Flexible Bayesian Inference by Weight Transfer for Robust Deep Neural Networks

Thi Thu Thao KHONGa, Nonmember, Takashi NAKADA, Member, and Yasuhiro NAKASHIMA, Fellow

SUMMARY Adversarial attacks are viewed as a danger to Deep Neural Networks (DNNs), which reveal a weakness of deep learning models in security-critical applications. Recent findings have been presented adversarial training as an outstanding defense method against adversaries. Nonetheless, adversarial training is a challenge with respect to big datasets and large networks. It is believed that, unless making DNN architectures larger, DNNs would be hard to strengthen the robustness to adversarial examples. In order to avoid iteratively adversarial training, our algorithm is Bayes without Bayesian Learning (BwoBL) that performs the ensemble inference to improve the robustness. As an application of transfer learning, we used learned parameters of pretrained DNNs to build Bayesian Neural Networks (BNNs) and focus on Bayesian inference without cost-learning. In comparison with no adversarial training, our method is more robust than activation functions designed to enhance adversarial robustness. Moreover, BwoBL can easily integrate into any pre-trained DNN, not only Convolutional Neural Networks (CNNs) but also other DNNs, such as Self-Attention Networks (SANs) that outperform convolutional counterparts. BwoBL is also convenient to apply to scaling networks, e.g., ResNet and EfficientNet, with better performance. Especially, our algorithm employs a variety of DNN architectures to construct BNNs against a diversity of adversarial attacks on a large-scale dataset. In particular, under L∞ norm PGD attack of pixel perturbation ε = 4/255 with 100 iterations on ImageNet, our proposal in ResNets, SANs, and EfficientNets increase by 58.18% top-5 accuracy on average, which are combined with naturally pretrained ResNets, SANs, and EfficientNets. This enhancement is 62.26% on average below L2 norm C&W attack. The combination of our proposed method with pretrained EfficientNets on both natural and adversarial images (EfficientNet-ADV) drastically boosts the robustness resisting PGD and C&W attacks without additional training. Our EfficientNet-ADV-B7 achieves the cutting-edge top-5 accuracy, which is 92.14% and 94.20% on adversarial ImageNet generated by powerful PGD and C&W attacks, respectively.

key words: Deep Neural Network, Bayesian Neural Network, image classification, adversarial attacks, adversarial training

1. Introduction

Deep neural networks (DNNs) have been shown an admirable performance in various tasks, such as computer vision, a field of recognition, etc. Despite their breakthroughs, DNNs can be vulnerable to adversarial attacks [1]–[4], where adversarial examples are crafted by adding imperceptible perturbations to original images, as shown in Fig. 1. The lack of robustness for DNNs with respect to adversarial instances exposes security threats on the real systems of safety-critical applications.

Several attack methods have recently been developed to generate adversarial images, i.e., Fast Gradient Sign Method (FGSM) [2], Projected Gradient Descent (PGD) [5], and C&W attack [4], which easily fool a network even when its accuracy is high. These algorithms are designed based on a gradient of the loss function to minimize the perturbation. They thus make adversarial images hard to distinguish from natural images. There are two types of assumptions of the attacker’s understanding: white-box [2], [4] and black-box attacks [6]. Furthermore, based on a goal of misclassification, we have two other kinds of attacks, including targeted and untargeted attacks. The fact that defending a DNN from a white-box and untargeted attack is more challenging than the defense from a black-box and targeted counterpart. Therefore, our experiment focuses on white-box and untargeted settings on both PGD and C&W attacks.

Due to the vulnerability of DNNs, many effective defenses have been proposed, such as defensive distillation [7], adversarial detection [8], robust activation functions [9]–[13], and adversarial training [2], [4], [5]. Currently, adversarial training has been the most outstanding defense of CNNs. Moreover, several studies have recently shown the potential robustness of adversarial Bayesian learning [14]. Liu et al. introduced the Random Self-Ensemble method [15] by adding random components to CNNs and conducted adversarial training by Bayesian Neural Networks (BNNs) in Adv-BNN algorithm [16]. Nevertheless, these defensive methods have been implemented on small datasets, e.g., MNIST, CIFAR-10, and become more ambitious on large-scale datasets like ImageNet.

As a solution to Bayesian inference, we proposed a new defense algorithm called Bayes without Bayesian Learning (BwoBL) to apply BNNs against adversarial attacks, which was published in [17]. We consecutively expand BwoBL al-
algorithm on various networks resisting a variety of adversarial attacks. We choose pretrained CNNs that have achieved state-of-the-art accuracy on ImageNet, e.g., ResNets and EfficientNets. Besides, we also prove that our algorithm is applied to not only CNNs but also other pretrained DNNs, such as Self-Attention Networks (SANs). To the best of our knowledge, our approach is the first defense that utilizes Bayesian inference on the big dataset and large networks to improve the robustness of DNNs against adversaries but does not cost the training computation. Our contributions can be summarized below:

- We build BNNs based on learned parameters of prominent DNN architectures. The uncertainty of BNNs can resist gradient-based attacks and create an ensemble model to enhance the robustness of models.
- BwoBL algorithm is shown as a capable defense against various attacks that can be utilized for a wide range of DNNs and solve the overhead of the training problem.
- The robustness of our proposal is assessed on the ImageNet dataset with pretrained CNNs and SANs under strong $l_\infty$ norm PGD attack and $l_2$ norm C&W attack without additional training.

The remainder of this paper is organized as follows. Section 2 introduces related work in Bayesian Neural Networks, adversarial attacks, and defense methods. Section 3 shows the challenges of adversarial training and the opportunities for Bayesian inference. Section 4 describes the construction of BNNs based on pretrained DNNs and Bayesian inference using BwoBL algorithm. Sections 5 and 6 demonstrate experimental setup and results. The conclusion is indicated in Sect. 7.

2. Related Work

2.1 BNN with Gaussian Approximation

BNNs are a robust form of DNNs with a valuable property of the uncertainty estimation, which can solve an overconfident decision of DNNs. For example, DNN-based solutions for diagnostic applications in medicine have widely been developed without any risk management. Meanwhile, the reliable information on automated decisions is a key requirement for DNNs. Hence, the integration of Bayesian ideas and DNNs is a principle for estimating the uncertainty of models.

In general, a BNN is built by replacing the weights of DNN that are single point estimates with probabilistic distributions, as seen in Fig. 2. The application of Bayesian methods for DNNs is finding the posterior distribution of the weights $w$ given the observed data $D$ [18]–[22]. Based on Bayes theorem, we have the posterior distribution of the weights as

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)}$$

(1)

where the denominator $p(D)$ is the evidence and given by:

$$p(D) = \int p(D|w)p(w)dw$$

(2)

However, this integration of the prior distribution $p(w)$ and the likelihood $p(D|w)$ is intractable. The approximation to the posterior distribution is thus used, and a variational inference has been proposed by Blundell et al. [20], which is called Bayes by Backprop. In this approach, the posterior distribution is approximated by finding the optimal approximating distribution $q(w|D)$ that minimizes the Kullback-Leibler (KL) divergence with the true posterior $p(w|D)$. Additionally, Monte Carlo sampling technique has been applied to draw the weight samples from the variational posterior $q(w|D)$. Blundell et al. supposed that the variational posterior was a Gaussian distribution $q(w|D) = N(\mu, \sigma^2)$. Model parameters can be obtained by sampling a unit Gauss $\xi \sim N(0, I)$, then shifting by a mean $\mu$ and scaling by a standard deviation $\sigma$. A posterior sample of the model parameter is produced as follows:

$$w = \mu + \sigma \odot \xi$$

(3)

where $\odot$ is point-wise multiplication. From this approximation, we face the generation of representative samples of the weights that is generally not easy in the learning process. Both the mean and the standard deviation are learned, which makes the learning of BNNs bulky.

2.2 Adversarial Attacks

Adversarial instances have been pointed out by Szegedy et al. firstly [1]. We assume that $x$ is an original input with a correct label $y$. Attack methods are able to seek an adversarial input $x'$ that is classified by a label $y' \neq y$. $x$ and $x'$ are close according to $l_p$ distance metric $\| \cdot \|_p$ noted as:

$$\|x - x'\|_p = \left( \sum_{i=1}^{n} |x_i - x'_i|^p \right)^{\frac{1}{p}}$$

(4)

With this attribute, $x'$ is called an untargeted adversarial example [4]. In this paper, an adversary is assumed to access the architecture and the parameters of a network, then it is used in the white-box manner [23] of baseline models.

Fig. 2 A comparison between a DNN and a BNN and an inner structure of a neuron that show the replacement of fixed values with the probabilities on the weights $w_{ij}$. Left: a DNN with specific values of the weights. Middle: a BNN with the distribution function of the weights. Right: an inner structure of a neuron with the weights $w_{ij}$ that are fixed values in a DNN replaced by probabilistic distributions in a BNN.
In recent years, several attack approaches have been introduced [2, 4, 5, 24, 25] that craft adversarial examples by using a gradient of the objective function in regard to the input data and adjusting to maximize the loss. FGSM is a single-step attack algorithm, which perturbs the original input by the direction of the gradient of the loss function \( J(\theta, x, y) \). \( \theta \) is the parameters of a model and a perturbation \( \epsilon \) is added to each pixel to control \( l_\infty \) norm \( \| x - x' \|_\infty \leq \epsilon \) [2], as follows:

\[
x' = x + \epsilon \text{ sign}(\nabla_x J(x, x', y))
\]  

(5)

Madry et al. [5] proposed a PGD attack that is a multi-step variant of FGSM. The PGD algorithm implements a strong iterative attack to generate adversarial instances, which follows the update as:

\[
x^{t+1} = \prod_{\epsilon} (x' + \beta \text{ sign}(\nabla_x J(x, x', y)))
\]  

(6)

in which \( \beta \) is an attack step size, and \( \prod_{\epsilon} \) is a projection to \( l_\infty \) norm adversary. Due to its power, we mainly utilize the PGD method to yield perturbed images in our experiment.

In addition, Carlini and Wagner introduced C&W approach [4] that constructs adversarial examples \( x' = \frac{1}{2} (\tanh(w) + 1) \) by searching for \( w \) in \( l_2 \) distance to solve the optimization problem:

\[
\min \frac{1}{2} \left( \tanh(w) + 1 \right) - x_2^2 + c \cdot f \left( \frac{1}{2} \left( \tanh(w) + 1 \right) \right)
\]  

(7)

where \( c \) is a constant that is chosen by the modified binary search and \( f(\cdot) \) is an objective function defined as:

\[
f(x') = \max \left( \max (Z(x'_i) : i \neq t) - Z(x'_i), -\kappa \right)
\]  

(8)

The parameter \( \kappa \) helps us find an adversarial example with high confidence. \( Z(\cdot) \) is the output of the network. C&W algorithm is also a powerful attack, which is the second attacker in our experiment.

2.3 Defense Methods

There are a large number of defense algorithms against adversarial attacks. [9]–[13], [26] showed the properties of activation functions could affect the robustness of DNNs and designed robust activation functions as defense methods. However, most of them are optimized by adversarial training. It is known that adversarial training is currently the most robust algorithm that trains DNNs on perturbed examples. Madry et al. [5] proposed adversarial training to improve the robustness of DNNs resisting PGD attacks on MNIST and CIFAR-10. Unfortunately, the high cost of generating adversarial instances makes this training unrealistic on large-scale datasets like ImageNet. Recently, many studies have applied PGD-based adversarial training to ImageNet [27]–[29]. “Free” adversarial training of Shafahi et al. [28] keeps the robustness of standard adversarial training but be faster. “Fast” adversarial training of Wong et al. [29] is noticeably better than “Free” counterpart. The trend of defense approaches must be both the robustness improvement and the overhead limitation of training problem.

The idea of adding stochastic components to DNNs and training these models on perturbed data has been perceived as a good defense to prevent strong gradient-based attacks. Liu et al. [16] indicated the dominance of BNNs in improving the robustness of DNNs to adversarial attacks on CIFAR-10 and ImageNet-143. However, adversarial Bayesian training will be harder on large-scale images like ImageNet. Therefore, we mainly focus on Bayesian inference with the proposed BwoBL algorithm to withstand strong adversarial attacks and avoid the additional adversarial training.

3. Challenges of Adversarial Training & Opportunities for Bayesian Inference

3.1 Challenges of Adversarial Training

Madry et al. [5] have proposed the natural saddle point (min-max) formulation that is optimized by adversarial training as follows.

\[
\min_{\theta} \max_{\epsilon \in S} \mathbb{E}_{(x, y) \sim D} \left[ J(\theta, x + \epsilon, y) \right]
\]  

(9)

where \( D \) is an underlying data distribution over the pairs of \( x \) examples and \( y \) corresponding labels, \( J(\cdot) \) is a suitable loss function, \( \theta \) is network parameters, and the per-pixel perturbation \( \epsilon \) is allowed in the perturbed range \( S \). This formula shows two computation steps of adversarial training: (1) an inner maximization, which takes adversarial instances, and (2) an outer minimization, which finds model parameters.

The robustness of adversarial training depends on the strength of the adversaries that are generated by the inner maximization in Eq. (9). There are two problems in the generation of adversarial examples for training. First, if \( \epsilon \) is fixed during training, the robustness of the network only resists an adversarial attack of that \( \epsilon \) value [30]. Second, the iteration of the gradient computation is required to produce the adversaries in PGD attack, which consumes the high cost of the computation time. Many FGSM adversarial training methods are designed to accelerate against PGD attacks. Nevertheless, the fast training on FGSM adversaries is only robust against non-iterative attacks but difficult to resist iterative counterparts like PGD adversaries. Besides, Wong et al. [29] have shown the lack of a diversity of adversarial examples generated by FGSM attack causes catastrophic overfitting in the training phase. Additionally, to achieve the outer minimization in Eq. (9), the training process must be performed in many epochs. The iteration of the inner maximization and the outer minimization make adversarial training challenging on realistic datasets.

3.2 Challenges of Adversarial Bayesian Training

In addition to the drawbacks of adversarial training,
Bayesian learning has some disadvantages that make BNNs hard to apply for big datasets and vast networks.

Firstly, the weights of BNNs are described by probabilistic distributions, e.g., Gaussian function. The number of parameters is thus double. If a network architecture is sizable, it shall have a huge number of parameters.

Secondly, since the functional form of BNNs does not enable it to exact the integral, we often take a variational approximation [20] to the posterior distribution on the weights. This makes the optimization problem much larger scale.

Thirdly, the uncertainty of the weights leads to a coherent variability of the training data during training. BNN learning hence trains an ensemble of the networks instead of a single network, which is called Ensemble Learning [21]. This is not tractable for BNNs of any practical size.

3.3 Opportunities for Bayesian Inference

In recent years, much research has proved that the ensemble model outperforms the single model, especially in adversarial attacks [15], [16], [31]. It is shown that ensemble inference is a crux of BNNs [21]. The uncertainty of the weights makes BNNs equivalent to the ensemble of random models but do not increase the number of models [31].

Liu et al. [16] performed an adversarial Bayesian training, which achieves the good performance of BNNs under strong PGD attacks on CIFAR-10. Nonetheless, the challenges of adversarial Bayesian learning make the training of BNNs difficult in real-world applications. To address this issue, we build a BNN model based on a pretrained DNN, which has already been the state-of-the-art performance in image classification. We apply transfer learning from DNNs to BNNs. The learned parameters of original DNNs are employed to execute Bayesian inference efficiently, but our method does not add any training phase.

We represent a preliminary evaluation to compare the robustness of defensive methods on CIFAR-10 under \( \ell_{\infty} \) PGD and \( l_2 \) C&W attacks in Tables 1 and 2. CIFAR-10 is a small dataset that is easy to implement in most defensive approaches. Table 1 verifies the construction of our BNN from pretrained CNN and executing Bayesian inference can boost the robustness of the pretrained model against adversarial attacks without the iterative adversarial training. Besides, our BNN is more robust than BNN trained on adversarial images (Adv-BNN). Table 2 demonstrates the robustness of our algorithm to adversarial attacks compared with robust activation functions when both methods do not perform adversarial training. [13] trained ResNet-20 with various activation functions on natural CIFAR-10 and proved the robustness of their SPLASH activation function. We primarily alter convolutional layers of pretrained ResNet-20 without changing the activation function and sharply improved the robustness to adversarial attacks. With these preliminary comparisons, the potential of our idea is revealed on the small dataset. The detailed evaluation of a realistic dataset will be implemented in the next section.

With the rapid deployment of deep learning, pretrained DNNs are universally utilized and importantly contribute to the research community. We take advantage of pretrained models to reinforce their resistance to adversarial attacks and avoid the overhead of the training phase. It should be emphasized that leveraging learned parameters of existing models is key in our algorithm.

4. Methodology

Based on pretrained DNNs, we build our BNNs that replace convolutional layers in CNNs and linear transformation layers in SANs with Bayes layers. In Bayes layers, instead of single point values, the parameters are the probabilistic distribution. Other layers in CNN and SAN architectures are originally kept. To overcome the drawbacks of Bayesian learning, we solve Bayesian inference via BwoBL algorithm to enhance the robustness of DNNs.

4.1 BNN Built on Pretrained CNN and SAN Architecture

As an application of Gaussian variational posterior [20] and Gaussian dropout [32], we approximate the variational posterior of our BNNs with a Gaussian distribution. Our BNNs have hence applied two convolutional operations to model parameters. The first convolution is the mean \( \nu \) of the variational distribution, which is treated as a single fixed estimate in the frequentist inference. The second is the standard deviation \( \sigma \), which controls the weight uncertainty. Particularly, the variational posterior distribution \( q(w) = N(\theta, \sigma^2) \) is accurately an approximation to implement a reparameterization in Bayes layers, which plays an essential role in the construction of our BNNs. The variance of \( w \) is tied by its magnitude. A larger weight is then valuable when it is robust to noise. From this perspective, we have a formula of

| Network            | Activation | \# of success attacks |
|--------------------|------------|-----------------------|
| ResNet-20 [13]     | ReLU       | 903 ± 11.8            |
|                    | Swish      | 911 ± 15.1            |
|                    | APL        | 894 ± 11.5            |
|                    | Tent       | 881 ± 11.1            |
|                    | SPLASH     | 870 ± 12.3            |
| Our proposed ResNet-20 | ReLU       | 665 ± 7.9             |

Table 1 Evaluation of top-1 accuracy to compare the robustness of adversarial training by CNN (FAST), adversarial learning by BCNN (Adv-BNN), and our proposal on CIFAR-10 with 10000 images of the validation set, under \( \ell_{\infty} \) attack with pixel perturbation (\( \epsilon = 8/255 \)) and 20 iterations.

| Method     | Network            | Top-1 (%) |
|------------|--------------------|-----------|
| FAST [29]  | PreactResNet-18 (CNN) | 46.77     |
| Adv-BNN [16]| VGG-16 (BCNN)       | 47.58     |
| Our proposal | pretrained PreactResNet-18 + proposal (BCNN) | 50.02     |

Table 2 A comparison of robust activation functions and our proposal on CIFAR-10 under \( l_2 \) C&W attack. We compare the results of SPLASH activation function [13] and our BNN model on 1000 images chosen from correctly classified images of ResNet-20. We also execute our experiment five times to calculate mean ± standard deviation of the number of success attacks as [13].
the weight in Bayes layers instead of Eq. (3) as below:

\[
\mathbf{w} = \theta + \alpha \theta \odot \xi
\]

where \(\xi\) is a Gaussian unit \(\mathcal{N}(0,I)\). We treat \(\theta\) as single fixed values of pretrained DNN parameters while \(\alpha = \sqrt{\sigma}\) is empirically determined by the robustness of the BNN model against the adversarial attack. Our aim is the use of pretrained DNNs to construct our BNNs and avoid Bayesian learning, which is a term of transfer learning. We mainly concentrate on adjusting the standard deviation \(\sigma\) to create uncertainty on the parameters of BNNs and conduct the stochastic inference of BNN models.

Generally, the parameter transformation that includes the weights and the biases of convolutional layers in CNNs and linear transformation layers in SANs is enough for building our BNN models, and it does not enlarge model parameters. Therefore, all convolutional layers in pretrained CNNs and linear transformation layers in pretrained SANs are replaced by Bayes layers in our BNN models with the sample of the parameters in Eq. (10). The replacement has been illustrated in Figs. 3 and 4. The others of original CNN and SAN architectures remain unchanged.

4.2 BNN Inference via BwoBL Algorithm against Adversarial Attacks

In this section, we introduce Bayes without Bayesian Learning algorithm applied for the BNN inference to improve the robustness of pretrained DNNs.

Let \(x\) be an original image and \(y\) be a corresponding label in a dataset \(D\). We use \(l_\infty\) norm PGD attack [5] and \(l_2\) norm C&W attack [8] to generate an adversarial image \(\hat{x}\) and employ pretrained DNN models with learned parameters \(\theta\) to construct BNNs. The inference process of BNN via BwoBL method is listed in Algorithm 1. From Eq. (10), we can see that a large \(\alpha\) leads to a big variance of \(\mathbf{w}\) that causes the uncertainty on model parameters. This uncertainty is beneficial in the resistance to gradient-based perturbation. In our experiment, \(\alpha\) depends on pretrained DNN architectures, it is thus called structural hyperparameter. This hyperparameter is chosen and fixed for each pretrained DNN before the execution of the inference phase. Based on the testing accuracy of our BNN model under adversarial attacks, we select fixed values of \(\alpha\) that are explained in Sect. 5.

In general, the solution to the parameters of the BNN model cannot be revealed by a closed-form expression because of the probabilistic variable \(\xi\sim\mathcal{N}(0,I)\). Consequently, Monte Carlo sampling is utilized to take one sample for each probabilistic variable by random number generation. If a pretrained DNN is \(f(w,x)\) with a parameter \(w\) and an input \(x\), our BNN can be denoted as \(f_\xi(w,x)\) that is an infinite number of stochastic models. Therefore, we can execute the probabilistic model multiple times, which is the ensemble inference procedure, to improve the accuracy of our models. To achieve the ensemble output, we apply the majority voting scheme that is the output with the most frequency presented in [17].

5. Experiment Setup

5.1 BNN Models and Dataset

We build up BNN models from the typical DNN architectures currently, which achieve state-of-the-art accuracy on ImageNet. This is the realistic dataset for the task of classifying 1000 classes with large-scale images. The fact that adversarial training on ImageNet considerably consumes the computation time and the training cost. Accordingly, there are only a few findings of adversarial training on ImageNet successfully.

Baseline CNN models used in our experiments are ResNet-50, ResNet-101, ResNet-152, ResNet-101-32x8d, EfficientNet-B0–B7 [33]–[36], in which ResNet-50 is the backbone network. ResNet-101 and ResNet-152 are re-
respectively 2x and 3x deeper than ResNet-50, and ResNet-101-32×8d is 2x deeper and 4x wider than ResNet-50. These models show that deeper and wider networks can further improve the performance in image recognition. Moreover, Tan et al. [34] argued that scaling up CNNs by their depth or width has not yet achieved the best accuracy and efficiency. They investigated a compound scaling method, which balanced three dimensions of networks, i.e., depth/width/resolution in EfficientNets. These are newly outstanding networks trained on ImageNet. We use all pre-trained EfficientNets on ImageNet [35], [36] to apply our BwoBL algorithm.

Our algorithm is integrated into not only CNNs but also other powerful DNNs, such as SANs. Recent work has shown effective architectures [37]–[41] that employ a self-attention mechanism as an alternative for convolutional networks in image classification. Zhao et al. [41] explored two types of SAN, including pairwise and patchwise self-attention. We replace all linear transformation layers in SANs with our Bayes layers and choose pretrained SAN19-pairwise and SAN19-patchwise, which are built on ResNet-50 architecture to apply our proposal. So as to validate our hypothesis, we compare the performance of BwoBL with adversarial training. Although we have mentioned robust activation functions in our preliminary evaluation, most of them are executed on small datasets and optimized through adversarial training to enhance their robustness further to strong attacks. Thus, adversarial training is still an outstanding defense. In particular, FAST [29] is the fastest method and maintains the robustness as previous adversarial training, which implements FGSM training against PGD attack on ImageNet with ResNet-50. We hence select this adversarial learning to combine our BwoBL and call them ResNet-50-FAST. Furthermore, Xie et al. [42] have executed adversarial training on ImageNet with EfficientNets, but their target is the enhancement of image recognition models. They treat perturbed examples as additional instances and propose an improved adversarial training scheme - AdvProp, which is trained on both natural and adversarial images. These networks are referred to as EfficientNet-ADV-Bx in our experiments.

All DNN architectures that we perform are summarized in Table 3. Our experiments are run on four GeForce GTX1080Ti and two GeForce RTX3090.

Table 3 State-of-the-art DNN architectures are used in our experiments. We apply BwoBL algorithm to pretrained DNNs on natural and adversarial ImageNet to construct our BNN models.

| Training data | Pretrained networks                  |
|---------------|-------------------------------------|
| Original ImageNet | ResNet-50  |
|                | ResNet-101   |
|                | ResNet-152   |
|                | ResNet-101-32×8d |
|                | EfficientNet-Bx |
|                | SAN19-pairwise |
|                | SAN19-patchwise |
| Adversarial ImageNet | ResNet-50-FAST |
|                | EfficientNet-ADV-Bx |

5.2 Adversarial Attacks

We set two forceful kinds of adversarial attacks that are PGD and C&W algorithm [43], to verify the robustness of our method.

From Eq. (6), we generate $l_\infty$ norm PGD attacks with the pixel-perturbed size $\epsilon = \{2/255, 4/255\}$, the step size $\beta = 1/255$, and the attack iterations $it = \{10, 50, 100\}$. It is known that increasing the number of attack iterations makes adversarial examples harder and is an obstruction of adversarial training.

With C&W attack, we compare the resistance of the models under the attack that is measured by $l_2$ norm of perturbed images, binary_search_steps = 4, max_iterations = 1000, initial_constant = 0.1, and confidence = 10. This establishment creates strong adversaries, which are enough to fool baseline networks.

Notice that both PGD and C&W are white-box and untargeted attacks. However, the white-box manner is set for pretrained networks in Table 3. Meanwhile, our BNNs are built on these pretrained models and utilize their learned parameters. It is then considered indirectly white-box settings.

5.3 Structural Hyperparameter

The crux of our BwoBL algorithm is structural hyperparameter $\alpha$ in Eq. (10), which adjusts the variance of Gaussian distribution on the parameters of Bayes layers. For each pretrained DNN in Table 3, we seek the best $\alpha$ so that the model achieves the highest accuracy under strong $l_\infty$ norm PGD attack of pixel perturbation $\epsilon = 4/255$, iteration $it = 10$.

We test the performance of our proposed networks in Table 3 with a wide range of $\alpha$ values, for example $\alpha = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$.

From Fig. 5, we realize the accuracy is a practically convex upward with the change of $\alpha$. The search space of $\alpha$ guarantees the optimal algorithm, and the best $\alpha$ corresponds to

![Fig. 5](image-url)
the peak of this convex upward. Figure 5 shows that the step size 0.1 of $\alpha$ makes a too big difference in the accuracy of the model, which is difficult to identify the peak exactly. We discover the step size 0.01 is good enough to find the peak of convex upwards. Especially, our ResNet-50 with $\alpha = 0.2$ and 0.3 get an approximate accuracy, but with $\alpha = \{0.21, 0.22, 0.23, 0.24, 0.25, 0.26, 0.27, 0.28, 0.29\}$, our proposed model continues increasing the accuracy, and we achieve its peak at $\alpha = 0.24$.

We hereby observe the effect of this factor in our proposed BNNs and treat it as a constant for each model, which is summed up in Table 4.

### 5.4 Ensemble Inference

The ensemble is a beneficial property of Bayesian inference by executing the stochastic model many times. As mentioned in Algorithm 1, we carry out several forward passes and take the ensemble of different output samples. We assess the majority voting output for our ensemble inference, which is stable and the best efficiency proved in [17].

Consequently, how many samples do we need to implement in the ensemble phase? In real-world applications, ensemble inference is a big challenge because of the longer execution time. A large quantity of samples brings a better performance of the networks but considerably consumes the inference time, especially on big datasets and deeper neural networks. Selecting a number of forward passes must be suitable to achieve high accuracy and reasonable inference time. Based on the achievement in Fig. 6, we are aware that the accuracy differential between the ensemble of 20, 30, 40, and 50 samples is not too great. We thus choose 20 that is tolerable for all our proposed models on ImageNet to obtain a trade-off between the accuracy and the computation cost.

It is apparent that the inference time of BNN is more massive than that of CNN due to the ensemble phase. However, Wan et al. [44] succeeded in the design of a FPGA-based hardware accelerator for BNNs, which executed the repeated inference of BNNs but still got $2.1 \sim 8.2\times$ speedup over the CNN accelerator. This proved that the ensemble inference of BNN could completely be spread to real-world applications to improve the performance of DNNs.

### 6. Evaluation

#### 6.1 Performance on Natural ImageNet

We assess the performance of all pretrained DNNs and our proposed models on natural ImageNet with 50000 images in the validation set. On large-scale ImageNet with 1000 classes, the top-5 evaluation might achieve the better prediction that we observe in our experiments. Our method mainly focuses on improving the robustness of classifiers via the uncertainty generation on the weights, which makes our BNNs robust to the adversaries rather than natural images. Therefore, applying BwoBL algorithm to pretrained DNNs affects the testing accuracy on natural examples, which is a 6.90% drop on average, as seen in Table 5. Many studies of adversarial training indicated that the trade-off between adversarial robustness and standard accuracy was inevitable. The fact that gaining robustness of the model would be losing accuracy on natural examples. We hence apply BwoBL algorithm to deeper and wider networks to yield comparable accuracies on natural images. Table 5 shows that scaling up networks enhances not only the accuracy of the original DNNs but also that of our proposed networks. In particular, the application of our proposal to scaling networks significantly boosts the accuracy and 95.51%, which is only a 1.72% drop, top-5 prediction of our proposed EfficientNet-ADV-B7 is the advanced accuracy on natural ImageNet among defense methods currently.

#### 6.2 Tolerance on Adversarial ImageNet

##### 6.2.1 PGD Attack

We create PGD attacks on the first 10000 images in the validation set of ImageNet and evaluate the tolerance of our BwoBL algorithm on these perturbed images.

We examine the robustness of our proposed models under $l_{\infty}$ norm PGD attack of pixel perturbation $\epsilon = \{2/255, 4/255\}$ and iterations $it = \{10, 50, 100\}$, as shown in Table 6. The combination of our BwoBL and pretrained

| Top-5 Accuracy [%] |
|-------------------|
| 1                |
| 2                |
| 3                |
| 4                |
| 5                |

### Table 4 Structural hyperparameter $\alpha$ that adjusts the variance of Gaussian distribution on parameters of Bayes layers fixed for our proposed BNNs.

| Architectures | $\alpha$ | Architectures | $\alpha$ |
|---------------|----------|---------------|----------|
| ResNet-50     | 0.24     | SAN19-pairwise| 0.10     |
| ResNet-101    | 0.25     | SAN19-patchwise| 0.15     |
| ResNet-152    | 0.26     | ResNet-50-FAST| 0.10     |
| ResNet-101-32x8d | 0.23 |               |          |
| EfficientNet-B0| 0.10     | EfficientNet-ADV-B0| 0.05     |
| EfficientNet-B1| 0.12     | EfficientNet-ADV-B1| 0.07     |
| EfficientNet-B2| 0.12     | EfficientNet-ADV-B2| 0.08     |
| EfficientNet-B3| 0.14     | EfficientNet-ADV-B3| 0.11     |
| EfficientNet-B4| 0.18     | EfficientNet-ADV-B4| 0.15     |
| EfficientNet-B5| 0.22     | EfficientNet-ADV-B5| 0.17     |
| EfficientNet-B6| 0.23     | EfficientNet-ADV-B6| 0.20     |
| EfficientNet-B7| 0.24     | EfficientNet-ADV-B7| 0.21     |

### Fig. 6 A number of forward passes in ensemble inference. The ensemble of 20 samples is chosen to attain a trade-off between the accuracy and the computation cost for all our proposed models. Our network: ResNet-50 + BwoBL. PGD attack: $l_{\infty}$ norm, $\epsilon = 4/255$, iteration = 10.
DNNs on natural images significantly enhances the robustness, which is 44.83% and 60.16% increases on average in ResNets and SANs, respectively. However, it is just a slight improvement when applying BwoBL to pretrained DNNs on adversarial images, i.e., ResNet-50-FAST, which rises by 3% with pixel perturbation $\epsilon = 4/255$.

SAN19 is built on ResNet-50 architecture, but its parameters and FLOPs are smaller and its accuracy is higher than ResNet-50 on natural images. The application of BwoBL algorithm to SAN19 sharply raises the robustness of naturally pretrained SAN19-pairwise and SAN19-patchwise. In which the top-5 accuracies of our SAN19-patchwise are 76.10% and 63.55% on 2-pixel and 4-pixel perturbed images, respectively, which outperform ResNet-50. This proves that BwoBL approach may be integrated into any DNN architecture to construct a robust network against adversaries.

Table 6 also demonstrates that scaling up ResNet by the depth and the width can considerably boost the resistance of our algorithm to adversarial attacks. For instance, under the $\epsilon = 4/255$ and it = 10 PGD attack, our ResNet-101-32x8d, which is built on naturally pretrained ResNet-101-32x8d, improves the robustness by 21.83% for our ResNet-50, 4.44% for our ResNet-50-FAST-4px, and 5.18% for our ResNet-50-FAST-2px. Furthermore, adversarial training must be iterated to achieve the highest accuracy when attack parameters change, such as ResNet-50-FAST-2px and ResNet-50-FAST-4px are ResNet-50 trained on 2-pixel and 4-pixel perturbed images. In contrast, our algorithm is conveniently applied to those models to resist forceful PGD attacks without an additional training phase.

Additionally, increasing the iteration generates stronger PGD attacks but it also raises the training cost. Thus, adversarial training usually uses single-step PGD as FGSM to lower the training cost. Nonetheless, the robustness of the models trained on FGSM examples that is ResNet-50-FAST-2px and ResNet-50-FAST-4px will decline when the number of PGD iterations rises, as seen in Table 6. Contrarily, our proposed models are more robust with powerful attacks.

EfficientNet is a strong family of DNNs for image classification, which determines the optimal factor for scaling networks. We observe the benefit of EfficientNet in our proposed algorithm with two pretrained EfficientNet schemes. Our EfficientNet is the connection between BwoBL method and pretrained EfficientNet on natural images. The proposed EfficientNet-ADV is applying BwoBL to pretrained EfficientNet on both natural and PGD adversarial images.

We evaluate the robustness of our BwoBL method on EfficientNets under $\ell_\infty$ PGD perturbation of pixels $\epsilon = (2/255, 4/255)$ with iterations it = (10, 50, 100). As shown in Table 6, pretrained EfficientNets on natural images are almost fooled by the attacks but our proposed EfficientNets remarkably boost the accuracy without the extra training. For example, the top-5 accuracy of EfficientNet-B7 rises from 0.02% to 81.41% below PGD attack of $\epsilon = 4/255$, it = 100. Furthermore, EfficientNet-ADV has just resisted weak attacks like $\epsilon = 2/255$, it = 10 and becomes unsteady with respect to intense attacks. Meanwhile, our EfficientNet-ADV is always robust against any attacks. Particularly, the proposed EfficientNet-ADV-B7 achieves the cutting-edge accuracy under both $\epsilon = 2/255$, it = 10 and 4/255, it = 100 perturbation, which are 93.72% and 92.14%, while the accuracies of pretrained EfficientNet-ADV-B7 are only 62.07% and 7.82%, respectively. In our understanding, 92.14% top-5 accuracy of our EfficientNet-ADV-B7 is currently the best robustness of DNN under strong PGD attack of $\epsilon = 4/255$, it = 100 among defense methods. From Table 6, it is emphasized that our proposal is stably robust on any networks against any attacks, although pretrained models are trained on natural or adversarial images.

6.2.2 C&W Attack

C&W attack is evaluated as an additional adversary to verify the robustness of our BwoBL algorithm. The convergence of C&W is slower than that of PGD. It hence expends considerable time to generate a C&W attack. In the original paper of C&W attack [4], Carlini et al. only assessed the first 1000 images in the test set of CIFAR-10 and MNIST. For that reason, we figure out the performance of all networks on the first 1000 images in the validation set of ImageNet.

Even though we determine structural hyperparameter $\alpha$ of the weights to build our BNNs based on PGD attack, our proposed models are still robust under strong C&W attack, as shown in Table 7. We realize ResNet architectures are powerful with regard to C&W attack, and our algorithm strengthens their performance more, i.e., a 28.6% rise on average. Other models, such as EfficientNets and SANs, are
and 84.4% maximum top-5 accuracy under strong C&W attacks on the proposed networks. Our robustness is substantially reinforced. From the recognition of baseline pretrained networks is less than 10%, our robustness is substantially reinforced. From the recognition of baseline pretrained networks is less than 10%, our proposed E

Table 7: Robustness of our proposed networks and pretrained DNNs to $l_2$ norm C&W attack are assessed by top-5 accuracies (%).

| Pretrained networks | Our proposal |
|---------------------|--------------|
| ResNet-50           | ResNet-101   | ResNet-152 | ResNet-101-32x8d |
| EffNet-B0           | EffNet-B1    | EffNet-B2  | EffNet-B3        |
| EffNet-ADV-B0       | EffNet-ADV-B1| EffNet-ADV-B2| EffNet-ADV-B3 |
| EffNet-B7           | EffNet-B8    | EffNet-B9  | EffNet-B0        |
| SAN19-pairwise      | SAN19-patchwise |             |
| ResNet-50-FAST-2px  | 70.16        | 56.61      | 56.49            |
| ResNet-50-FAST-4px  | 66.95        | 57.83      | 57.41            |
| EfficientNet-ADV-B0 | 42.42        | 12.72      | 8.31             |
| EfficientNet-ADV-B1 | 53.90        | 18.11      | 10.29            |
| EfficientNet-ADV-B2 | 55.52        | 18.90      | 10.34            |
| EfficientNet-ADV-B3 | 65.52        | 30.36      | 19.55            |
| EfficientNet-ADV-B4 | 64.57        | 30.53      | 16.61            |
| EfficientNet-ADV-B5 | 66.50        | 32.97      | 15.49            |
| EfficientNet-ADV-B6 | 64.90        | 31.51      | 12.33            |
| EfficientNet-ADV-B7 | 62.07        | 30.30      | 9.17             |

Table 6: Robustness to PGD attacks: $l_\infty$ norm, pixel perturbation $\epsilon = (2/255, 4/255)$, iteration $\mu = (10, 50, 100)$, are evaluated by top-5 accuracies (%) and compared between pretrained DNNs and our proposed networks.

| Pretrained networks | Our proposal |
|---------------------|--------------|
| ResNet-50           | ResNet-101   | ResNet-152 | ResNet-101-32x8d |
| EffNet-B0           | EffNet-B1    | EffNet-B2  | EffNet-B3        |
| EffNet-ADV-B0       | EffNet-ADV-B1| EffNet-ADV-B2| EffNet-ADV-B3 |
| EffNet-B7           | EffNet-B8    | EffNet-B9  | EffNet-B0        |
| SAN19-pairwise      | SAN19-patchwise |             |
| ResNet-50-FAST-2px  | 70.16        | 56.61      | 56.49            |
| ResNet-50-FAST-4px  | 66.95        | 57.83      | 57.41            |
| EfficientNet-ADV-B0 | 42.42        | 12.72      | 8.31             |
| EfficientNet-ADV-B1 | 53.90        | 18.11      | 10.29            |
| EfficientNet-ADV-B2 | 55.52        | 18.90      | 10.34            |
| EfficientNet-ADV-B3 | 65.52        | 30.36      | 19.55            |
| EfficientNet-ADV-B4 | 64.57        | 30.53      | 16.61            |
| EfficientNet-ADV-B5 | 66.50        | 32.97      | 15.49            |
| EfficientNet-ADV-B6 | 64.90        | 31.51      | 12.33            |
| EfficientNet-ADV-B7 | 62.07        | 30.30      | 9.17             |

It is pointed out that adversarial training with PGD attack cannot resist C&W attack. In Table 7, ResNet-50-FAST-2px and ResNet-50-FAST-4px, which are trained with PGD adversarial images only achieve 12.2% and 9.3% on C&W attack images. Moreover, when applying our algorithm to both of them, their top-5 accuracies increase by 4.4% and 6.0%, respectively. It means the application of our BwoBL to naturally pretrained networks is more robust than the application to pretrained models on adversarial examples. However, pretrained DNNs on the mixed dataset, including natural and PGD adversarial images, as EfficientNet-ADV might become the best defense against any attacks when they are combined with our BwoBL algorithm. As seen in Table 7, EfficientNet-ADV models are almost fooled by C&W attack but our EfficientNet-ADV networks considerably enhance the robustness. 94.2% of the proposed EfficientNet-ADV-B7 is the cutting-edge accuracy currently under strong C&W attack. Tables 6 and 7 verify the best robustness of our EfficientNet-ADV resisting both PGD and C&W attacks.

6.2.3 Comprehensive Assessment

To sum up, our BwoBL approach can construct BNNs based on learned parameters of DNNs and sharply improves the defense of pretrained DNNs against any adversarial attacks but does not need iteratively adversarial training. Figure 7 summarizes a comparison between the robustness of our proposal, standard and adversarial training. Standard train-
learning. Particularly, under $\ell_\infty$ norm PGD attack of pixel perturbation $\epsilon = 4/255$ with 100 iterations and strong $\ell_2$ norm C&W attack on ImageNet, the top-5 accuracies of our proposed models increase by 58.18% and 62.26% on average, respectively, which are combined with naturally pretrained networks. The most powerful model, i.e., our EfficientNet-ADV-B7, achieves 92.14% and 94.20% accuracy under these intense PGD and C&W attacks, which are the best robustness among defense methods recently.

Acknowledgments

Part of this research was supported by JSPS KAKENHI JP17H00730.

References

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” 2nd Int. Conf. Learning Representations - ICLR 2014, arXiv:1312.6199, pp.1–10, 2014.
[2] I.J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 3rd Int. Conf. Learning Representations - ICLR 2015, arXiv:1412.6572, pp.1–11, 2015.
[3] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, “Boosting adversarial attacks with momentum,” Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.9185–9193, 2018.
[4] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” 2017 IEEE Symposium on Security and Privacy (SP), pp.39–57, IEEE, 2017.
[5] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” 6th Int. Conf. Learning Representations - ICLR 2018, arXiv:1706.06083, pp.1–28, 2018.
[6] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z.B. Celik, and A. Swami, “Practical black-box attacks against machine learning,” Proc. 2017 ACM on Asia Conf. Computer and Communications Security, pp.506–519, April 2017.
[7] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, “Distillation as a defense to adversarial perturbations against deep neural networks,” 2016 IEEE Symposium on Security and Privacy (SP), pp.582–597, IEEE, 2016.
[8] N. Carlini and D. Wagner, “Adversarial examples are not easily detected: Bypassing ten detection methods,” Proc. 10th ACM Workshop on Artificial Intelligence and Security, pp.3–14, Nov. 2017.
[9] G.S. Dhillon, K. Azizzadenesheli, Z.C. Lipton, J. Bernstein, J. Kossaifi, A. Khanna, and A. Anandkumar, “Stochastic activation pruning for robust adversarial defense,” ICLR 2018, arXiv preprint arXiv:1803.01442, pp.1–13, 2018.
[10] H. Zhang, T.W. Weng, P.Y. Chen, C.J. Hsieh, and L. Daniel, “Efficient neural network robustness certification with general activation functions,” Advances in Neural Information Processing Systems NIPS, pp.4939–4948, 2018.
[11] A. Rozsa and T.E. Boult, “Improved adversarial robustness by reducing open space risk via tent activations,” arXiv preprint arXiv:1908.02435, pp.1–15, 2019.
[12] B. Wang, A. Lin, P. Yin, W. Zhu, A.L. Bertozzi, and S.J. Osher, “Adversarial defense via the data-dependent activation, total variation minimization, and adversarial training,” Inverse Problems & Imaging, vol.15, no.1, p.129, Feb. 2020.
[13] M. Tavakoli, F. Agostinelli, and P. Baldi, “Splash: Learnable activation functions for improving accuracy and adversarial robustness,” Neural Networks, vol.140, pp.1–12, Aug. 2021.
[14] N. Ye and Z. Zhu, “Bayesian adversarial learning,” Proc. 32nd Int.
Conf. Neural Information Processing Systems, pp.6892–6901, 2018.

[15] X. Liu, M. Cheng, H. Zhang, and C.J. Hsieh, “Towards robust neural networks via random self-ensemble,” Proc. European Conf. Comput. Vis. (ECCV), pp.369–385, 2018.

[16] X. Liu, Y. Li, C. Wu, and C.J. Hsieh, “Adv-bnn: Improved adversarial defense through robust bayesian neural network,” 7th Int. Conf. Learning Representations, pp.21–29, 2019.

[17] T.T.T. Khong, T. Nakada, and Y. Nakashima, “Bayes without bayesian learning for resisting adversarial attacks,” 2020 Eighth Int. Symp. Computing and Networking (CANDAR), pp.221–227, 2020.

[18] C.M. Bishop, “Bayesian methods for neural networks,” Technical report, Aston University, Birmingham, 1995.

[19] R.M. Neal, Bayesian learning for neural networks, Springer Science & Business Media, 2012.

[20] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, “Weight uncertainty in neural networks,” 32nd Int. Conf. Machine Learning, JMLR: W&CP, arXiv:1505.05424, pp.1–10, 2015.

[21] D. Barber and C.M. Bishop, “Ensemble learning in bayesian neural networks,” Nato ASI Series F Computer and Systems Sciences, vol.168, pp.215–238, 1998.

[22] Y. Gal and Z. Ghahramani, “Bayesian convolutional neural networks with bernoulli approximate variational inference,” arXiv preprint arXiv:1506.02158, pp.1–12, 2015.

[23] [Online], “Adversarial example generation,” Available: https://pytorch.org/tutorials/beginner/fgsmbeginner.html

[24] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” 5th Int. Conf. Learning Representations, ICLR 2017 - Workshop Track Proceedings, arXiv:1607.02533, pp.1–14, 2016.

[25] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” 2016 IEEE European symposium on security and privacy (EuroS&P), pp.372–387, IEEE, 2016.

[26] V. Zantedeschi, M.I. Nicolae, and A. Rawat, “Efficient defenses against adversarial attacks,” Proc. 10th ACM Workshop on Artificial Intelligence and Security, pp.39–49, Nov. 2017.

[27] C. Xie, Y. Wu, L.v.d. Maaten, A.L. Yuille, and K. He, “Feature denoising for improving adversarial robustness,” Proc. IEEE Conf. Comput. Vis. Pattern Recognit., pp.501–509, 2019.

[28] A. Shafahi, M. Najibi, A. Ghiass, Z. Xu, J. Dickerson, C. Studer, L.S. Davis, G. Taylor, and T. Goldstein, “Adversarial training for free!,” Advances in Neural Information Processing Systems, pp.3358–3369, 2019.

[29] E. Wong, L. Rice, and J.Z. Kolter, “Fast is better than free: Revisiting adversarial training,” 8th Int. Conf. Learning Representations, ICLR 2020, arXiv:2001.03994, pp.1–17, 2020.

[30] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial machine learning at scale,” 5th Int. Conf. Learning Representations - ICLR 2017, arXiv:1611.01236, pp.1–17, 2017.

[31] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, “Ensemble adversarial training: Attacks and defenses,” 6th Int. Conf. Learning Representations, ICLR 2018, arXiv:1705.07204, pp.1–22, 2018.

[32] S. Wang and C. Manning, “Fast dropout training,” International Conf. Mach. Learn., JMLR: W&CP, pp.118–126, June 2013.

[33] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” Proc. IEEE Conf. Comput. Vis. Pattern Recognit., pp.770–778, 2016.

[34] M. Tan and Q.V. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” 36th Int. Conf. Machine Learning, ICML 2019, pp.10691–10700, 2019.

[35] [Online], “Torchvision models,” Available: https://pytorch.org/docs/stable/torchvision/models.html

[36] [Online], “Efficientnet pytorch,” Available: https://github.com/lukemelas/EfficientNet-PyTorch

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need;” 31st Int. Conf. Neural Information Processing Systems (NIPS 2017), arXiv:1706.03762, pp.1–15, 2017.

[38] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, “Residual attention network for image classification,” Proc. IEEE Conf. Comput. Vis. Pattern Recognit., pp.3156–3164, 2017.

[39] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, “Self-attention generative adversarial networks,” International Conf. Mach. Learn., pp.7354–7363, PMLR, 2019.

[40] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, “Stand-alone self-attention in vision models,” arXiv preprint arXiv:1906.05909, 2019.

[41] H. Zhao, J. Jia, and V. Kolm, “Exploring self-attention for image recognition,” Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., pp.10076–10085, 2020.

[42] C. Xie, M. Tan, B. Gong, I. Wang, A.L. Yuille, and Q.V. Le, “Adversarial examples improve image recognition,” Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., pp.819–828, 2020.

[43] G.W. Ding, L. Wang, and X. Jin, “Advvertorch v0.1: An adversarial robustness toolbox based on pytorch,” arXiv preprint arXiv:1902.07623, pp.1–6, 2019.

[44] Q. Wan and X. Fu, “Fast-bbnn: Massive neuron skipping in bayesian convolutional neural networks,” 2020 53rd Annual IEEE/ACM Int. Symp. Microarchitecture (MICRO), pp.229–240, IEEE, 2020.

Thi Thu Thao Khong received the M.E. degree from VNU University of Engineering and Technology, Ha Noi, Vietnam in 2014. She has been a Ph.D. student in Division of Information Science, Nara Institute of Science and Technology, Japan since 2019. Her research interests include machine learning, signal processing, modeling and simulation.

Takashi Nakada received his M.E. and Ph.D. degrees from Toyohashi University of Technology in 2004 and 2007, respectively. He has been an Associate Professor at the Nara Institute of Science and Technology since 2016. His research interests include Normally-Off Computing, system architecture and related simulation technologies. He is a member of IEEE, ACM, and IPSJ.

Yasuhiro Nakashima received the B.E., M.E., and Ph.D. degrees in Computer Engineering from Kyoto University in 1986, 1988, and 1998, respectively. He was a computer architect in the Computer and System Architecture Department, FUJITSU Limited from 1988 to 1999. From 1999 to 2005, he was an Associate Professor in the Graduate School of Information Science, Nara Institute of Science and Technology. His research interests include processor architecture, emulation, CMOS circuit design, and evolutionary computation. He is a member of IEEE CS, ACM, and IPSJ.