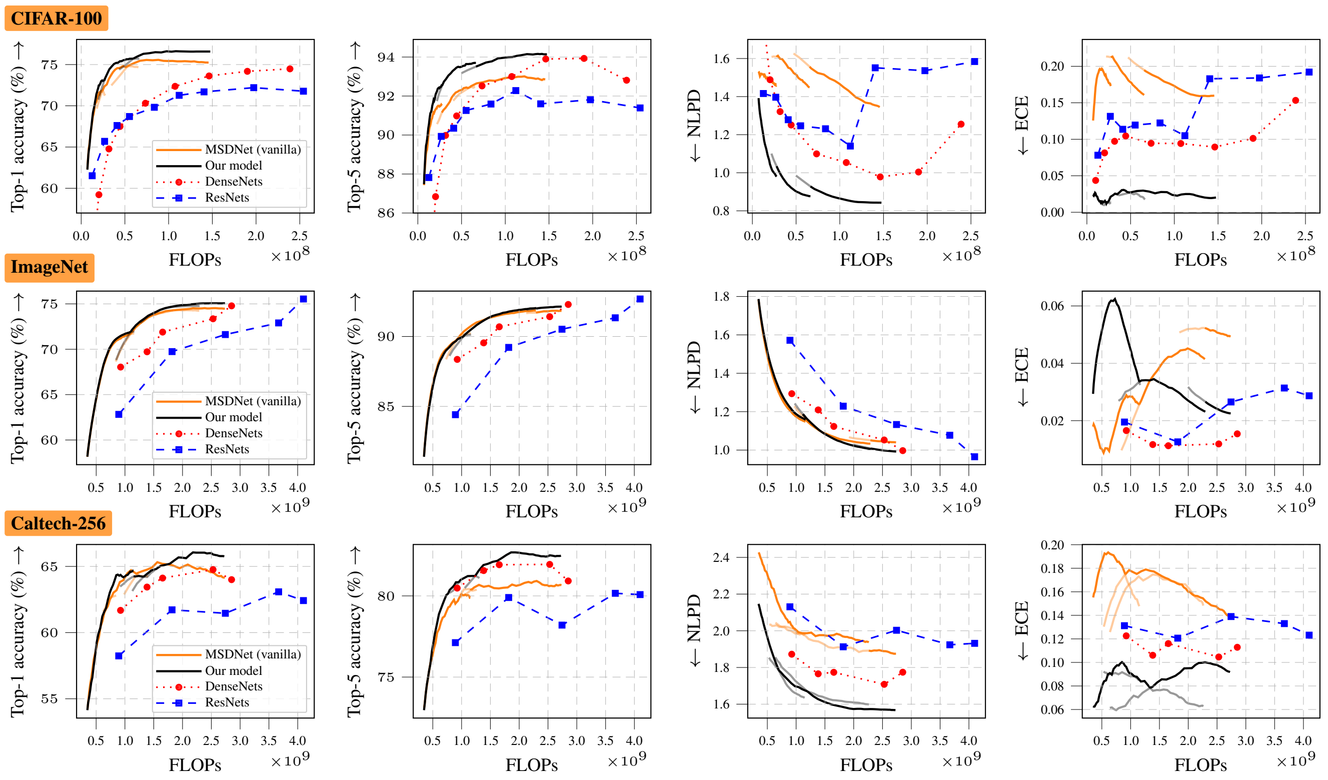


Figure 7. Accuracy (Top-1 & Top-5) and uncertainty metrics (NLPD and ECE) on a budgeted batch classification task as a function of average computational budget per image (FLOPs) on different data sets with a small/medium/large model, and ResNet/DenseNet baselines.

5. Conclusion and Discussion

We have demonstrated the importance of uncertainty quantification in dynamic neural networks (DNNs). For this purpose, we employed a probabilistic treatment of DNNs and leveraged a computationally efficient post-hoc posterior approximation through multiple last-layer Laplace approximations together with model-internal ensembling (MIE). Our approach substantially improves the internal decision-making process fundamental to DNNs, as evidenced by improved Top-1 and Top-5 accuracy, NLPD, and ECE on CIFAR-100, ImageNet, and Caltech-256. We found that uncertainty quantification and calibration are especially crucial for large-scale models that overfit training data—stressing their importance in DNNs applied to real-world scenarios.

Why Laplace and MIE? Uncertainty quantification comes in many facets and, if done in DNNs, has to be computationally efficient to be viable for budget-constrained applications. In this work, we proposed to employ last-layer Laplace approximations at each exit of the DNN. Although the Laplace approximation is a more crude approximation to the posterior in comparison to techniques such as deep ensembles, it provides a good trade-off between accuracy of the approximation and computational costs, which is essential to DNNs. In the experiments, we showed that our efficient Laplace approach adds little computational overhead and provides overall NLPD and ECE improvements in most experiments. In addition, we showed that MIE can further boost performance with little additional costs by informing consecutive blocks in the DNN about the predictive uncertainties of previous exits. As a conclusion, the Laplace approximations account for epistemic uncertainties of each exit *independently* while MIE allows us to incorporate dependence between the exits—hence, complementary to each other.

Broader Impact. This work contributes towards improving the resource and energy efficiency of often prohibitively expensive deep learning models. For example, Microsoft reported [1] that answering Bing queries using BERT requires 2000 Azure GPU virtual machines to run in parallel. Uncertainty quantification is typically seen as a means of improving robustness and even safety of deep learning models, generally adding to the compute. This work takes the opposite direction leveraging uncertainty to reduce the overall number of floating point operations required for making predictions—while also providing more reliable uncertainty estimates for downstream applications. Improvements in DNNs allow using more powerful models in edge computing and on mobile hardware, and decreases the total energy usage of a heavy deep-learning task on non-constrained hardware.

The codes to replicate the results are available at `https://github.com/AaltoML/calibrated-dnn`.

References

[1] Bing delivers its largest improvement in search experience using Azure GPUs. `https://tinyurl.com/tzhj3o8`. Accessed: 2022-11-11. 8

[2] Sergei Alyamkin, Matthew Ardi, Alexander C. Berg, Achille Brighton, Bo Chen, Yiran Chen, Hsin-Pai Cheng, Zichen Fan, Chen Feng, Bo Fu, Kent Gauen, Abhinav Goel, Alexander Goncharenko, Xuyang Guo, Soonhoi Ha, Andrew Howard, Xiao Hu, Yuanjun Huang, Donghyun Kang, Jaeyoun Kim, Jong-gook Ko, Alexander Kondratyev, Junhyeok Lee, Seungjae Lee, Suwoong Lee, Zichao Li, Zhiyu Liang, Juzheng Liu, Xin Liu, Yang Lu, Yung-Hsiang Lu, Deeptanshu Malik, Hong Hanh Nguyen, Eunbyung Park, Denis Repin, Liang Shen, Tao Sheng, Fei Sun, David Svitov, George K. Thiruvathukal, Baiwu Zhang, Jingchi Zhang, Xiaopeng Zhang, and Shaojie Zhuo. Low-power computer vision: Status, challenges, and opportunities. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9(2):411–421, 2019. 1

[3] Javier Antorán, David Janz, James Urquhart Allingham, Erik A. Daxberger, Riccardo Barbano, Eric T. Nalisnick, and José Miguel Hernández-Lobato. Adapting the linearised laplace model evidence for modern deep learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pages 796–821. PMLR, 2022. 1

[4] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017. 2

[5] Aleksandar Botev, Hippolyt Ritter, and David Barber. Practical Gauss-Newton optimisation for deep learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 557–565. PMLR, 2017. 4

[6] Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, et al. Symbolic discovery of optimization algorithms. *arXiv preprint arXiv:2302.06675*, 2023. 7

[7] Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux–Effortless Bayesian deep learning. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 20089–20103. Curran Associates, Inc., 2021. 1

[8] Jia Deng, Wei Dong, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarcgical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009. 1

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[10] Zhengcong Fei, Xu Yan, Shuhui Wang, and Qi Tian. Deecap: Dynamic early exiting for efficient image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12206–12216. IEEE, 2022. 2

[11] Vincent Fortuin, Adrià Garriga-Alonso, Florian Wenzel, Gunnar Rätsch, Richard E. Turner, Mark van der Wilk, and Laurence Aitchison. Bayesian neural network priors revisited. In *Third Symposium on Advances in Approximate Bayesian Inference*, 2021. 1

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059. PMLR, 2016. 2

[13] Abhinav Goel, Caleb Tung, Yung-Hsiang Lu, and George K. Thiruvathukal. A survey of methods for low-power deep learning and computer vision. In *Proceedings of the 6th IEEE World Forum on Internet of Things (WF-IoT)*, pages 1–6. IEEE, 2020. 1

[14] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 2017. 1, 4

[15] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7436–7456, 2022. 1, 2

[16] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 41–50, 2019. 1

[17] Sanghyun Hong, Yiğitcan Kaya, Ionuţ-Vlad Modoranu, and Tudor Dumitraş. A panda? No, it's a sloth: Slowdown attacks on adaptive multi-exit neural network inference. In *International Conference on Learning Representations (ICLR)*, 2020. 2

[18] 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 *International Conference on Learning Representations (ICLR) Workshops*, 2018. 1, 2, 3, 5, 6, 7, 13

[19] Alexander Immer, Matthias Bauer, Vincent Fortuin, Gunnar Rätsch, and Mohammad Emtiyaz Khan. Scalable marginal likelihood estimation for model selection in deep learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pages 4563–4573. PMLR, 2021. 1

[20] Alexander Immer, Maciej Korzepa, and Matthias Bauer. Improving predictions of Bayesian neural nets via local linearization. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 130 of *Proceedings of Machine Learning Research*, pages 703–711. PMLR, 2021. 2

[21] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-deep networks: Understanding and mitigating network overthinking. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Pro-*

ceedings of Machine Learning Research, pages 3301–3310. PMLR, 09–15 Jun 2019. 2

[22] Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems 30 (NIPS)*, pages 5574–5584. Curran Associates, Inc., 2017. 1, 2, 3

[23] Alexandros Kouris, Stylianos I. Venieris, Stefanos Laskaridis, and Nicholas D. Lane. Multi-exit semantic segmentation networks. In *Proceedings of the 17th European Conference on Computer Vision (ECCV)*, volume 13681 of *Lecture Notes in Computer Science*, pages 330–349. Springer, 2022. 2

[24] Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being Bayesian, even just a bit, fixes overconfidence in ReLU networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 5436–5446. PMLR, 2020. 2, 4

[25] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems 30 (NIPS)*, pages 6402–6413. Curran Associates, Inc., 2017. 1, 2, 4

[26] 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*, pages 1–6, 2021. 2

[27] Hao Li, Hong Zhang, Xiaojuan Qi, Ruigang Yang, and Gao Huang. Improved techniques for training adaptive deep networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1891–1900, 2019. 2

[28] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proceedings of the 13th European Conference on Computer Vision (ECCV)*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014. 1

[29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002. IEEE, 2021. 1

[30] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 13132–13143. Curran Associates, Inc., 2019. 1

[31] Lena Maier-Hein, Annika Reinke, Evangelia Christodoulou, Ben Glocker, Patrick Godau, Fabian Isensee, Jens Kleesiek, Michal Kozubek, Mauricio Reyes, Michael A. Riegler, Manuel Wiesenfarth, Michael Baumgartner, Matthias Eisenmann, Doreen Heckmann-Nötzel, A. Emre Kavur, Tim Rädsch, Minu Dietlinde Tizabi, Laura Ación, Michela Antonelli, Tal Arbel, Spyridon Bakas, Peter Bankhead, Arriel Benis, M. Jorge Cardoso, Veronika Cheplygina, Beth Cimini, Gary S. Collins, Keyvan Farahani, Bram van Ginneken, Daniel A. Hashimoto, Michael M. Hoffman, Merel Huisman, Pierre Jannin, Charles E. Kahn, Alexandros Karargyris, Alan Karthikesalingam, Hannes Kenngott, Annette Kopp-Schneider, Anna Kreshuk, Tahsin M. Kurç, Bennett A. Landman, Geert Litjens, Amin Madani, Klaus H. Maier-Hein, Anne L. Martel, Peter Mattson, Erik Meijering, Bjoern H. Menze, David Moher, Karel G. M. Moons, Henning Müller, Felix Nickel, Brennan Nichyporuk, Jens Petersen, Nasir Rajpoot, Nicola Rieke, Julio Saez-Rodriguez, Clarisa Sánchez Gutiérrez, Shravya Shetty, Maarten van Smeden, Carole H. Sudre, Ronald M. Summers, Abdel A. Taha, Sotirios A. Tsaftaris, Ben Van Calster, Gaël Varoquaux, and Paul F. Jäger. Metrics reloaded: Pitfalls and recommendations for image analysis validation. *arXiv preprint arXiv:2206.01653*, 2022. 5

[32] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. `https://github.com/pytorch/vision`, 2016. 13

[33] James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, volume 37 of *Proceedings of Machine Learning Research*, pages 2408–2417. PMLR, 2015. 4

[34] Lassi Meronen, Christabella Irwanto, and Arno Solin. Stationary activations for uncertainty calibration in deep learning. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, pages 2338–2350. Curran Associates, Inc., 2020. 1

[35] Lassi Meronen, Martin Trapp, and Arno Solin. Periodic activation functions induce stationarity. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 1673–1685. Curran Associates, Inc., 2021. 1

[36] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *International Conference on Learning Representations (ICLR)*, 2019. 1

[37] Radford M Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, Toronto, Canada, 1995. 2

[38] 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 *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 8024–8035. Curran Associates, Inc., 2019. 12

[39] Mary Phuong and Christoph H Lampert. Distillation-based training for multi-exit architectures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1355–1364, 2019. 2

[40] Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable Laplace approximation for neural networks. In *International Conference on Learning Representations (ICLR)*, 2018. 2, 4

[41] Subhankar Roy, Martin Trapp, Andrea Pilzer, Juho Kannala, Nicu Sebe, Elisa Ricci, and Arno Solin. Uncertainty-guided

source-free domain adaptation. In *Proceedings of the 17th European Conference on Computer Vision (ECCV)*, volume 13685 of *Lecture Notes in Computer Science*, pages 537–555. Springer, 2022. 1

[42] Simone Scardapane, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. Why should we add early exits to neural networks? *Cognitive Computation*, 12(5):954–966, 2020. 2

[43] Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A. Smith. The right tool for the job: Matching model and instance complexities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6640–6651. Association for Computational Linguistics, 2020. 1, 2

[44] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, pages 3645–3650. Association for Computational Linguistics, 2019. 1

[45] Yihong Sun, Adam Kortylewski, and Alan L. Yuille. Amodal segmentation through out-of-task and out-of-distribution generalization with a Bayesian model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1205–1214. IEEE, 2022. 1

[46] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *International Conference on Pattern Recognition (ICPR)*, pages 2464–2469. IEEE, 2016. 2, 3

[47] Jianfeng Wang and Thomas Lukasiewicz. Rethinking bayesian deep learning methods for semi-supervised volumetric medical image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 182–190. IEEE, 2022. 1

[48] Andrew Gordon Wilson. The case for Bayesian deep learning. `https://cims.nyu.edu/~andrewgw/caseforbdl.pdf`, 2019. Technical Report. 1

[49] Andrew Gordon Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, pages 4697–4708. Curran Associates, Inc., 2020. 1

[50] Xiaowei Xu, Yukun Ding, Sharon Xiaobo Hu, Michael Niemier, Jason Cong, Yu Hu, and Yiyu Shi. Scaling for edge inference of deep neural networks. *Nature Electronics*, 1(4):216–222, 2018. 1

[51] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution adaptive networks for efficient inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2369–2378, 2020. 2