QUTE: Quantifying Uncertainty in TinyML models with Early-exit-assisted ensembles

Nikhil P Ghanathe [1] **Steve Wilton** [1]

Abstract

Existing methods for uncertainty quantification incur massive memory and compute overhead, often requiring multiple models/inferences. Hence they are impractical on ultra-low-power KB-sized TinyML devices. To reduce overhead, prior works have proposed the use of early-exit networks as ensembles to quantify uncertainty in a single forward-pass. However, they still have a prohibitive cost for tinyML. To address these challenges, we propose QUTE, a novel resource-efficient early-exit-assisted ensemble architecture optimized for tinyML models. QUTE adds additional output blocks at the final exit of the base network and distills the knowledge of early-exits into these blocks to create a diverse and lightweight ensemble architecture. Our results show that QUTE outperforms popular prior works, and improves the quality of uncertainty estimates by 6% with **3.1× lower model size** on average compared to the most relevant prior work. Furthermore, we demonstrate that QUTE is also effective in detecting co-variate shifted and out-of-distribution inputs, and shows competitive performance relative to G-ODIN, a state-of-the-art generalized OOD detector.

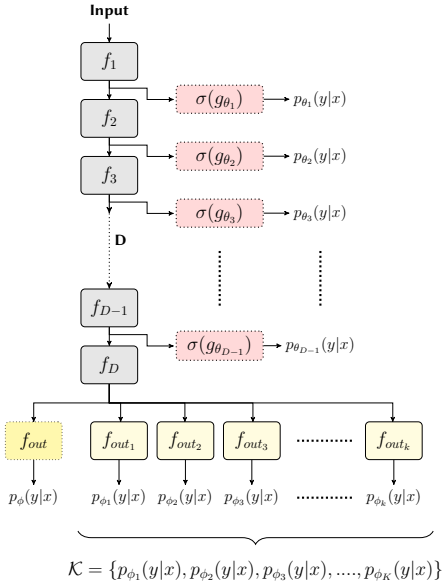


Figure 1: QUTE architecture. $\{f_{out}(.)\}_{k=1}^{K}$ represents the additionally created output blocks at the final exit, which are *assisted* by the early-exit blocks $\{g_{\theta_i}(.)\}_{i=1}^{D-1}$ during training to promote diversity (see Figure 3). Here, $K = D - 1$. f_{out} is the output block of base network. For inference, all dotted blocks (early-exit & f_{out}) are removed and the mean of prediction vectors of all members of $\mathcal{K}$ is calculated before final prediction.

1. Introduction

Recent advancements in embedded systems and machine learning (ML) have produced a new class of edge devices containing powerful ML models. These milliwatt-scale KB-sized devices, often termed *TinyML* devices, have low compute and memory requirements. They are often deployed in remote environments with no availability of true labels. This makes them susceptible to both out-of-distribution (OOD) and covariate-shifted data caused by environmental and sensor variations that manifest often unpredictably in the field.

The ability of tinyML devices to accurately measure the uncertainty of their predictions is crucial for two reasons. First, these devices generate data that is frequently used in critical decision-making. For example, in the context of an autonomous vehicle (AV), uncertain predictions could result in the downstream system making cautious driving decisions (Tang et al., 2022). Second, tinyML devices often operate in harsh and remote environments (Vargas et al., 2021). If the inputs to the model changes (perhaps due to spatter, fog, frost, noise, motion blur, etc.), being aware of the model's unreliability in its predictions may prompt an

*Equal contribution [1]Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada. Correspondence to: Nikhil P Ghanathe <nikhilghanathe@ece.ubc.ca>.

engineer to take remedial action such as repairing/replacing sensors or adapting to changes in the input distribution.

We distinguish between two types of uncertainty: 1) that due to semantic changes in inputs i.e., out-of-distribution (OOD), and 2) that due to non-semantic changes i.e., covariate-shifted/corrupted inputs. The latter category, which we refer to as *corrupted-in-distribution* (CID) data, occurs when partially-malfunctioning sensors or environmental factors cause inputs that are corrupted versions of expected data elements (E.g., fogged camera lens).

Unfortunately, modern neural networks are poor in estimating uncertainty of their predictions (Ovadia et al., 2019). Many prior works have proposed uncertainty-aware networks: ensemble networks (Lakshminarayanan et al., 2017) have been found to be extremely effective in providing good uncertainty estimates. Alternatively, early-exit networks have been converted into ensembles (Antoran et al., 2020; Qendro et al., 2021; Ferianc & Rodrigues, 2023). However, both these approaches incur high memory/compute overheads and are untenable for tinyML.

In this paper, we propose QUTE (Figure 1), a novel and resource-efficient early-exit-assisted ensemble architecture that enables high-quality uncertainty quantification in tinyML models in the context of both in-distribution (ID) and corrupted-in-distribution (CID) data. Our approach appends additional classification heads that are lightweight at the final exit to create ensemble members, and crucially, these classification heads are assisted by the early-exits in a manner similar to the Early-View assistance technique (Ghanathe & Wilton, 2023) to promote diversity. QUTE performs better than prior methods in estimating uncertainty caused due to CID error sources, and on-par with prior methods for uncertainty due to OOD. Our approach has significantly less memory and compute overhead compared to prior works (3.1$\times$ smaller models and 3.8$\times$ fewer FLOPS compared to the most relevant prior work (Qendro et al., 2021)). We further show that higher uncertainty is correlated with a drop in accuracy. We evaluate QUTE's ability to detect such accuracy drop events caused by CID against prior methods, and show that it either outperforms or achieves comparable performance compared to prior works including G-ODIN (Hsu et al., 2020), a generalized OOD detection framework. Additionally, in our experiments we observe an intriguing result that OOD detectors like G-ODIN perform poorly on extremely small models. We demonstrate this empirically and show QUTE's superior ability to detect OOD samples on models with lower sizes. To the best of our knowledge, this is the first early-exit ensemble architecture for uncertainty quantification optimized for tinyML models.

This paper is organized as follows. Section 2 presents related work. The context and problem formulation are given in Section 3. Our approach is described in Section 4. The experimental methodology and evaluation results are provided in Sections 5 and 6 respectively. Section 7 concludes the paper.

2. Related Work

Uncertainty quantification Bayesian neural networks (BNN) are well-suited to quantify uncertainty of a model (Blundell et al., 2015; Hernández-Lobato & Adams, 2015; Teye et al., 2018), However, they are parameter-inefficient and incur a high resource/compute overhead. Alternatively, bayesian-approximation techniques have been proposed that address these challenges. Monte Carlo Dropout (MCD) (Gal & Ghahramani, 2016) creates implicit ensembles by enabling dropout during inference and running multiple inference passes. Recently, ensembles have been shown to produce good uncertainty estimates (Lakshminarayanan et al., 2017; Zaidi & Zela, 2020; Wenzel et al., 2020; Rahaman et al., 2021). However, this too requires multiple inferences of individual networks, and is impractical in terms of memory for tinyML. Some other prior works have proposed multi-input and multi-output networks that combine multiple independent networks into one (Havasi et al., 2021; Ferianc & Rodrigues, 2023), but they do not scale well and remain impractical for tinyML. Alternatively, prior works (Qendro et al., 2021; Ferianc & Rodrigues, 2023) have leveraged early exit networks to create implicit ensembles. However, they still incur a prohibitive cost (memory in particular), which make them unsuitable for tinyML (Section 3). The closest work to QUTE is EE-ensemble (Qendro et al., 2021), which uses outputs of early-exits as ensemble members.

OOD detetection There has been a plethora of work to detect OOD samples, which represents a semantic shift from in-distribution (Yang et al., 2021; 2022; Zhang et al., 2023; Liu et al., 2021; Hendrycks & Gimpel, 2018) . However, there have been fewer works that deal with detecting corrupted-ID, which represent a non-semantic shift w.r.t. ID, and are much harder to detect (Liang et al., 2020). Some prior works utilize corrupted-ID/covariate-shifted ID to better generalize on OOD (Bai et al., 2023; Katz-Samuels et al., 2022). However, this assumes knowledge of such data to tune the model hyperparameters (sometimes in the field) for better OOD detection, which is difficult on tinyML devices. The most relevant work to our setting that leverages *only* training data is generalized-ODIN (G-ODIN) (Hsu et al., 2020). It adds 1) a preprocessing layer that perturbs the input image and 2) decomposes confidence score for better OOD detection. However, the preprocessing might prove costly on tinyML devices.

Early-exit networks Early-exit networks add intermediate exits along the length of the base network thereby, providing avenues for reduction in average inference time (Teer-

apittayanon et al., 2016; Kaya et al., 2019; Huang et al., 2017; Bonato & Bouganis, 2021; Ghanathe & Wilton, 2023). Many prior works either attach multiple early-exits or include additional learning layers at the early-exits, which does not suit tinyML. (Kaya et al., 2019) utilizes early-exits to study the problem of *network overthinking*, which shows that representations learnt by earlier layers in a network are sometimes more accurate than that of the later ones. This also explains the overconfidence in deep modern neural networks for CID/OOD inputs. Further, (Ghanathe & Wilton, 2023) introduces a early-exit architecture optimized for tinyML models. In addition, it introduces the *early-view assistance* method to mitigate network overthinking in neural networks. We leverage a modified version of this method in our work to create a diverse and lightweight ensemble architecture (see Section 4).

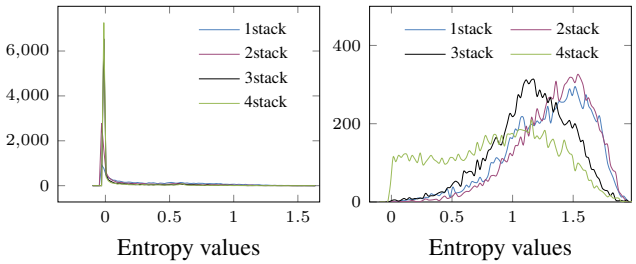
3. Background and Problem formulation

Consider an *in-distribution* dataset $\mathrm{S_{ID}} = \{x_n, y_n\}_{n=1}^{N}$ of size N where, x_n and y_n is the n$^{\text{th}}$ input sample and its corresponding true label respectively. A discriminative model $\mathcal{M}_\Theta(x)$ learns parameters Θ on $\mathrm{S_{ID}}$ and outputs a class posterior probability vector $p_\Theta(y|x)$, which yields a predicted label $\hat{y}$. For a classification problem, $\hat{y} \in \{1, 2, ..L\}$, where L is the number of classes. Uncertainty quantification methods endeavor to improve uncertainty estimation quality of $\mathcal{M}_\Theta(x)$ on S_{ID}. Specifically, we want to calibrate the model such that its *predictive confidence* ($\mathcal{C}_{\mathcal{M}_\Theta}$) is in sync with its accuracy ($\mathcal{A}_{\mathcal{M}_\Theta}$). The predictive confidence is given by,

$$\mathcal{C}_{\mathcal{M}_\Theta}(x) = \max_{l \in \{1,2,..,L\}} p_\Theta(y = l|x) \tag{1}$$

Confidence-calibration is a well studied problem, and it helps achieve synergy between the predicted probabilities and ground truth correctness likelihood (Guo et al., 2017). A well-calibrated model will see a commensurate drop in $\mathcal{C}_{\mathcal{M}_\Theta}$ as $\mathcal{A}_{\mathcal{M}_\Theta}$ drops.

When $\mathcal{M}_\Theta$ is deployed in a real-world scenario, it may encounter either 1) out-of-distribution data ($\mathrm{S_{OOD}}$) or 2) a corrupted/covariate-shifted version ($\mathrm{S_{CID}}$), both of which can cause $\mathcal{A}_{\mathcal{M}_\Theta}$ to drop. Therefore, it is imperative that the uncertainty estimates are reliable even in the presence of $\mathrm{S_{CID}}/\mathrm{S_{OOD}}$. However, we find that despite good uncertainty estimation quality on $\mathrm{S_{ID}}$, many uncertainty-aware networks see a drop in quality and remain overconfident, particularly on S_{CID} (Hsu et al., 2020). Interestingly, we find that the model becomes *less overconfident* to $\mathrm{S_{CID}}$ as its *size shrinks*. This phenomenon is illustrated in Figure 2. Figure 2 plots the histogram of predictive entropy of the predictions of Resnet (He et al., 2016) on CIFAR10, with 4 different sizes ranging from 1 to 4 residual stacks. Further, the predictive entropy is plotted for 1) test set of $\mathrm{S_{ID}}$ and



(a) In-distribution (b) Contrast corruption

Figure 2: Histogram of the predictive entropy on CIFAR10 in-distribution (ID) test set (left), and ID test set corrupted with increased contrast (right) using Resnet model with {1,2,3,4} residual stacks

2) test set of $\mathrm{S_{ID}}$ corrupted by increasing contrast of all images. As expected, for ID, the distribution of predictive entropy in all models is concentrated around 0. However, for the corrupted dataset, we find that Resnet-1stack displays the highest predictive entropy followed by 2stack, 3stack and 4stack models in that order. It is evident that smaller models are more *sensitive* to corruptions.

One possible way to leverage the power of smaller models to obtain higher quality uncertainty estimates on both S_{ID} and S_{CID} is by using early-exit networks to create an ensemble. For example, we observe that the popular Deep ensemble (DEEP) method (Lakshminarayanan et al., 2017) has a negative log-likelihood (NLL) of 0.227 on MNIST ID while the Early-exit ensemble (EE-ensemble) (Qendro et al., 2021) has a NLL of 0.266 (lower is better). In contrast, on MNIST-C (Mu & Gilmer, 2019), which has 15 corrupted versions of MNIST ID such as fog, spatter, noise, blur and so on, the average NLL of DEEP and EE-ensemble are 3.553 and 3.299 respectively. It is evident that while DEEP has better calibration on ID, EE-ensemble, which leverages early-exit outputs to create an ensemble has far better uncertainty estimation capability on corrupted-ID/S_{CID}. This result demonstrates the significance of early-exits in achieving higher quality uncertainty overall.

However, we find that this approach is extremely resource-hungry, and is impractical on tinyML devices. For example, our preliminary evaluations with a 4-layer CNN on MNIST using the EE-ensemble method (Qendro et al., 2021) increased the model parameters by 2.7×. Thus, most prior works do not optimize for memory. They often 1) require buffering of early-exit outputs (Ferianc & Rodrigues, 2023; Antoran et al., 2020) or 2) require additional learning layers at the early-exits (Qendro et al., 2021). Both approaches are not suitable for tiny-edge devices. We present an alternative strategy to leverage the knowledge of early-exits in creating ensembles that is extremely resource-efficient. We show that the assistance of early-exits in QUTE enables it to obtain superior uncertainty estimates on both S_{ID} and

S_{CID}. We further show that the higher quality of uncertainty estimation on S_{CID} allows QUTE to detect a drop in model's accuracy due to corruptions more precisely, which helps maintain model reliability.

4. QUTE

A neural network (NN) like the one shown in Figure 1, with depth D is composed of several blocks of linear/non-linear operations (e.g., convolution, pooling) that are stacked. The NN consists of $D-1$ intermediate blocks $\{f_i(.)\}_{i=1}^{D-1}$; $f(.) = f_1(.) \circ f_2(.) \circ f_3(.) .... \circ f_{D-1}(.) \circ f_D(.)$ and an output block $f_{out}(.)$. $f_0(.)$ is the input block. In the base network, the network processes the input $f_0(.)$ through each block $f_i(.)$ until layer D. Finally, the output of layer D is processed by the output block $f_{out}(.)$ to produce the prediction $p_\phi(y|x)$.

QUTE adds early exit blocks along the length of the base network as shown in Figure 1. Each early-exit block is allowed to learn a probabilistic predictive distribution $p_{\theta_i}(y|x)$ such that

$$p_{\theta_i}(y|x) = \sigma(g_{\theta_i}(a_i)) \tag{2}$$

where $g_{\theta_i}(.)$ is the early exit block at layer i with parameters θ_i, a_i is the intermediate output of layer i and σ is the output activation (E.g., dense+softmax). Furthermore, QUTE creates additional classification heads/output blocks $\{f_{out_k}\}_{k=1}^{K}$ at the final exit (after the final layer) as shown in Figure 1 such that the number of additional output blocks is equal to the number of early exits (K). These output blocks constitute the ensemble $\mathcal{K}$. In Figure 1, K is set to $D-1$ and f_{out} is the original output block at the final exit of the base network. In practice, K is a hyperparameter that depends on computation/resource budget and the required ensemble size $|\mathcal{K}|$. Therefore, the ensemble size is bounded below by the required quality of uncertainty estimates (Gal & Ghahramani, 2016; Lakshminarayanan et al., 2017) and has an upper bound of $D-1$. Thus, $\{f_{out_k}\}_{k=1}^{K}$ produces K predictions. We discard the output block of the base network, f_{out} while creating the ensemble because we find that it is overconfident and hampers the quality of uncertainty quantification.

$$\mathcal{K} = \{p_{\phi_1}(y|x), p_{\phi_2}(y|x), p_{\phi_3}(y|x), ..., p_{\phi_K}(y|x)\} \tag{3}$$

Further, for each f_{out_k}, its corresponding early-exit, g_{θ_k} *assists* it in the final classification using a modified version of *early-view assistance* method (Ghanathe & Wilton, 2023) described in the next section.

4.1. Network training with QUTE

Figure 3 illustrates the architecture of *early-view assistance* method, where each final exit block $f_{out_k}(.)$ and its *assisting* early-exit $g_{\theta_k}(.)$ is shown. As shown in the figure, at the k^{th} output block, we create an additional learning layer $h_{\phi_k}(.)$ after the final layer of the base network. $h_{\phi_k}(.)$ is identical to $g_{\theta_k}(.)$ and is a depthwise convolution layer in our evaluations owing to its low computation and memory demand. The input to all $\{h_{\phi_k}(.)\}_{k=1}^{K}$ is the output of the layer $f_D(.)$ of the base network. The output of $h_{\phi_k}(.)$ (termed as *early-view*) is fed to a non-linear (dense) layer $\sigma(.)$ for classification. At the early-exits, the input to the early-exit block $g_{\theta_k}(.)$ is the intermediate output of layer i (a_i). Figure 1 shows early-exits after each layer i of the base network. However, in practice, fewer early-exits at different depths are added depending on resource/compute budget.

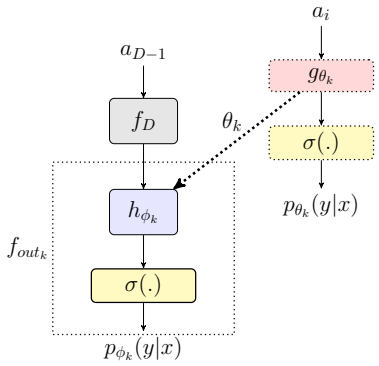


Figure 3: Early-view assistance architecture for $\{f_{out_k}\}_{k=1}^{K}$. Parameters θ_k from early-exit are transferred to h_{ϕ_k} block during training. During inference, early-exit blocks are removed.

After attaching early-exits (EE) and early-view-assisted (EV) exits to the untrained base network (the grey blocks in Figure 1), all weights of the aforementioned blocks are learned simultaneously during training. Unlike some prior works that use extra data (e.g., adversarial samples), we only use the available training data. The early-exits are trained with a *weighted-loss* function, similar to previous studies (Kaya et al., 2019). Further, we assign a *higher weighting* factor, w_{EV_k} for losses at EV-exits compared to that of f_{out} to minimize the effect of an overconfident f_{out}. In addition, we increase each w_{EV_k} by δ such that $w_{EV_k} = w_{EV_{k-1}} + \delta$. This is done to further promote diversity across the EV-exits (see Appendix A.1 for details). During training, at the end of each training batch, the weights from $g_{\theta_k}(.)$ are transferred to $h_{\phi_k}(.)$ i.e., $\phi_k = \theta_k$ till completion of training. The weights of the base network are frozen for the last 10% of epochs so that the EV-exits are trained in isolation to promote diversity while continuing weight transfer from early-exits. In this way, the early-exit knowledge is transferred to the final exit(s). Since each early-exit, at different depths learns to extract different features, they are inherently diverse. Therefore, using the filter weights of the early-exits on the output of base network's

final layer fosters diversity across the ensemble and boosts its *sensitivity* to corruptions as mentioned in Section 3. In contrast, without the EV-assistance method, the various final exits risk learning the same predictive distribution, leading to poor ensemble behavior, which in turn leads to poorly calibrated uncertainty estimates. We compare the effectiveness of the EV-assistance method in Section 6.4.

4.2. Network inference with QUTE

During inference, the early-exit blocks are removed from the model architecture. Therefore, the final architecture contains only the base network in addition to the output blocks $\{f_{out_k}(.)\}_{k=1}^{K}$, which are significantly lightweight compared to early-exits. In addition, we also remove f_{out}, the original output block. The forward pass produces $|\mathcal{K}|$ predictions from the ensemble members of $\mathcal{K}$. The final prediction ($p_\Theta(y|x)$) is obtained by calculating the mean of all the prediction vectors of $\mathcal{K}$.

$$p_\Theta(y|x) = \frac{1}{|\mathcal{K}|}\Big(\sum_{k=1}^{K} p_{\phi_k}(y|x)\Big) \tag{4}$$

where, $\{\phi_1, \phi_2, \phi_3, ..., \phi_k\}$ are the parameters of the $\{f_{out_1}(.), f_{out_2}(.), f_{out_3}(.), ..., f_{out_K}(.)\}$ respectively.

5. Evaluation Methodology

We evaluate QUTE in two settings: 1) uncertainty estimation quality (Section 6.1) and 2) OOD/CID detection capability (Section 6.2, 6.3).

Datasets and Models We evaluate QUTE on three image classification tasks of differing complexities. 1) MNIST (LeCun et al., 1998) on a 4-layer CNN, 2) CIFAR10 (Krizhevsky, 2009) on Resnet-8 (from MLPerf tiny benchmark suite (Banbury et al., 2021)) and 3) TinyImagenet (Le & Yang, 2015) on MobilenetV2 (Howard et al., 2017). For CID datasets, we use corrupted versions of ID i.e., MNIST-C (Mu & Gilmer, 2019), CIFAR10-C and TinyImagenet-C (Hendrycks & Dietterich, 2019), which corrupt the ID with various corruptions. For example, CIFAR10-C includes 19 different types of corruptions with 5 severity levels, which gives us 19×5=95 corrupted versions of CIFAR-10 ID. For OOD datsets, we use commonly-used datasets from the literature i.e., FashionMNIST (Xiao et al., 2017) for MNIST and SVHN (Netzer et al., 2011) for CIFAR10. More details on models and datasets used can be found in Appendix A.1.

Additionally, we evaluate QUTE on a popular tinyML audio classification task of keyword spotting (Zhang et al., 2017) from MLPerf Tiny using Speech Commands dataset (Warden, 2018). The Speech commands contains utterances of 35 words. Like (Banbury et al., 2021), we train a 4-layer depthwise-separable convolution model (DS-CNN) to recognize ten words and use the rest as OOD.

Placement of Early-exits The number of early-exits (K) is influenced primarily by resource budget in tinyML models. Moreover, there are only a handful locations in a tinyML model where we can place the early-exits. Our strategy is to place early-exits at equally-spaced locations along the length of the base network. For MNIST, we place two early exits after the first and second convolution layer of the 4-layer CNN. For Resnet-8 with 3 residual stacks, we place early exits after each of the first two residual stacks. Further, we place 5 early-exits for MobilenetV2 at equally-spaced locations starting from the input. A detailed analysis of the effect of number of early-exits on uncertainty estimation is provided in Appendix A.4.

5.1. Uncertainty quantification

For uncertainty estimation, we report the class-weighted F1 score and two proper scoring metrics (Gneiting & Raftery, 2007): Brier score (BS) (Brier, 1950), which measures the accuracy of predicted probabilities and negative log-likelihood((NLL), which measures how close the predictions are to the ground truth (see Appendix A.5). We report these metrics on both ID and CID. For results on CID, we report the average of the metrics over all corrupted datasets. Details of corrupted datasets is provided in Appendix A.2.

5.2. Corruption and Out-of-distribution detection

We evaluate QUTE's capability to detect a drop in accuracy caused by CID inputs. To that end, we formulate a realistic scenario where we monitor the *confidence* (maximum of the softmax scores) to detect a possible drop in accuracy. We formulate this problem as a binary classification problem. First, we construct corrupted datasets that contain both ID and CID samples by appending ID with each corrupted version of ID described previously. For example, experiments with MNIST-C containing 15 types of corruptions will lead to 15 ID+CID datasets. Second, for each method, we iterate over *only* ID and obtain model predictions while computing the moving average of accuracy of the past m predictions using a sliding-window. The accuracy distribution thus obtained is denoted as $\mathcal{A}_{ID}$ and the accuracy of the sliding-window is denoted as $\mathcal{A}_{SW}$. In our empirical evaluations, we find that $m = 100$ is a reasonable sliding-window size. Third, we compute the mean (μ_{ID}) and standard-deviation (σ_{ID}) of $\mathcal{A}_{ID}$. Next, we iterate over all ID+CID datasets while computing the moving average of confidence of the past m predictions using a sliding-window, denoted by $\mathcal{C}_{SW}$. A true positive event occurs when $\mathcal{C}_{SW} < \rho \; and \; \mathcal{A}_{SW} \leq \mu_{ID} - 3 \cdot \sigma_{ID}$, where ρ is a user-defined threshold. Finally, we iterate over each dataset while computing $\mathcal{C}_{SW}$ and $\mathcal{A}_{SW}$, and record

			In-distribution data			*Corrupted-in-distribution data*		
Model	**Params**	**FLOPS**	**F1 (↑)**	**BS (↓)**	**NLL (↓)**	**F1 (↑)**	**BS (↓)**	**NLL (↓)**
MNIST								
- BASE	3.9K	155.2K	0.910±0.002	0.013±0.000	0.292±0.006	0.534±0.300	0.072±0.056	5.218±8.359
- MCD	3.9K	310.4K	0.886±0.004	0.018±0.000	0.382±0.004	0.556±0.303	0.067±0.053	4.872±7.879
- DEEP	7.9K	310.4K	0.931±0.005	0.010±0.000	0.227±0.002	0.574±0.316	0.060±0.046	3.553±5.391
- EE-ensemble	10.9K	1.51M	0.939±0.002	0.011±0.000	0.266±0.005	0.571±0.323	0.059±0.042	3.299±4.996
- QUTE	4.4K	158.2K	**0.941±0.004**	**0.009±0.000**	**0.199±0.010**	**0.591±0.305**	**0.058±0.045**	**2.769±3.798**
CIFAR10								
- BASE	78.6K	25.2M	0.834±0.005	0.023±0.000	0.523±0.016	0.603±0.202	0.060±0.035	1.918±1.612
- MCD	78.6K	50.5M	0.867±0.002	0.019±0.000	0.396±0.003	**0.739±0.087**	**0.036±0.011**	**0.791±0.282**
- DEEP	157.3K	50.5M	**0.877±0.003**	**0.017±0.000**	**0.365±0.015**	0.651±0.193	0.051±0.030	1.348±0.994
- EE-ensemble	477.5K	26.1M	0.854±0.001	0.021±0.000	0.446±0.011	0.648±0.173	0.049±0.024	1.256±0.784
- QUTE	86.4K	25.3M	0.858±0.001	0.020±0.000	0.428±0.019	0.637±0.183	0.053±0.028	1.406±0.963
TinyImagenet								
- BASE	2.51M	50.6M	0.351±0.005	0.004±0.000	5.337±0.084	0.171±0.062	0.006±0.000	9.852±2.081
- MCD	2.51M	253M	0.332±0.004	**0.003±0.000**	2.844±0.028	0.199±0.048	**0.004±0.000**	**4.149±0.639**
- DEEP	12.57M	253M	0.414±0.006	**0.003±0.000**	3.440±0.049	0.200±0.074	0.005±0.000	7.148±1.896
- EE-ensemble	3.53M	52.4M	**0.430±0.005**	**0.003±0.000**	**2.534±0.046**	**0.207±0.077**	**0.004±0.000**	4.888±1.327
- QUTE	2.58M	51.3M	0.395±0.014	0.004±0.000	3.700±0.123	0.175±0.064	0.005±0.000	7.044±1.417

Table 1: Calibration Metrics for all baselines on ID data. For F1, higher is better, and for BS and NLL, lower is better. The best results are marked in bold. For ID data, all results are mean ± std-dev for 3 independent splits of ID test data. For results on CID data, we construct corrupted datasets for each corruption (all severity levels) and average the metrics over all of them. See Appendix A.2 for more details.

instances when $\mathcal{C}_{SW} < \rho$. We vary ρ from 0 to 1 in step sizes of 0.1 and repeat the whole experiment. Appendix A.2 provides additional details on problem formulation (such as defining true/false positives & true/false negatives in this context) and experimental setup. In addition, Appendix A.2 also provides details on how the ID+CID datasets are constructed. Finally, we report the area under precision-recall curve (AUPRC), the highest F1 score along with precision and recall associated with the highest F1. All metrics are *averaged* over all corrupted datasets. The same procedure is followed for OOD detection except that the sliding-window size is set to 1 because all predictions are treated as incorrect on OOD.

5.3. Baselines

We compare QUTE against several state-of-the-art methods listed below.

BASE: Unmodified implementations of models evaluated.

Monte Carlo Dropout (MCD) (Gal & Ghahramani, 2016): a bayesian approximation technique that creates implicit ensembles by activating *dropout* during inference. For fairness, dropout layers are placed at the exact locations as early-exit points. We use a dropout rate of 0.1 for MNIST, and 0.2 for CIFAR10 and TinyImagenet. We collect probability vectors from K inference passes before computing their mean, where K is number of early-exits.

Early-exit Ensembles (EE-ensemble) (Qendro et al., 2021): attaches multiple early-exits with additional fully-connected (FC) layers. The size of the FC layers is chosen such that the model delivers the best uncertainty estimation. All early-exit prediction vectors including that of the final exit are averaged to get the final ensemble prediction. We place the early-exits at the same locations as QUTE.

Deep Ensembles (DEEP) (Lakshminarayanan et al., 2017): explicit ensemble of models with same base model architecture but with different random weight initializations. The number of ensemble members = K.

Generalized-ODIN (G-ODIN) (Hsu et al., 2020): a generalized OOD detection method that decomposes the confidence score and introduces a preprocessing layer that perturbs the inputs to make confidence a better scoring function to detect OOD samples. The perturbation magnitude is determined on a small held-out validation set.

6. Results and Discussions

First, we compare the calibration metrics of all baselines against QUTE for ID and CID in Section 6.1. We demonstrate that predictions from QUTE are consistently better-calibrated (closer to ground truth) for both ID and various CID datasets with the **3.1× lower model size** and **3.8× fewer FLOPS** on average compared to prior works. Next, we evaluate QUTE's capability to detect drops in accuracy due to CID in Section 6.2. Our results show that QUTE is superior to all baselines, including EE-ensemble, the most relevant prior work. Furthermore, on OOD detection, we demonstrate that QUTE outperforms all uncertainty-estimation baselines comfortably and is even better than G-ODIN on MNIST (Section 6.3). Finally, we conduct an ablation study to assess the effectiveness of EV-assistance method (Section 6.4). In addition, we also analyze the ef-

fect of number of early-exits on uncertainty estimation in Appendix A.4.

6.1. Uncertainty quantification

Table 1 reports the calibration metrics for all three datasets we evaluate on both ID and CID data. Our results show that QUTE either *outperforms* or is *competitive* with existing methods on all metrics. All methods perform better than BASE, except MCD on MNIST. Further inquiry revealed that applying dropout during inference on the extremely tiny CNNs takes away from its model capacity thereby leading to a drop in accuracy and calibration. Therefore, MCD is not suitable for extremely tiny models. Further, as expected, DEEP and EE-ensemble also surpass the performance of BASE owing to their larger model capacities. However, QUTE consistently outperforms both with the *lowest memory-footprint* i.e. *44.3%* and *59.6% lower* model sizes compared to DEEP and EE-ensemble respectively. In addition, on corrupted-MNIST, QUTE achieves the highest F1 and the lowest NLL despite corruptions, which indicates its robustness to covariate-shift and superiority for extremely small model sizes. On CIFAR10, MCD, DEEP and QUTE perform similarly on ID. QUTE is either equal in performance or comes in a close second compared to other baselines despite using significantly lower resources. In fact, QUTE uses 5.5× fewer resources than EE-ensemble while delivering better uncertainty estimation quality (+4%) and F1. EE-ensemble seeks to match the model capacity of the early-exits with that of the final exit to achieve better calibration by adding additional fully-connected layers. This leads to the massive increase in model size. Interestingly, MCD is the most resilient on corrupted-CIFAR10 by a considerable margin followed by DEEP, QUTE and EE-ensemble. Unlike in MNIST, we employ a higher-capacity Resnet-8 model for CIFAR10 which can better withstand dropout of neurons during inference.

Next, we also experiment with a larger dataset, TinyImagenet on a larger model, MobilenetV2 to demonstrate QUTE's effectiveness and resource-efficiency in the traditional ML space. Our results show that QUTE is on-par with prior methods, which are optimized for such larger models. All methods achieve significantly higher calibration quality compared to BASE both on ID and CID data. Both MCD and EE-ensemble outperform other baselines. EE-ensemble achieves the best F1 on ID and CID data. This is consistent with previous studies that show addition of early-exits improves the network accuracy (Teerapittayanon et al., 2016). Interestingly, MCD is still the most robust method to corruptions. Following MCD and EE-ensemble, QUTE betters DEEP's uncertainty estimation quality by +2% on TinyImagenet-C despite having the lowest model size by a large margin. QUTE's model size is 79.4% and 26.9% lower than DEEP and EE-ensemble respectively. The superiority of QUTE wr.r.t. DEEP underlines the upper hand of early-exit assistance in obtaining well-calibrated models on corrupted data.

MNIST	AUPRC (↑)	F1 (↑)	Precision (↑)	Recall (↑)
- BASE	0.54	0.73	0.73	0.72
- MCD	0.56	0.72	0.73	0.71
- DEEP	0.59	0.85	0.84	0.86
- EE-Ensemble	**0.67**	0.9	0.9	0.9
- G-ODIN	0.52	0.73	0.74	0.72
- QUTE	**0.67**	**0.92**	**0.91**	**0.93**

Table 2: MNIST CID detection results.

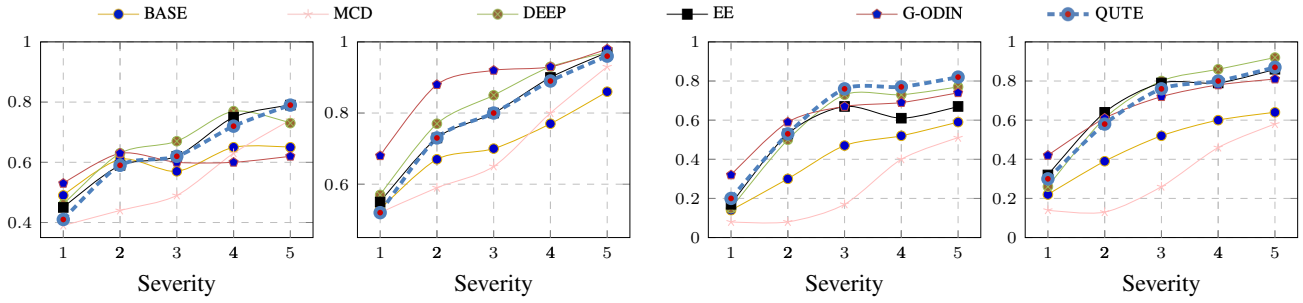
6.2. CID detection

As described in Section 5.2, we simulate a realistic setup to detect accuracy drop events in a model due to CID inputs by monitoring for a drop in confidence. Figure 4 plots the AUPRC and the best F1 score achieved over all corruptions for CIFAR10-C and TinyImagenet-C across 5 severity levels. Additional results such as precision and recall for individual severity levels is reported in Appendix A.3. Table 2 reports results on MNIST-C.

MNIST-C As shown in Table 2, QUTE outperforms all baselines comfortably in detecting accuracy drop events, with EE-ensemble a close second. Since both methods leverage early-exits, it further shows the effectiveness of early-exits in detecting corruptions. Furthermore, due to reasons described in Section 6.1, MCD performs poorly due to reduced model capacity. G-ODIN is outperformed by all baselines except MCD. This is in line with previous findings that OOD detectors find it hard to detect non-semantic shifts in data (Hsu et al., 2020).

CIFAR10-C Figures a) and b) in Fig 4 show that DEEP outdoes all baselines consistently on CIFAR10-C, followed by EE-ensemble and QUTE. For sev=1 and sev=2, all methods perform similarly because most corruptions with severity levels 1 & 2 are not severe enough to cause accuracy drop. Therefore, we observe a steady increase in the number of accuracy-drop events only as severity rises. For sev=3 and up, QUTE, along with EE-ensemble, is better at detecting more true positives. However, for sev=5, the corruption is so severe for EE-ensemble that it becomes overconfident, which results in a 19.7% drop in its performance. This phenomenon is also seen in G-ODIN. In contrast, QUTE exhibits the best corruption detection capability at the highest severity level.

TinyImagenet-C QUTE has the best AUPRC and F1 compared to all baselines for sev=3 and up (Figure 4). While MCD and EE-ensemble both surpass other baselines on uncertainty metrics (Table 1), QUTE outperforms both comfortably on CID detection. Further investigations revealed that high confidence-calibration in MCD and EE-ensemble



(a) AUPRC CIFAR10-C (b) F1 CIFAR10-C (c) AUPRC TinyImagenet-C (d) F1 TinyImagenet-C

Figure 4: CID detection results on CIFAR10-C and TinyImagenet-C. AUPRC and F1 is reported for all evaluated baselines for severity levels 1-5 (*higher is better*). Detailed results are available in Appendix A.3

ID / OOD	AUPRC	AUROC	F1	Precision/ Recall
MNIST/ Fashion MNIST				
- BASE	0.46	0.45	0.66	0.49 / 0.98
- MCD	0.44	0.4	0.66	0.49 / 0.99
- DEEP	0.71	0.74	0.69	0.63 / 0.77
- EE-Ensemble	0.72	0.81	**0.8**	**0.73 / 0.89**
- G-ODIN	0.38	0.31	0.66	0.5/ 1
- G-ODIN+	0.62	0.65	0.79	0.74 / 0.84
- QUTE	**0.81**	**0.85**	0.79	0.74 / 0.84
CIFAR10 / SVHN				
- BASE	0.91	0.85	0.92	0.97/ 0.97
- MCD	0.91	0.84	0.83	0.8/ 0.96
- DEEP	0.93	0.89	0.88	0.88/ 0.97
- EE-Ensemble	0.9	0.85	0.86	0.86/ 0.97
- G-ODIN	**0.97**	**0.95**	**0.92**	**0.92/ 0.96**
- QUTE	0.93	0.87	0.87	0.87/ 0.96

Table 3: OOD detection results on MNIST / FashionMNIST and CIFAR10 / SVHN

is in fact detrimental to their ability to detect accuracy drops. For example, for MCD with an accuracy of 0.34 on ID, the median confidence value of all predictions on ID is 0.345. However, on corrupted-ID, the overparameterized nature of modern neural networks does not allow the median confidence of MCD to drop any further than 0.27 i.e., a drop of 0.075, which is just not discernible enough to detect drop in accuracy. In contrast, QUTE's median confidence on ID is 0.69 and the average median confidence on CID is 0.57, which helps create a much clearer distinction between ID and CID. Furthermore, QUTE shows its effectiveness yet again in detecting coruptions, surpassing even DEEP and G-ODIN, especially for high severity levels.

6.3. OOD detection

As described in Section 5.2, we simulate an OOD rejection scenario by classifying a input sample as OOD if the confidence on that sample is below a certain threshold. Table 3 reports the evaluation metrics for the same using MNIST and CIFAR10 as ID and FashionMNIST and SVHN as OOD respectively. Surprisingly, QUTE and EE-ensemble outperform G-ODIN in addition to all baselines on FashionMNIST, which demonstrates the advantage of early-exit knowledge in OOD detection as well. Further analysis revealed that G-ODIN is under-confident on MNIST ID and over-confident on OOD, which results in a poor performance. As an ablation study, we increased the number of training epochs of G-ODIN from 20 to 50 epochs to improve its confidence. We denote this as G-ODIN+ in Table 3. Interestingly, G-ODIN+ achieves a significant 63% improvement in performance compared to G-ODIN. This is a notable observation for OOD detectors in tinyML, and perhaps suggests the need to rethink OOD detection with extremely tiny models. In contrast, G-ODIN outperforms all baselines on OOD detection of SVHN comfortably. Nonetheless, QUTE comes in a close second to G-ODIN along with DEEP and EE-ensemble. These results show the efficacy of QUTE in detecting both CID and OOD data.

6.4. Effectiveness of EV-assistance method

Model	F1 (↑)		BS (↓)		NLL (↓)	
	EV	no-EV	EV	no-EV	EV	no-EV
CIFAR10	0.858	0.858	**0.0205**	0.0206	**0.43**	0.46
TinyImagenet	**0.38**	0.35	**0.0043**	0.0046	**3.81**	4.74

Table 4: Effectiveness of *early-view* (EV) assistance. Results are averaged over 3 independent training runs.

We investigate the effect of the early-view (EV) assistance method on ensemble quality. To that end, we train non-EV versions of QUTE i.e., without the weight transfer from early-exits. Table 4 reports the calibration metrics for both EV and non-EV versions for CIFAR10 and TinyImagenet. As shown, EV-assistance has a massive influence on NLL in both cases, and improves the network accuracy on TinyImagenet by 8%. Furthermore, we find that the ability of non-EV versions to detect accuracy-drop events/CID detection reduces by 15% on average compared to EV-versions. These results show that the notably lower NLL obtained with EV-assistance (9% lower on CIFAR10 and 24% lower on TinyImagenet) not only improves uncertainty estimates, but it also proves crucial in both CID and OOD detection.

KWS/DS-CNN	Params (K)	F1↑	BS↓	NLL↓
- BASE	24.8	0.93	0.0082	0.184
- EE-Ensemble	87.1	0.94	0.0079	0.178
- QUTE	**28.8**	**0.94**	**0.0077**	**0.174**

Table 5: Calibration metrics for KWS

ID / OOD	AUPRC	F1
KWS-ID/KWS-OOD		
- BASE	0.82	0.82
- EE-Ensemble	**0.85**	0.82
- G-ODIN	0.67	0.72
- G-ODIN+	0.68	0.70
- QUTE	0.82	0.81

Table 6: OOD detection results on KWS.

6.5. Results for Keyword spotting

To demonstrate QUTE's general applicability, we evaluate it on keyword spotting (KWS) task. Table 5 shows the uncertainty metrics and Table 6 reports the OOD detection results respectively. All baselines are trained for 10 epochs. As seen, QUTE outperforms EE-ensemble on uncertainty metrics while using **3× fewer model parameters**. Further, on OOD detection, G-ODIN yet again performs poorly compared to other baselines. Like in the case of MNIST, we train G-ODIN+ for a higher amount of epochs i.e., 30 epochs. This yields a minimal improvement compared to G-ODIN but it is still suboptimal compared to QUTE and EE-ensemble. While the results with G-ODIN are not conclusive enough to draw definitive conclusions about the suitability of popular OOD detectors for extremely-tiny models, it shows an interesting trend.

7. Conclusions

Uncertainty quantification is crucial in tinyML models but many prior works are not optimized for tinyML resource constraints. We propose a novel resource-efficient ensemble architecture for uncertainty estimation that is optimized for tinyML. At the core of our methodology is our finding that model overconfidence decreases with its size. To that end, we leverage the better uncertainty estimation quality of early-exits by injecting early-exit weights into the multiple lightweight classification heads created at the output. QUTE provides better and reliable uncertainty estimates in a single forward pass with **3.1× lower model size** and **3.8× fewer FLOPS** compared to the most relevant prior work.

References

Antoran, J., Allingham, J., and Hernández-Lobato, J. M. Depth uncertainty in neural networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 10620–10634. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/781877bda0783aac5f1cf765c128b437-Paper.pdf.

Bai, H., Canal, G., Du, X., Kwon, J., Nowak, R., and Li, Y. Feed two birds with one scone: Exploiting wild data for both out-of-distribution generalization and detection, 2023.

Banbury, C. R., Reddi, V. J., Torelli, P., Holleman, J., Jeffries, N., Király, C., Montino, P., Kanter, D., Ahmed, S., Pau, D., Thakker, U., Torrini, A., Warden, P., Cordaro, J., Guglielmo, G. D., Duarte, J. M., Gibellini, S., Parekh, V., Tran, H., Tran, N., Niu, W., and Xu, X. Mlperf tiny benchmark. *CoRR*, abs/2106.07597, 2021. URL https://arxiv.org/abs/2106.07597.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural network. In *International conference on machine learning*, pp. 1613–1622. PMLR, 2015.

Bonato, V. and Bouganis, C.-S. Class-specific early exit design methodology for convolutional neural networks. *Applied Soft Computing*, 107:107316, 2021. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2021.107316. URL https://www.sciencedirect.com/science/article/pii/S1568494621002398.

Brier, G. W. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Ferianc, M. and Rodrigues, M. Mimmo: Multi-input massive multi-output neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 4563–4568, June 2023.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Ghanathe, N. P. and Wilton, S. T-recx: Tiny-resource efficient convolutional neural networks with early-exit. In *Proceedings of the 20th ACM International Conference on Computing Frontiers*, pp. 123–133, 2023.

Gneiting, T. and Raftery, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Havasi, M., Jenatton, R., Fort, S., Liu, J. Z., Snoek, J., Lakshminarayanan, B., Dai, A. M., and Tran, D. Training independent subnetworks for robust prediction, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Hendrycks, D. and Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks, 2018.

Hernández-Lobato, J. M. and Adams, R. P. Probabilistic backpropagation for scalable learning of bayesian neural networks, 2015.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Hsu, Y.-C., Shen, Y., Jin, H., and Kira, Z. Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10951–10960, 2020.

Huang, G., Chen, D., Li, T., Wu, F., van der Maaten, L., and Weinberger, K. Q. Multi-Scale Dense Networks for Resource Efficient Image Classification. *arXiv e-prints*, art. arXiv:1703.09844, March 2017.

Katz-Samuels, J., Nakhleh, J., Nowak, R., and Li, Y. Training ood detectors in their natural habitats, 2022.

Kaya, Y., Hong, S., and Dumitras, T. Shallow-deep networks: Understanding and mitigating network overthinking. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3301–3310. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/kaya19a.html.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6405–6416, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Le, Y. and Yang, X. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Liang, S., Li, Y., and Srikant, R. Enhancing the reliability of out-of-distribution image detection in neural networks, 2020.

Liu, W., Wang, X., Owens, J. D., and Li, Y. Energy-based out-of-distribution detection, 2021.

Mu, N. and Gilmer, J. Mnist-c: A robustness benchmark for computer vision, 2019.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift, 2019.

Qendro, L., Campbell, A., Lio, P., and Mascolo, C. Early exit ensembles for uncertainty quantification. In *Machine Learning for Health*, pp. 181–195. PMLR, 2021.

Rahaman, R. et al. Uncertainty quantification and deep ensembles. *Advances in Neural Information Processing Systems*, 34:20063–20075, 2021.

Tang, X., Yang, K., Wang, H., Wu, J., Qin, Y., Yu, W., and Cao, D. Prediction-uncertainty-aware decision-making for autonomous vehicles. *IEEE Transactions on Intelligent Vehicles*, 7(4):849–862, 2022. doi: 10.1109/TIV.2022.3188662.

Teerapittayanon, S., McDanel, B., and Kung, H. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2464–2469, 2016. doi: 10.1109/ICPR.2016.7900006.

Teye, M., Azizpour, H., and Smith, K. Bayesian uncertainty estimation for batch normalized deep networks, 2018.

Vargas, J., Alsweiss, S., Toker, O., Razdan, R., and Santos, J. An overview of autonomous vehicles sensors and their vulnerability to weather conditions. *Sensors*, 21(16):5397, 2021.

Warden, P. Speech commands: A dataset for limited-vocabulary speech recognition. *CoRR*, abs/1804.03209, 2018. URL http://arxiv.org/abs/1804.03209.

Wenzel, F., Snoek, J., Tran, D., and Jenatton, R. Hyperparameter ensembles for robustness and uncertainty quantification. *Advances in Neural Information Processing Systems*, 33:6514–6527, 2020.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Yang, J., Zhou, K., Li, Y., and Liu, Z. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021.

Yang, J., Wang, P., Zou, D., Zhou, Z., Ding, K., Peng, W., Wang, H., Chen, G., Li, B., Sun, Y., Du, X., Zhou, K., Zhang, W., Hendrycks, D., Li, Y., and Liu, Z. Openood: Benchmarking generalized out-of-distribution detection, 2022.

Zaidi, S. and Zela, A. Neural ensemble search for performant and calibrated predictions. 2020.

Zhang, X.-Y., Xie, G.-S., Li, X., Mei, T., and Liu, C.-L. A survey on learning to reject. *Proceedings of the IEEE*, 111 (2):185–215, 2023. doi: 10.1109/JPROC.2023.3238024.

Zhang, Y., Suda, N., Lai, L., and Chandra, V. Hello edge: Keyword spotting on microcontrollers. *CoRR*, abs/1711.07128, 2017. URL http://arxiv.org/abs/1711.07128.

A. Appendix

A.1. Training and dataset details

In this section, we describe the details of the models evaluated and specifics of training and the experimental setup.

A.1.1. Datasets

In our evaluations, we use three in-distribution datasets for training all baselines methods we evaluate: 1) MNIST (LeCun et al., 1998), 2) CIFAR10 (Krizhevsky, 2009) and 3) TinyImagent (Le & Yang, 2015).

MNIST MNIST is a dataset of handwritten digits containing 60,000 grayscale images of size 28×28 and a test set of 10,000 images. MNIST contains 10 classes

CIFAR10 CIFAR10 dataset consists of 60,000 32×32 rgb images out of which 10,000 images are in the test set. It contains 10 classes and thus 6000 images per class.

TinyImagenet TinyImagenet is a smaller version of the Imagenet (Deng et al., 2009) dataset containing 200 classes instead of 1000 classes of the original Imagenet. Each class in TinyImagenet has 500 images in the train set and the validation set contains 50 images per class. The size of the images are resized and fixed at 64×64×3.

For corrupted datasets, we use the corrupted versions of ID listed above:1) MNIST-C (Mu & Gilmer, 2019), 2) CIFAR10-C and 3) TinyImagenet-C (Hendrycks & Dietterich, 2019). All corruptions are drawn from 4 major sources: noise, blur, weather and digital.

MNIST-C The MNIST-C dataset contains 15 corrupted versions of MNIST - *shot_noise, impulse_noise, glass_blur, fog, spatter, dotted_line, zigzag, canny_edges, motion_blur, shear, scale, rotate, brightness, translate, stripe, identity*. All the corruptions are of a fixed severity level.

CIFAR10-C CIFAR10-C includes 19 different types of corruptions with 5 severity levels which gives us 19×5=95 corrupted versions of CIFAR-10 ID. We use all 95 corrupted versions in our experiments. The list of corruptions are: *gaussian_noise, brightness, contrast, defocus_blur, elastic, fog, frost, frosted_glass_blur, gaussian_blur, impulse_noise, jpeg_compression, motion_blur, pixelate, saturate, shot_noise, snow, spatter, speckle_noise, zoom_blur*.

TinyImagenet-C TinyImagenet-C includes 15 different types of corruptions with 5 severity levels which gives us 15×5=75 corrupted versions of ID. We use all 75 corrupted versions in our experiments. The list of corruptions are: *gaussian_noise, brightness, contrast, defocus_blur, elastic_transform, fog, frost, glass_blur, impulse_noise, jpeg_compression, motion_blur, pixelate, shot_noise, snow, zoom_blur*.

A.1.2. Training details

We evaluate three models in our experiments:1) 4-layer CNN on MNIST, 2) Resnet-8 with 3 residual stacks from the MLPerf Tiny benchmark suite (Banbury et al., 2021) on CIFAR10 and 3) MobilenetV2 (Howard et al., 2017) on TinyImagnet.

The 4-layer CNN is trained for 20 epochs with a batch size of 256, the Resnet-8 model is trained for 200 epcohs with batch size of 32 and the MobilenetV2 model is trained for 200 epochs with a batch size of 128. All models are trained with Adam optimizer with momentum of 0.9 and an initial learning rate of 0.001. The learning rate is decayed by a factor of 0.99 every epoch. For models with QUTE architecture, we achieve the weight transfer from early-exits to all $\{f_{out}(.)\}_{k=1}^{K}$ using a callback that sets the weights of each f_{out_k} with weights copied from the corresponding assisting early-exit at the end of each training batch.

Weighting the loss at EV-exits As described in Section 4.1, we weight the loss at EV-exits with a higher factor w_{EV_k} such that $w_{EV_k} = w_{EV_{k-1}} + \delta$. This is done to further promote diversity across the EV-exits. We empirically set δ to be 0.5. To determine w_{EV_0}, we repeat training with w_{EV_0} set to $\{2, 3, 4, 5\}$. We find that for $w_{EV_0} > 3$, the NLL starts dropping steadily because the higher loss at EV-exits overshadow the losses of the early-exits. This diminishes the influence of early-exits at EV-exits, which is undesirable. Hence, we set $w_{EV_0} = 3$ in all our experiments.

Framework and Hardware details

- **Program:** Tensorflow v2.3
- **Compilation:** Python v3.7, Tensorflow v2.3
- **Run-time environment:** Ubuntu Linux with miniconda
- **Hardware:** GeForce RTX 2080 GPU for model training
- **Run-time state:** set by our scripts
- **How much disk space required (approximately)?:** $\sim$200GB
- **How much time is needed to prepare workflow?:** 30-40min
- **How much time is needed to complete experiments?:** Several hours. 4-layer CNN on MNIST takes a few minutes to traim whereas, Resnet-8 requires 2-3 hours on the GPU we use. MobilenetV2 requires around 8 hours to train on RTX 2080 GPU.
- **Publicly available?:** Yes, but currently withheld for review. Datasets are also publicly available.

A.2. Uncertainty quantification on corrupted-in-distribution data

A.2.1. Uncertainty quantification experiments

For experiments to quantify uncertainty on corrupted datasets in Section 6.1, we use all the corrupted versions of ID described in Appendix A.1 and report the average metrics over all datasets. For MNIST-C with a fixed severity level, we calculate the F1, BS and NLL on each of the 15 corrupted datasets and report the average. For CIFAR10-C and TinyImagenet-C, which have 19 and 15 different corruptions respectively with 5 severity levels each, we construct the corrupted datasets as follows. For each type of corruption, we randomly select p samples from each of the 5 severity levels. Next, we concatenate all these samples to create a new corrupted dataset of size $5 \times p$. p is selected such that $5 \times p$ = size of ID dataset. This process is repeated for all corruptions. The datasets thus obtained contain samples from all severity levels. For example, for CIFAR10-C which consists of 19 corruptions, this process yields 19 corrupted datasets. We report the average F1, BS and NLL obtained on these newly created corrupted datasets.

A.3. Corruption detection experiments

For experiments to detect accuracy drop/CID detection described in Section 6.2, we append the ID dataset with each corrupted dataset obtained from the methodology described above. For example, for experiments with CIFAR10-C, we obtain 19 ID+CID datasets that contains both ID samples and corrupted samples (from all severity levels). Next, as described in Section 5.2, for each prior work we evaluate, we first iterate over *only* ID and obtain predictions from the model while computing the moving average of accuracy of the past m predictions using a sliding-window. In this way, we obtain the accuracy distribution of the sliding-window $\mathcal{A}_{SW}$ on ID, and then compute its mean μ_{ID} and standard-deviation σ_{ID}. Next, we iterate over all ID+CID datasets while computing the moving average of confidence of the past m predictions using a sliding-window ($\mathcal{C}_{SW}$). At the same time, we also compute $\mathcal{A}_{SW}$. We record all instances of $\mathcal{C}_{SW}$ dropping below a certain threshold ρ. These events are denoted as CID-predicted events. Finally, the description of true positive (TP), false positives (FP) , false negatives (FN) and true negatives (TN) are as follows.

- True positive: $\mathcal{C}_{SW} < \rho$ and $\mathcal{A}_{SW} \leq \mu_{ID} - 3 \cdot \sigma_{ID}$
- False positive: $\mathcal{C}_{SW} < \rho$ and $\mathcal{A}_{SW} > \mu_{ID} - 3 \cdot \sigma_{ID}$
- True negative: $\mathcal{C}_{SW} > \rho$ and $\mathcal{A}_{SW} > \mu_{ID} - 3 \cdot \sigma_{ID}$
- False negative: $\mathcal{C}_{SW} > \rho$ and $\mathcal{A}_{SW} \leq \mu_{ID} - 3 \cdot \sigma_{ID}$

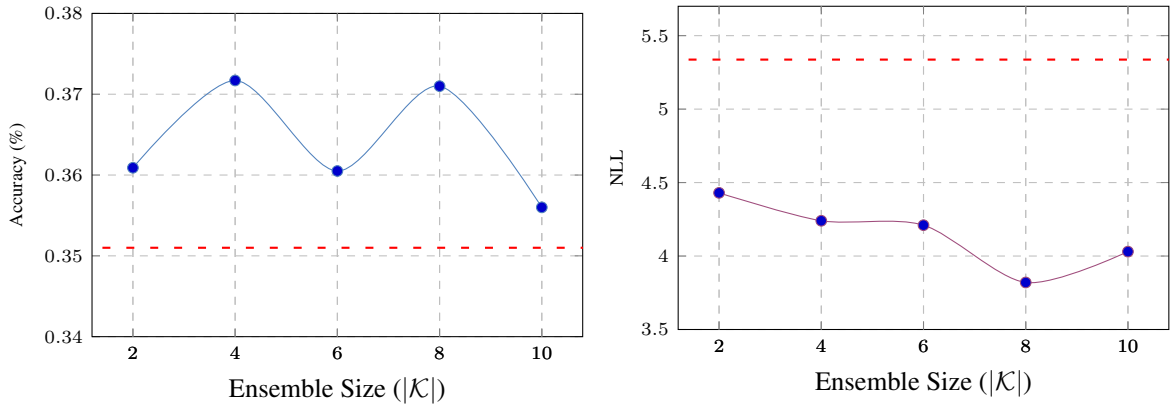
In this way, we collect the TP, FP, TN, FN from each ID+CID datasets and compute AUPRC, F1, precision and recall. Finally, we average these metrics over all ID+CID datasets and report them. Tables 7 and 8 report the AUPRC, highest F1 achieved, and the precision and recall associated with the highest F1 for all severity levels on CIFAR10-C and TinyImagnet-C.

Model	AUPRC / F1 (↑)				
	Severity				
	1	**2**	**3**	**4**	**5**
CIFAR10					
- BASE	0.49 / 0.53	0.61 / 0.67	0.57 / 0.7	0.65 / 0.77	0.65 / 0.86
- MCD	0.39 / 0.52	0.44 / 0.59	0.49 / 0.65	0.63 / 0.8	0.74 / 0.93
- DEEP	0.46 / 0.57	**0.63 / 0.77**	**0.67 / 0.85**	0.77 / 0.93	0.73 / 0.97
- EE-Ensemble	0.45 / 0.55	0.59 / 0.73	0.62 / 0.8	0.75 / 0.9	**0.79 / 0.97**
- G-ODIN	**0.53 / 0.68**	0.63 / 0.88	0.6 / 0.92	0.6 / 0.93	0.62 / 0.98
- QUTE	0.41 / 0.52	0.59 / 0.73	0.62 / 0.8	0.72 / 0.89	**0.79 / 0.96**
TinyImagenet					
- BASE	0.14 / 0.22	0.3 / 0.39	0.47 / 0.52	0.52 / 0.6	0.59 / 0.64
- MCD	0.08 / 0.14	0.08 / 0.13	0.17 / 0.26	0.4 / 0.46	0.51 / 0.58
- DEEP	0.15 / 0.26	0.5 / 0.61	0.73 / 0.8	0.73 / 0.86	0.77 / 0.92
- EE-Ensemble	0.17 / 0.32	0.53 / 0.64	0.67 / 0.79	0.61 / 0.79	0.67 / 0.86
- G-ODIN	**0.32 / 0.42**	**0.59 / 0.61**	0.67 / 0.72	0.69 / 0.78	0.74 / 0.81
- QUTE	0.2 / 0.3	0.53 / 0.58	**0.76 / 0.76**	**0.77 / 0.8**	**0.82 / 0.87**

Table 7: CID detection results for CIFAR10 and TinyImagenet. Area under the precision recall curve (AUPRC), the highest F1 score achieved is reported. We repeat the experiment using the corrupted versions of the ID dataset for all severity levels. For all metrics, *higher is better*

A.4. Uncertainty quantification vs Ensemble size

In this section, we conduct an ablation study to study the effect of number of early-exits on uncertainty estimation quality. The ensemble size $|\mathcal{K}|$ is an hyperparameter that depends on the computation/resource budget, and it is bounded above by the depth of the base network. We vary the ensemble size and investigate its effect on calibration quality in MobilenetV2. Figure 5 shows the effect on accuracy and NLL in MobilnetV2 for $|\mathcal{K}|$ ranging from 2-10. The red line shows the accuracy



(a) Accuracy (↑)

(b) NLL (↓)

Figure 5: Effect of Ensemble size on Accuracy and NLL in MobilenetV2 on TinyImagenet

and NLL of base network in respective plots. As shown, the accuracy does not always improve with ensemble size. This demonstrates that there is a limit to the improvement in accuracy due to early-exits. On the other hand, NLL steadily improves as ensemble size increases with the least NLL obtained for $|\mathcal{K}| = 8$. For $|\mathcal{K}| = 10$, the NLL rises slightly. Our investigations revealed that since the base network, early-exits and the early-view (EV) exits are trained simultaneously, the disruption in base network's weights reaches a tipping-point after a certain ensemble size thereby, causing a drop off in performance.

Model	Precision / Recall (↑)				
	Severity				
	1	**2**	**3**	**4**	**5**
CIFAR10					
- BASE	0.52 / 0.54	0.77 / 0.59	0.77 / 0.65	0.83 / 0.72	0.93 / 0.81
- MCD	0.54 / 0.5	0.59 / 0.59	0.72 / 0.59	0.84 / 0.76	0.96 / 0.9
- DEEP	0.54 / 0.59	0.91 / 0.67	0.93 / 0.77	0.98 / 0.88	0.99 / 0.95
- EE-Ensemble	0.56 / 0.53	0.82 / 0.67	0.96 / 0.69	**0.99 / 0.83**	**0.99 / 0.94**
- G-ODIN	**0.62 / 0.76**	**0.98 / 0.8**	**0.97 / 0.87**	0.99 / 0.88	0.99 / 0.96
- QUTE	0.57 / 0.47	0.85 / 0.64	0.88 / 0.74	0.94 / 0.84	**0.99 / 0.94**
TinyImagenet					
- BASE	0.19 / 0.27	0.32 / 0.49	0.47 / 0.58	0.52 / 0.7	0.67 / 0.61
- MCD	0.07 / 1	0.08 / 0.4	0.17 / 0.48	0.44 / 0.49	0.55 / 0.6
- DEEP	0.18 / 0.49	0.52 / 0.75	0.73 / 0.88	0.85 / 0.88	**0.91 / 0.93**
- EE-Ensemble	0.24 / 0.47	**0.62 / 0.67**	**0.76 / 0.81**	0.84 / 0.75	0.89 / 0.83
- G-ODIN	**0.38 / 0.46**	0.59 / 0.62	0.72 / 0.72	0.76 / 0.79	0.79 / 0.84
- QUTE	0.22 / 0.51	0.49 / 0.72	0.69 / 0.85	**0.86 / 0.75**	**0.9 / 0.85**

Table 8: CID detection results for CIFAR10 and TinyImagenet. Precision and Recall associated with the highest F1 obtained (see Table 7) is reported. We repeat the experiment using the corrupted versions of the ID dataset for all severity levels. For all metrics, *higher is better*

A.5. Evaluation metrics

We evaluate uncertainty using the following metrics.

Brier Score (lower is better): It is a proper scoring rule that measures the accuracy of predictive probabilities. Incorrect predictions with high predictive confidence are penalized less by BS. Thus, BS is less sensitive to corruptions and incorrect predictions. Therefore, NLL is a better measure of uncertainty to compare with other methods.

$$BS = \frac{1}{N} \sum_{l \in \{1,2,..L\}} (p_\Theta(y = l|x) - \mathbb{1}(y = l))^2 \tag{5}$$

where, $\mathbb{1}$ is an identity function and L is the number of classes.

Negative log-likelihood (lower is better): It is a proper scoring rule that measures how probable it is that the predictions obtained are from the in-distribution set. It depends on both the uncertainty (predictive confidence) and the accuracy of the predictions.

$$NLL = - \sum_{l \in \{1,2,..L\}} \mathbb{1}(y = l) \cdot log p_\Theta(y = l|x) + (1 - \mathbb{1}(y = l)) \cdot \ log(1 - p(y = l|x)) \tag{6}$$