IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 30 December 2023, accepted 14 January 2024, date of publication 19 January 2024, date of current version 19 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3355924

RESEARCH ARTICLE

Multi-Task Learning With Self-Defined Tasks for Adversarial Robustness of Deep Networks

CHANGHUN HYUN AND HYEYOUNG PARK
School of Computer Science and Engineering, Kyungpook National University, Daegu 41566, South Korea

Corresponding author: Hyeyoung Park (hypark@knu.ac.kr)

This work was supported in part by the Brain Korea (BK21) FOUR Project (Artificial Intelligence (AI)-Driven Convergence Software Education Research Program) funded by the Ministry of Education, School of Computer Science and Engineering, Kyungpook National University, South Korea, under Grant 4199990214394; in part by the National Research Foundation of Korea (NRF) Grant funded by the Korea Government [Ministry of Science and ICT (MSIT)] under Grant NRF-2020R1A2C1010020; and in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) Grant funded by the Korea Government (MSIT) (AI Innovation Hub) under Grant 2021-0-02068.

ABSTRACT Despite the considerable progress made in the development of deep neural networks (DNNs), their vulnerability to adversarial attacks remains a major hindrance to their practical application. Consequently, there has been a surge of interest and investment in researching adversarial attacks and defense mechanisms, with a considerable focus on comprehending the properties of adversarial robustness. Among these intriguing studies, a couple of works show that multi-task learning can enhance the adversarial robustness of DNNs. Based on the previous works, we propose an efficient way to improve the adversarial robustness of a given main task in a more practical multi-task learning scenario by leveraging self-defined auxiliary task. The core concept of our proposed approach lies not just in jointly training predefined auxiliary tasks but in manually defining auxiliary tasks based on the built-in labels of given data, which enables users to efficiently perform multi-task learning without the need for pre-defined auxiliary tasks. The newly generated self-defined tasks remain ''hidden'' from attackers and serve a supplementary role in improving the adversarial accuracy of the main task. In addition, the hidden auxiliary tasks also enable to build a rejection module that utilizes predictions from the auxiliary tasks to enhance the reliability of the prediction results. Through experiments conducted on five benchmark datasets, we confirmed that multi-task learning with self-defined hidden tasks can be actively employed to enhance the adversarial robustness and reliability.

INDEX TERMS Adversarial attack, adversarial robustness, multi-task learning, adversarial training, self-defined auxiliary tasks.

I. INTRODUCTION

Over the past few years, research on deep neural networks (DNNs) has advanced significantly, surpassing human abilities in multiple domains [1], [2], [3]. Particularly in computer vision, there has been substantial growth in research and practical applications such as autonomous vehicles and surveillance systems [4], [5], [6]. Consequently, experts forecast the gradual replacement of most human jobs by AI systems [7]. However, efforts to mitigate the adversarial vulnerability of DNNs to adversarial attacks are still insufficient, presenting a crucial and challenging problem for the future era of AI. Some previous studies have already emphasized the necessity of adversarial robustness from this perspective [8], [9], [10], [11].

In response to this need, the importance of research in the field of adversarial defense has increased substantially. Research on adversarial defense methods [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], the existence of adversarial examples [9], [26], [27], [28], [29], [30], and the properties of adversarial robustness has been conducted from various perspectives [23], [31], [32], [33], [34], [35], [36], [37], [38]. Among the various approaches, adversarial training [9], [23], [26], [39], [40] is one of the most widely used defense methods that employs adversarial examples as training data. While adversarial training is an

The associate editor coordinating the review of this manuscript and approving it for publication was Gustavo Olague.

effective method to significantly improve the robustness of DNNs, it suffers from decrease in the generalization ability of models [35], [36], [37]. In other words, adversarial training aims to classify adversarial examples, but this may complicate the model's decision boundaries and result in lower generalization performance on clean samples.

As a means of improving the generalization ability of DNNs, there are studies that utilize multi-task learning (MTL) [41], [42], [43], [44], [45]. Multi-task learning models perform better than models trained on a single task, as it enables generalized representation learning by optimizing the model for multiple tasks. Interestingly, there are several studies that have leveraged the benefits of multi-task learning to improve adversarial robustness [38], [46], [47]. Mao et al. [38] empirically demonstrated that a model trained on multiple tasks exhibits enhanced adversarial robustness for the main task compared to models trained individually. In a follow-up study, Ghamizi et al. [46] discussed the condition that multi-task learning can have a positive effect on adversarial robustness and proposed a guide for practitioners to develop multi-task learning models by conducting experiments under diverse conditions.

While previous studies [38], [46] have focused on improving the adversarial accuracy for the main task of multi-task learning models with the assumption that several pre-defined tasks are given, this paper considers more practical scenarios: When aiming to enhance the accuracy against adversarial examples for a particular main task through multi-task learning, there is no guarantee that pre-defined auxiliary tasks are always available or that the data for auxiliary tasks is fully labeled for training. Thus, we take an approach to self-defining auxiliary tasks for applying multi-task learning. The main goal of this research is to mitigate the degradation of accuracy for the main task against adversarial attacks, which means that the model produces correct output predictions for given adversarial examples. In the extreme case, we can assume that the attacker can perform white-box attack. However, our proposed self-defined task generation approach offers a vast array of potential auxiliary tasks that users can create which means unrecognizable to attacker. In this regard, we consider a gray-box attack scenario to evaluate the proposed multi-task learning approach.

The proposed self-defined task generation method enables users to easily define auxiliary tasks and employ them in both multi-task learning and adversarial training to improve the adversarial accuracy of the main task. As auxiliary tasks are generated by recombining the classes of the main task or by utilizing the inherent attributes of the data, there is no need for costly data-wise labeling. Furthermore, since auxiliary tasks are additionally defined, they are treated as hidden tasks that are not exposed to external attackers.

In this paper, we conducted experiments on five widely used datasets in computer vision. We measured the accuracy for clean samples and adversarial examples generated in the gray-box setting for each dataset. Additionally, in experiments involving adversarial sample rejection, we measured the accuracy of samples that passed through the rejection module. The experimental results demonstrate that self-defined hidden tasks have positive effects on the adversarial accuracy for the main task. We also endeavored to further improve the adversarial accuracy of the main task through adversarial sample rejection by leveraging the relationship between the main task and auxiliary tasks. To summarize, our contributions are as follows:

- We propose a multi-task learning approach to enhance the adversarial robustness of the main task in more practical situations by enabling users to efficiently generate auxiliary tasks without relying on predefined ones.
- We show that our multi-task learning approach can be easily combined with adversarial training to maximize the robustness of the model.
- We leverage the predictions of self-defined auxiliary tasks to reject inputs suspected of being adversarial examples and improve the reliability of prediction results of the main task.
- We empirically show that the adversarial accuracy of the main task can be improved by combining multi-task learning using self-defined hidden tasks with adversarial training and the rejection module.

II. RELATED WORKS

A. MULTI-TASK LEARNING

Multitask learning for neural networks was pioneered by Caruana [41]. The core idea of multi-task learning is to improve the generalization performance of the model by sharing and transferring knowledge between various tasks. It also enables a single model to perform multiple tasks, which is computationally more efficient than training an independent model for each task. Due to its advantages, multi-task learning has been widely applied across various domains of deep learning [43], [44], [45], [48].

Multi-task learning has also been applied to research on adversarial robustness [38], [46], [47]. Mao et al. [38] positively evaluated that the existing trade-off problem between clean accuracy and robustness gain [35] can be resolved since multi-task learning preserves clean accuracy while also improving adversarial robustness. In a follow-up study [46], Ghamizi et al claimed that there are varying levels of inherent adversarial vulnerability across tasks, and simply increasing the number of tasks does not guarantee a more robust model. Additionally, they proposed a guideline for task selection.

Although previous works [38], [46] have found intriguing properties of multi-task learning regarding adversarial robustness, the experiments were only conducted on limited types of tasks and datasets. The tasks in the previous works [38], [46] consist of pre-defined pixel-level prediction tasks, such as semantic segmentation, edge detection, and depth estimation. However, the discussion of improving robustness through

a combination of pre-defined auxiliary tasks is not always feasible in real-world scenarios. In this regard, we propose an approach to self-define new auxiliary tasks to enhance the robustness of a given main task, instead of using pre-defined tasks. Our proposed method involves defining the auxiliary tasks based on the built-in labels of the data and leveraging them to improve the reliability of the main task.

B. ADVERSARIAL ATTACK AND DEFENSE

Starting with the discussion of the high vulnerability of DNN systems to data with tiny, undetectable perturbations [9], there has been substantial research on adversarial attack and defense. Adversarial attacks fall into several categories based on their purpose and methodology. Among these, adversarial attack in this paper refers to evasion attacks, which have gained significant attention in this field. Evasion attacks generate adversarial examples by injecting human-imperceptible noise into input data, leading to malfunction of target model in the inference phase. Well-known adversarial attack algorithms include the Limited Memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) method [9], Fast Gradient Sign Method (FGSM) [26], Jacobian-based Saliency Map Attack (JSMA) [49], Basic Iterative Method (BIM) [50], Projected Gradient Descent (PGD) [23], and the Carlini & Wagner (CW) attack [55]. Among these, iterative first-order optimization-based adversaries [23], [55] are most often used as benchmarks due to their efficiency and intensity.

In line with the research on attack methods, adversarial defense has been studied from many aspects, including strengthening the model itself by training adversarial examples [9], [23], hiding the gradients of the model to make it difficult to attack [56], transforming input samples to remove perturbations [58], [59], [60], and randomized smoothing to guarantee correct prediction within a certain range of inputs [24], [25]. However, many defense methods are considered unreliable due to their limited applicability [51], [52], [53], [54], [55], [56], [57]. Developing reliable defense methods is challenging because there is a lack of precise comprehension of the phenomenon of adversarial examples, coupled with the need to consider various factors such as the attack and defense environment.

Instead of correctly classifying adversarial examples, other approaches for detection and rejection also form part of defense research [14], [15], [16], [17], [18], [19], [20], [21]. These detection and rejection methods also involve research in various aspects [21] such as dimensionality reduction for reducing attack success rate [18], detecting adversarial examples through statistical test and retraining them by assigning new labels [19], designing additional detectors to distinguish between adversarial examples and benign input [20]. An edge-consistency based detection method that utilizes multi-task learning to jointly train semantic segmentation and edge detection was also proposed [47].

While various defense methods have been studied, adversarial training [9], [23], [26], [39] is considered to be a reliable defense method in the literature to date. This is because, despite the limitations of adversarial training, it allows the model to perform robustly against adversarial attacks, and it also provides advantages that can be applied to general tasks. Adversarial training can serve as a benchmark for assessing both attack and defense by itself, or it can be combined with other defense methods. By combining adversarial training with our proposed multi-task learning approach, significant improvements in adversarial accuracy can be achieved.

III. METHODS

In this section, we first describe the problem we aim to address and the attack methods for model evaluation in Subsection A. In Subsection B, we explain the mechanism of self-defined task generation and its application for designing multi-task learning models, as well as the threat model. Subsection C covers the combination of the proposed multi-task learning models with adversarial training. Finally, in Subsection D, we explain the adversarial sample rejection using prediction consistency between main task and self-defined hidden tasks.

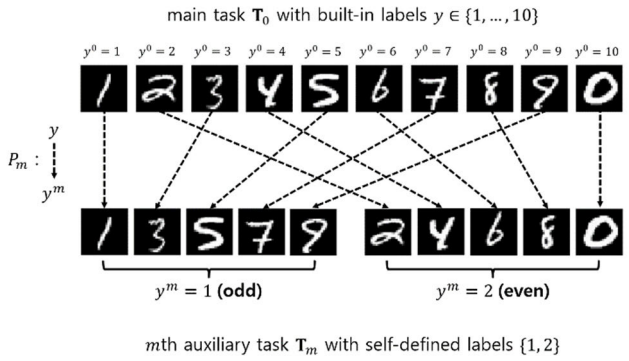


FIGURE 1. Illustration of self-defined task generation. Here, an auxiliary task is generated by transforming the main task's built-in labels with those for odd-even classification task, which is related to the mathematical characteristics of the data.

A. PROBLEM FORMULATION

In this study, our goal is to improve the adversarial accuracy of the main task, which involves recognizing the built-in labels of data. A model that is exclusively trained on the main task $\mathbf{T}_0$ is referred to as a "single task learning model". Considering the dataset for the main task $\mathrm{D}_0 = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, where $y_n \in \{1, \ldots, K\} (K \geq 2)$ denotes the target labels for K-class classification, we can write the single task loss L_0 as

$$L_0 = \sum_{n=1}^{N} l(\boldsymbol{x}_n, y_n) \tag{1}$$

where l denotes the loss function for each sample $(\boldsymbol{x}_n, y_n)$.

We assume that the adversary has white-box knowledge of the single task victim model: The adversary can access all specific settings of the victim model, including network

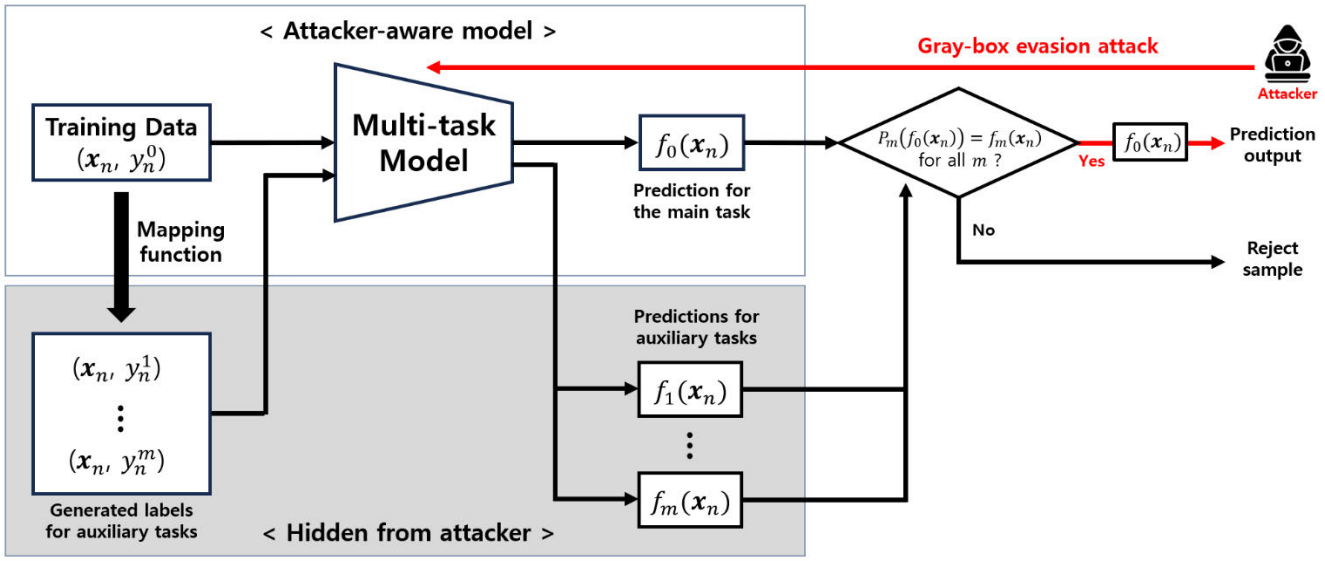


FIGURE 2. The overall process of our proposed multi-task learning approach with self-defined auxiliary tasks. The overall structure is simple, making it easily applicable to any dataset or network. The mapping function that generates auxiliary tasks varies based on user preferences or criteria, making it almost impossible to define a specific form. Hence, it can be naturally assumed that attackers are limited to conducting attacks within a gray-box environment.

structure, hyperparameters, model's output, dataset, etc. In this scenario, the adversary can generate adversarial examples to attack the victim model through the optimization,

$$\underset{\boldsymbol{\eta}}{\text{argmax}}\ L_0\left(\boldsymbol{\theta}_0, \boldsymbol{x}_n + \boldsymbol{\eta}, y_n\right), \quad \text{s.t.}\ \|\boldsymbol{\eta}\|_p \leq \epsilon, \tag{2}$$

where $\boldsymbol{\theta}_0$ is the trained weights of the single task learning model, $\boldsymbol{\eta}$ denotes the perturbation to be optimized and ϵ is the constraints. We now try to add some auxiliary tasks for multi-task learning that can be easily defined by using built-in labels.

Figure 2. illustrates the overall process of our proposed approach. By applying the mapping function which will be described in the following section, the users can generate auxiliary tasks and utilize them for training multi-task model. In inference phase, the trained model makes a prediction for the main task for the given query and does not return any prediction if the sample does not pass the rejection module. The self-defined auxiliary tasks and their predictions are only used in the rejection module and training the model, which means they are hidden from attacker. Thereby, we naturally assume that the adversary can attack the model with gray-box knowledge.

B. SELF-DEFINED HIDDEN TASKS

In the context of multi-task learning, auxiliary tasks are required alongside the previously defined main task $\mathbf{T}_0$ that needs to be solved originally. We manually generate additional tasks through "self-defined task generation". This process involves transforming the built-in labels (labels for the main task) of the data from different user-oriented perspectives.

For the mth auxiliary task, we define a new target label y_n^m for each input $\boldsymbol{x}_n$ using the built-in label y_n. In this context, we need to define a mapping function P_m from y_n to y_n^m. The first step is to determine the number of classes K_m for the mth auxiliary task to be generated. Then, each built-in label value $k \in \{1, \ldots, K\}$ is mapped to a new value $P_m(k) = k_m \in \{1, \ldots, K_m\}$ based on user-established criteria, such as visual or abstract characteristics of the data, or random selection. By applying the mapping function P_m to each target output $y_n \in \{1, \ldots, K\}$, we obtain the new target output $y_n^m \in \{1, \ldots, K_m\}$ for the auxiliary task, which can be written as $P_m(y_n) = y_n^m$. Applying this transformation to whole dataset D_0 results in generation of a new auxiliary task with dataset $\mathrm{D}_m = \{(\boldsymbol{x}_n, y_n^m)\}$. Note that the mapping function can be defined in the case where K is larger than K_m. Therefore, in this paper, we assume K is larger than 2.

The self-defined task generation allows users to generate new tasks from both visual and abstract characteristics of data (e.g., "curve shaped" for visual characteristics of road sign data, and "odd-even" for mathematical characteristics of digit data). Note that the users can easily generate auxiliary tasks by transforming the built-in labels from as many different perspectives as they desire. This approach offers significant convenience as intricate data analysis or costly labeling for individual data points is not required. An example of this process is illustrated in Figure 1.

We refer to a model trained on one or more additional auxiliary tasks $\mathbf{T}_m$ $(m = 1, \ldots, M)$ along with the main task $\mathbf{T}_0$ as "multi-task learning model". Let $\{1, \ldots, K_0\}$ and $\{1, \ldots, K_m\}$ denote the composition of classes for $\mathbf{T}_0$ and $\mathbf{T}_m$ respectively, where K_0 and K_m are the number of target labels for the main task and mth auxiliary task, respectively.

When training the multi-task learning model, the single task loss L_0 can be extended to a multi-task loss L by integrating the dataset D_m (m= 0, . . . ,M) as follows:

$$L = \sum_{n=1}^{N} \sum_{m=0}^{M} l\left(\boldsymbol{x}_n, y_n^m\right). \tag{3}$$

Note that these self-defined tasks are utilized to train the victim models, but do not directly produce outputs for queries. Therefore, they are ''hidden'' tasks that are unknown to the attacker, and we can naturally assume a gray-box scenario. The adversary perceives the multi-task victim model to be a single task victim model trained on the main task only. In the pursuit of generating strong attack samples, we assume that the adversary can access the trained weights in all layers except the output layers for auxiliary tasks in the multi-task victim model. Under this assumption, adversarial examples to attack the multi-task victim model can be generated through the optimization:

$$\underset{\boldsymbol{\eta}}{\text{argmax}}\ L\left(\boldsymbol{\theta}, \boldsymbol{x}_n + \boldsymbol{\eta}, y_n^0\right), \quad \text{s.t.}\ \|\boldsymbol{\eta}\|_p \leq \epsilon, \tag{4}$$

where $\boldsymbol{\theta}$ denotes the adversary-aware trained weights of the multi-task learning model, and y_n^0 is the target label for the main task.

C. COMBINATION WITH ADVERSARIAL TRAINING

Our proposed multi-task learning approach can be easily combined with other existing adversarial defense methods. In this paper, we apply the adversarial training methods to the proposed multi-task learning and investigate how the self-defined auxiliary tasks affect the adversarial robustness of the main task. We choose the PGD adversarial training [23], which is one of the most commonly used adversarial training methods. It generates adversarial examples using PGD attack for every mini batch in the training phase and trains the model to minimize its loss. The objective function of the PGD adversarial training in single task victim model can be formulated as:

$$\min_{\boldsymbol{\theta}_0} \mathbb{E}_{(\boldsymbol{x}, y^0) \sim D}\left[\min_{\boldsymbol{\eta} \leq \in} l(\boldsymbol{\theta}_0, \mathbf{x} + \boldsymbol{\eta}, y^0)\right]. \tag{5}$$

Based on the adversary's gray-box knowledge of the multi-task victim model assumed earlier, the adversarial examples used to attack the victim models are constructed only using the target labels for the main task. Equation (5) can be extended to our PGD adversarial training in multi-task victim model as follows:

$$\min_{\boldsymbol{\theta}} \sum_{m=0}^{M} l(\rho\left(\boldsymbol{\theta}\right), y^m), \tag{6}$$

where

$$\rho\left(\boldsymbol{\theta}\right) = \mathbb{E}_{(\boldsymbol{x}, y^0) \sim D}\left[\min_{\boldsymbol{\eta} \leq \in} l\left(\boldsymbol{\theta}, \boldsymbol{x} + \boldsymbol{\eta}, y^0\right)\right]. \tag{7}$$

As a result, the multi-task learning model with adversarial training is trained to minimize the multi-task loss for all tasks with given adversarial examples generated solely based on the information from the main task.

D. ADVERSARIAL SAMPLE REJECTION

The proposed self-defined hidden tasks can also be utilized for prediction-consistency based adversarial sample rejection. The rejection module validates the main task prediction for a given input sample by leveraging the predictions of auxiliary tasks, enabling the rejection of samples suspected to be adversarial examples. That is to say, it rejects given samples when the prediction results of the main task and auxiliary tasks conflict with each other. Referring to Figure 1, when the model recognizes the main task as ''1'' and the auxiliary task as ''even'' for a given input, a rejection occurs because ''1'' cannot be ''even''.

Denoting the prediction result for the main task is $f_0\left(\boldsymbol{x}_n\right)$ and that for the auxiliary task is $f_m\left(\boldsymbol{x}_n\right)$, the proposed rejection module R rejects $\boldsymbol{x}_n$ according to the following conditions:

$$R\left(\boldsymbol{x}_n\right) \begin{cases} = 1 \text{ if } P_m(f_0(\boldsymbol{x}_n)) \neq f_m\left(\boldsymbol{x}_n\right) \text{ for some } m \\ = 0 \text{ otherwise} \end{cases} \tag{8}$$

where P_m denotes the mapping function that is defined in Section III. The rejection module R rejects the input sample $\boldsymbol{x}_n$ if its result is 1 and passes otherwise. As the number of auxiliary tasks increases, the criteria for rejection can be adjusted more strictly. This rejection approach is possible because a mapping relationship between the labels of the auxiliary task and the main task is formed during the process of self-definition.

Note that this approach leverages the gray-box attack setting, where information about auxiliary tasks is hidden from the adversary. Thereby, the generated adversarial examples, which are based solely on the main task information, have limited impact on the predictions of the auxiliary tasks. This could reduce a significant drop in recognition performance on the main task when adversarial examples are given to the multi-task victim model. In addition, this approach of rejection does not need to set a data-dependent threshold, unlike conventional rejection methods.

IV. EXPERIMENTS

To confirm the effectiveness of our proposed method, we conducted experiments on five benchmark datasets. We performed these experiments using one of the widely employed repositories, ''advertorch'' [61]. In this section, we cover the general contents of the experimental settings: datasets, networks for victim models, training strategies, and attack parameters for generating adversarial examples. In the following subsections, we show the experimental results and analysis. We intend to describe our experimental setup in as much detail as possible for reproducibility.

A. DATASETS

We now proceed to describe the specific compositions of five datasets used in our experiments. We employed MNIST [62]

and CIFAR10 [65], which are frequently used not only in computer vision but also in the field of adversarial attack and defense for base experiments. However, these two datasets are relatively well-refined, making it difficult to evaluate the real-world applicability of the proposed method. Hence, we chose the SVHN [63] (Street View House Number) dataset which is known to be noisier and more challenging than MNIST because its images contain shape of other numbers around the central number being labeled. In order to assess the practical applicability, we used the GTSRB [64] dataset, commonly used in research related to traffic sign recognition of autonomous driving systems. Additionally, we employed the Tiny-ImageNet [69] dataset to test the applicability on a larger dataset.

TABLE 1. Self-defined hidden tasks used in the experiments. Each entry in the table represents properties of classes and their corresponding label values in parentheses.

Dataset	Auxiliary task 1	Auxiliary task 2
MNIST	odd(1), even(2)	prime(1), composite(2)
SVHN	odd(1), even(2)	prime(1), composite(2)
GTSRB	circle(1), polygon(2)	character(1), symbol(2)
CIFAR10	animal(1), vehicle(2)	sky(1), ground(2), water(3)
Tiny ImageNet	natural (1), artefact(2)	animal(1), machine(2), others(3)

MNIST contains 28×28 gray-scale images with a total of 10 class labels and consists of 60,000 training data and 10,000 test data. SVHN includes 32×32 RGB-scale numeric data which also has 10-digit class labels consists of 73,257 training and 26,032 test data. GTSRB (German Traffic Sign Recognition Benchmark) [64] dataset contains 43 different types of traffic signs found on the road and consists of 39,209 training and 12,630 test data. While the image sizes exhibit variation, we resize them to 112×112 pixels. CIFAR10 [65] is composed of 32×32 RGB-scale data featuring 10 distinct objects or animals and comprises 50,000 training and 10,000 test data. Lastly, Tiny ImageNet contains 100,000 training data of 200 classes (500 per class), each sample is 64×64 RGB-scale data and we resize them to 32×32. Since the test set of Tiny ImageNet is unlabeled data, we used the 10,000 validation samples to evaluate the model.

Table 1 shows the auxiliary tasks used in the experiments, which are made by the proposed self-defined task generation. The contents of the table represent the classes of each auxiliary task, with the corresponding label values in parentheses. When defining mapping function P_m for MNIST and SVHN, we utilized characteristics of integers. The ''odd-even'' task is to classify odd numbers and even numbers, and the ''prime number'' task is to classify prime and composite. For GTSRB, we used visual attributes: The label of ''circular-shaped'' task is determined by the frame shape of traffic sign (circular or polygon), and the ''include character'' task is to distinguish the presence of characters besides graphics. In the case of CIFAR10, we used semantic attributes: The ''animal'' task is to determine whether the object in the image is animal or not, and the ''activity area'' task is to categorize them based on their primary activity areas. Tiny ImageNet contains a much larger number of diverse classes than other datasets. Therefore, there exists a significantly larger variety of potential auxiliary tasks that users can generate compared to other datasets. For simple application of our proposed approach, we generated the ''natural or artefact'' task based on whether a sample is a natural thing (e.g., animal, vegetation, and natural landscape) or an artifact (e.g., building, structure, machine, and processed food). For the second auxiliary task, we generated the ''animal or machine'' task with three classes based on whether the sample belongs to animal, machine, or others. The data files used in our experiments are available at https://github.com/ChanghunHyun/Self-defined_MTL.

B. TRAINING VICTIM MODELS

We basically followed the backbone networks and other experimental settings in [23] for MNIST and CIFAR10: LeNet for MNIST and ResNet [66] and its wider version for CIFAR10, and other hyperparameter settings for the victim models. We employed AlexNet for GTSRB, and ResNet-18 was used for SVHN and Tiny ImageNet. The hyperparameters and optimizers are manually tuned. In the experiments with CIFAR10 and Tiny ImageNet, we utilized random data augmentation techniques such as horizontal flip to enhance the model's performance. In order to perform multi-task learning, the backbone layers prior to the final output layer are shared, and the output nodes are specifically designed based on the number of classes of each task. This hard parameter sharing with multi-head structure is commonly used in multi-task learning and is known to have the best effect in preventing overfitting and improving generalization performance [45].

We conducted both natural training and adversarial training on five datasets, adjusting the training hyperparameters for each run as necessary for learning convergence. We applied the SGD optimizer with a learning rate of 0.1 (decayed by a factor of ten every 50 epochs) for all datasets. The victim models for all datasets are trained until convergence which takes an average of 200 to a maximum of 300 epochs depending on the dataset and task. As discussed in [67] and [68], evaluating the model's performance using the checkpoint at the end of training epoch is appropriate for models that consistently exhibit a decreasing test loss. However, since overfitting commonly occurs during adversarial training, we selected a checkpoint that exhibited the best adversarial accuracy within the range where the training loss had sufficiently converged without a consistent increase in the test loss.

C. ATTACK METHODS

In order to evaluate the effect of multi-task learning models against various attacks, we choose three attack methods.

TABLE 2. Main-task classification accuracy on clean samples and three different types of adversarial attack samples (%). For each dataset, the first row shows the baseline performances of a single-task victim model trained for main task only, and the remaining three rows show the performances of the multi-task learning models with different task combinations. The second and third column blocks, separated by bold vertical lines, represent the training strategies of the victim models.

Dataset	Task Combination	Natural trained models				PGD Adversarial-Trained models			
		Clean samples	PGD attack targeted	PGD attack untargeted	CW attack untargeted	Clean samples	PGD attack targeted	PGD attack untargeted	CW attack untargeted
MNIST	**Main only**	**99.30**	**0.14**	**3.37**	**3.81**	**98.89**	**97.84**	**93.88**	**85.28**
	Main+Aux1	99.29	**0.30**	2.78	**3.93**	**99.09**	**98.03**	**94.46**	82.96
	Main+Aux2	**99.33**	**1.07**	2.50	**4.09**	**98.95**	**98.05**	**94.82**	**87.16**
	Main+Aux1+Aux2	**99.40**	**0.74**	1.40	**4.02**	**98.99**	**98.13**	**95.13**	**86.84**
SVHN	**Main only**	**95.05**	**1.08**	**0.64**	**4.09**	**90.15**	**70.18**	**52.51**	**53.32**
	Main+Aux1	**95.47**	**2.19**	**0.84**	**4.31**	**90.42**	69.99	**54.26**	**56.14**
	Main+Aux2	**95.33**	**1.99**	**0.74**	4.00	89.56	68.72	**53.20**	**54.23**
	Main+Aux1+Aux2	**95.37**	**2.64**	**1.47**	**4.23**	90.05	**70.82**	**54.18**	**54.34**
GTSRB	**Main only**	**97.58**	**13.23**	**25.46**	**4.65**	**85.65**	**71.35**	**52.16**	**48.50**
	Main+Aux1	**98.48**	**15.89**	**32.76**	**7.43**	**90.05**	**76.28**	**58.72**	**57.44**
	Main+Aux2	**97.83**	**17.24**	**31.08**	**6.06**	**88.87**	**75.12**	**59.20**	**56.65**
	Main+Aux1+Aux2	**98.14**	**14.53**	21.72	4.59	**89.83**	**76.18**	**59.94**	**56.48**
CIFAR10	**Main only**	**93.87**	**0.05**	**0.00**	**0.00**	**84.85**	**68.63**	**44.32**	**40.10**
	Main+Aux1	**94.19**	**0.21**	**0.05**	**0.05**	84.62	67.83	**45.23**	39.78
	Main+Aux2	**94.17**	**0.25**	**0.01**	**0.04**	84.79	67.85	**45.77**	**40.68**
	Main+Aux1+Aux2	**94.07**	**0.48**	**0.11**	**0.05**	**84.94**	**68.64**	**46.68**	**42.13**
Tiny ImageNet	**Main only**	**44.39**	**0.45**	**0.01**	**0.02**	**20.05**	**19.20**	**4.70**	**7.56**
	Main+Aux1	44.01	**0.58**	0.01	**0.04**	18.68	18.82	**6.14**	**7.93**
	Main+Aux2	44.18	**0.77**	**0.02**	**0.05**	17.88	18.04	**5.51**	**7.57**
	Main+Aux1+Aux2	43.04	**0.82**	**0.10**	0.02	18.39	18.31	**6.03**	**8.16**

Firstly, we evaluate the models against l_∞-bounded PGD attack which is one of the strongest first order adversaries. For PGD attack on MNIST dataset, we perform l_∞-bounded PGD attack, 40 iteration runs with step size 0.01 and $\epsilon = 0.3$ for perturbation, while 7 iteration runs with step size 2/255 and $\epsilon = 8/255$ for SVHN, GTSRB, CIFAR10, and Tiny ImageNet. In PGD targeted attack, the target is assigned to a class that is $+1$ to the sample's ground-truth. These attack parameters are chosen considering previous studies to generate sufficiently small perturbations for deceiving the model [23].

The experiments against CW [55] attack, which is another strong optimization-based unbounded attack method, serve the purpose of evaluating the model using a different attack strategy from that used in adversarial training. We set 5 binary search steps, 0 confidence, initial constant as 1 equally, and the remaining parameters are set as follows: 0.01 learning rate for MNIST, and CIFAR10, 0.001 for SVHN, GTSRB and Tiny ImageNet.

D. EFFECT OF MULTI-TASK LEARNING

In this section, we investigate how multi-task learning affects the adversarial robustness of the main task. We first present and analyze the experimental results of self-defined multi-task learning without adversarial sample rejection. Table 2 shows the classification accuracies on test data for various multi-task learning models. The second column of the table indicates task combinations: single task learning (main task only), and three multi-task combinations. Next, the two large column blocks, each divided into four branches, show the evaluation results for two different learning strategies, depending on the data and loss used for training. Specifically, the left four columns show the results for the natural training using only the original data, and the right four columns show the results for the model obtained by performing adversarial training. For each trained model, we investigate its performance on four different test data: original clean samples and adversarial samples by different attack methods. In order to assess the robustness gain of the multi-task models, we need to compare the performance of single task model with other multi-task combinations. To enhance visibility, the performance values of single task model are highlighted in bold black font and shaded cells, and those of the multi-task models that exhibit improvement over the single task model are shown in bold blue.

As is well known in the literature, we can see that the overall accuracy of natural trained models against adversarial samples are very low without any defense technique. However, compared to the single task model, the multi-task learning models exhibit an overall improvement in the clean and adversarial accuracy. In the case of adversarial-trained models, the accuracies of single task model against adversarial examples exhibit strong robustness. Furthermore, compared to the single task model, the multi-task learning with the proposed self-defined tasks exhibits a substantial improvement in the majority of cases.

In terms of the trade-off between clean accuracy and adversarial robustness [35], we can see that there is a significant trade-off induced by adversarial training when comparing

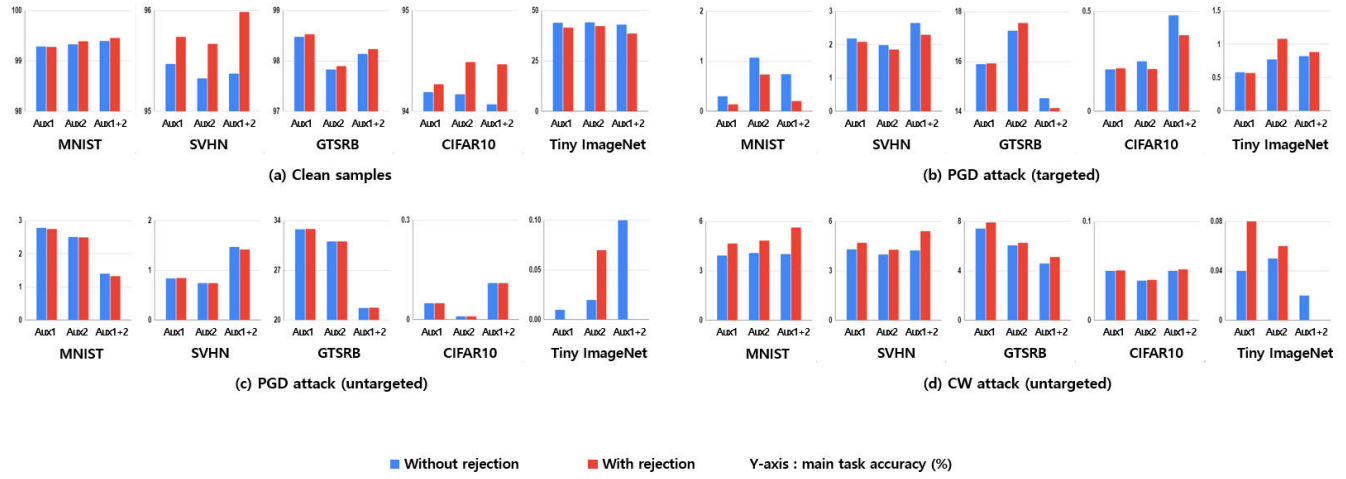


FIGURE 3. Main task adversarial accuracy of natural training models with and without rejection.

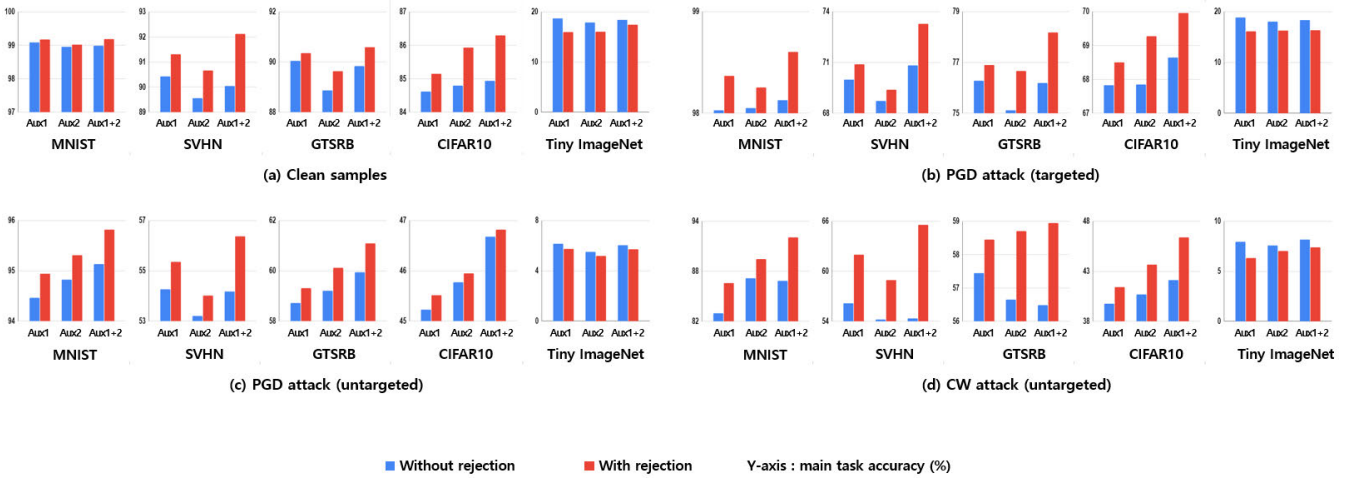


FIGURE 4. Main task adversarial accuracy of PGD adversarial-trained models with and without rejection.

PGD adversarial trained models and naturally trained models. However, when comparing single task and multi-task learning models under the same training strategy, we can see that multi-task learning compensates for the generalization ability lost through adversarial training.

In summary, when comparing single task to multi-task victim models, multi-task learning with natural training improved adversarial accuracy by an average of 0.44%, with a maximum improvement of 7.30%. With adversarial training, multi-task learning yielded an average adversarial accuracy improvement of 1.44%, with a maximum improvement of 8.95%. These results suggest that employing multi-task learning with self-defined hidden tasks can enhance a model's adversarial robustness.

However, the increase in the number of tasks or task combinations in the multi-task models does not always increase the robustness gain, which is in the same context as the results in [46]. In a minority of cases, the multi-task victim model has no robustness gain, or rather, the robustness was slightly lower than that of the single task victim model. As can be seen in the experiments on GTSRB and MNIST, we observed that using one auxiliary task for multi-task learning sometimes resulted in a more significant robustness gain compared to using two auxiliary tasks. This suggests that the model's robustness gain can vary depending on the combination of tasks, emphasizing the need to establish criteria for generating appropriate self-defined hidden tasks. In this regard, it might be worthwhile to consider using the guidance for task selection proposed by Ghamizi et al. [46].

For more exploration of the experiment, the results appear to be related to the difficulty level of the original tasks. In the case of MNIST, which involves simple grayscale digit recognition, the task's difficulty is relatively lower than other datasets. Consequently, the adversarial-trained model exhibits recognition rates on clean data nearly identical to those of the naturally trained model, while also demonstrat-

TABLE 3. Classification accuracy of auxiliary tasks in multi-task learning models (%). For each dataset, the values located in the third row represent the average accuracy for the two auxiliary tasks. Compared to the PGD adversarial-trained model, the recognition performance of the auxiliary tasks in the natural training model is generally lower and exhibits higher variance. The values highlighted in red represent the accuracies of auxiliary tasks in cases where the main task's accuracy decreased through the rejection module.

Dataset	Task Combination	Natural training models				PGD Adversarial-Trained models			
		Clean samples	PGD attack targeted	PGD attack untargeted	CW attack untargeted	Clean samples	PGD attack targeted	PGD attack untargeted	CW attack untargeted
MNIST	Main+Aux1	99.59	1.65	69.45	80.22	99.32	97.95	96.67	95.82
	Main+Aux2	99.67	39.87	72.88	85.15	99.34	98.63	97.60	96.85
	Main+Aux1+Aux2	99.65	20.78	67.47	81.69	99.53	98.43	97.51	96.78
SVHN	Main+Aux1	97.56	4.01	70.31	79.27	95.92	72.79	75.20	87.28
	Main+Aux2	97.81	37.52	76.13	84.28	94.95	83.27	76.76	88.59
	Main+Aux1+Aux2	97.68	21.52	65.83	77.59	95.13	78.99	76.21	87.30
GTSRB	Main+Aux1	99.83	75.49	93.95	92.38	98.88	96.53	94.50	97.80
	Main+Aux2	99.61	61.36	94.42	92.23	96.93	92.48	88.25	94.45
	Main+Aux1+Aux2	99.69	66.05	91.93	92.83	98.03	95.06	91.85	96.00
CIFAR10	Main+Aux1	99.22	80.27	89.99	91.08	97.42	96.58	91.28	93.92
	Main+Aux2	50.30	60.03	50.17	54.30	50.86	57.02	50.79	54.27
	Main+Aux1+Aux2	98.21	50.64	71.57	73.58	94.91	88.95	80.49	85.27
Tiny ImageNet	Main+Aux1	76.87	83.37	43.82	72.91	66.82	68.53	63.94	66.73
	Main+Aux2	67.47	61.30	26.90	61.66	57.66	58.53	55.65	57.58
	Main+Aux1+Aux2	72.57	72.85	34.53	67.92	62.58	62.94	60.10	62.49

ing a high level of robustness against adversarial samples. Conversely, the SVHN dataset presents a more challenging and practical digit recognition task compared to MNIST due to the increased presence of noise around the digits being recognized. Often, there are neighboring digits near the target digit, making the task significantly more challenging. As a result, SVHN generally shows lower overall recognition performance compared to MNIST. GTSRB has a larger number of classes compared to MNIST, SVHN, and CIFAR10 datasets. In terms of adversarial-trained model performance, it exhibits performance levels similar to SVHN. CIFAR10 has fewer classes compared to GTSRB, yet it is characterized by significant variation within classes. In both natural-trained and adversarial-trained models, CIFAR10 exhibits lower performance compared to other datasets. Tiny ImageNet, which has the largest number of classes and substantial intra-class variation, faced challenges in establishing a high base performance for the model. Consequently, this difficulty leads to low robustness in adversarial-trained models as well.

One notable point can be observed in the experimental results of GTSRB. The naturally trained GTSRB model exhibits a higher recognition rate against PGD attacks compared to other datasets. This phenomenon seems to stem from the characteristics inherent in the dataset itself. GTSRB comprises relatively well-refined data where the frames of traffic signs and the symbols or characters within them are distinct and clear. The variability of GTSRB mainly arises from changes in lighting or background noise. This characteristic indicates high inter-class variation while exhibiting low intra-class variation. Consequently, even when attacked using the same method and parameters as other datasets, natural trained GTSRB model shows a higher resilience.

E. RESULTS WITH REJECTION MODULE

In order to increase the reliability of the system's output, we can use the proposed rejection module for refusing to make predictions for data that is suspected of being an attack (or for which the confidence on the prediction is low). First, we analyze the experimental results of four datasets except for Tiny ImageNet. When we applied the rejection module to the four datasets, the average rejection rate for clean samples is 0.44% in the natural trained models, while 1.64% in adversarial-trained models, indicating that the rejection rate for clean samples is not significant in both training strategies. The average rejection rate for adversarial examples in the natural trained model is 4.40%, while that of in adversarial training is 4.71%. This rejection rate may be considered insufficient as the module's performance in detecting the attack. However, the significance of the rejection mechanism in this study lies in its ability to supplement the low recognition performance of the main task against adversarial examples by utilizing the relatively higher performance of the auxiliary tasks as will be discussed more in the last part of this section.

Figures 3 and 4 show the changes in accuracy when the rejection module is employed. Figure 3 displays the performance changes with the natural trained model and Figure 4 shows the changes achieved through the PGD adversarial-trained model. The blue bars represent the accuracy of the main task without utilizing the rejection module, which are equivalent to the performance values presented in Table 2 for the multi-task victim models. The red bars represent the accuracy for the input samples filtered through the rejection module. In the natural trained model (Figure 3), the overall effect of utilizing the rejection module to filter input samples is not significantly pronounced. However, in the PGD adversarial-trained model (Figure 4), the main task adversarial accuracies are improved for all datasets and task combinations through the sample filtering effect of the rejection module.

Table 3 shows the accuracy of auxiliary tasks for multi-task learning models based on various learning strategies,

datasets, and task combinations explored in this study. In cases where two auxiliary tasks are employed, the average accuracy for both tasks is indicated. The difference between the rejection effects is due to uneven and low auxiliary task accuracies of natural trained models which are highlighted in red in Table 3. Precisely, in 13 out of 48 task combinations in natural trained models show negative effect through the rejection module. On the other hand, in the adversarial-trained model, the recognition performance of the auxiliary task against adversarial examples is at least around 73%, significantly higher than that of the natural trained model. Therefore, it can be considered that the adversarial-trained model performs a more reliable rejection compared to the natural trained model.

Experimental results with rejection module on Tiny ImageNet require a different interpretation. Since the Tiny ImageNet is a very complex dataset, the clean accuracy of the natural trained model is very low, around 44%. The trade-off between clean accuracy and robustness was severe, and adversarial training did not improve the recognition performance of auxiliary tasks compared to natural trained model. We estimate that the low base performance of the model is the reason why the proposed rejection module did not improve the performance of the model. However, experiments on Tiny ImageNet without rejection module still consistently demonstrated that multi-task learning through self-defined auxiliary tasks improves robustness in both natural and adversarial-trained models. Moreover, the significance of this study is that the effectiveness of the proposed rejection module is evident on practical datasets such as SVHN and GTSRB. For larger and more complex datasets, it is expected that applying the proposed rejection module after maximizing the base performance of the model through appropriate learning strategies would yield positive effect.

V. LIMITATIONS

While our proposed multi-task learning with self-defined task generation demonstrates a positive impact on enhancing the adversarial robustness of the model diverse datasets, it exhibits several limitations. Firstly, as mentioned in Section III-B, our experiments to date have been limited to cases where the number of labels of built-in label K is larger than the number of labels of mth auxiliary task K_m. However, our proposed approach holds potential applicability in cases where K_m is larger than K, and experimental verification will be one of our future works. The second limitation is that our proposed rejection module is dependent on the model's recognition performance for auxiliary tasks. As shown in the experimental results of the natural trained model on various datasets, when the recognition performance for auxiliary tasks against adversarial examples is inadequate, the proposed rejection module does not always positively impact on the main task performance. Hence, there is a need for further research aimed at refining the rejection mechanism or enhancing model's performance for auxiliary tasks against adversarial examples.

VI. CONCLUSION

In this paper, we propose an efficient method for enhancing adversarial robustness of DNNs by multi-task learning with self-defined hidden tasks and adversarial training. By mapping built-in labels of main task to new labels, users can generate auxiliary tasks without additional labeling costs and leverage them for multi-task learning. Moreover, it is also possible to use the prediction results of auxiliary tasks for adversarial sample rejection. In practical scenarios, self-defined hidden tasks are difficult for attackers to discern, and these are suitable as supplementary tasks for improving the main task's adversarial accuracy. Through experiments conducted on five datasets, we confirmed that our proposed approach is effective in robustness gain for the main task. However, since not all combinations of tasks improve the adversarial robustness of the main task, establishing criteria for appropriate task generation is still needed, and this will be one of our future works. In addition, we plan to explore the application of our proposed method to binary classification problems as well as regression problems. Furthermore, the method of using information from self-defined hidden tasks in the prediction process of the main task will also be studied.

REFERENCES

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[2] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1701–1708.

[3] C. Lu and X. Tang, "Surpassing human-level face verification performance on LFW with GaussianFace," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 1–9.

[4] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 7073–7083.

[5] F. Angelos, T. Panagiotis, M. Rowan, R. Nicholas, L. Sergey, and G. Yarin, "Can autonomous vehicles identify, recover from, and adapt to distribution shifts?" in *Proc. IEEE Int. Conf. Mach. Learn.*, Nov. 2020, pp. 3145–3153.

[6] D. Wu, S.-J. Zheng, X.-P. Zhang, C.-A. Yuan, F. Cheng, Y. Zhao, Y.-J. Lin, Z.-Q. Zhao, Y.-L. Jiang, and D.-S. Huang, "Deep learning-based methods for person re-identification: A comprehensive review," *Neurocomputing*, vol. 337, pp. 354–371, Apr. 2019.

[7] K. Grace, J. Salvatier, A. Dafoe, B. Zhang, and O. Evans, "ViewPoint: When will AI exceed human performance? Evidence from AI experts," *J. Artif. Intell. Res.*, vol. 62, pp. 729–754, Jul. 2018.

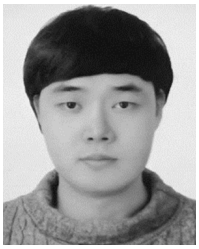
[8] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, 2015, pp. 427–436.

[9] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," 2013, *arXiv:1312.6199*.

[10] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Proc. Eur. Conf. ECML PKDD*. Berlin, Germany: Springer, Sep. 2013, pp. 387–402.

[11] D. Hendrycks, N. Carlini, J. Schulman, and J. Steinhardt, "Unsolved problems in ML safety," 2021, *arXiv:2109.13916*.
[12] S. Henrique Silva and P. Najafirad, "Opportunities and challenges in deep learning adversarial robustness: A survey," 2020, *arXiv:2007.00753*.
[13] H. Liang, E. He, Y. Zhao, Z. Jia, and H. Li, "Adversarial attack and defense: A survey," *Electronics*, vol. 11, no. 8, p. 1283, Apr. 2022.
[14] J. Chen, J. Raghuram, J. Choi, X. Wu, Y. Liang, and S. Jha, "Revisiting adversarial robustness of classifiers with a reject option," in *Proc. AAAI-22 Workshop Adversarial Mach. Learn. Beyond*, Dec. 2021, pp. 1–10.
[15] P. Habib Zadeh, R. Hosseini, and S. Sra, "Deep-RBF networks revisited: Robust classification with rejection," 2018, *arXiv:1812.03190*.
[16] F. Crecchi, M. Melis, A. Sotgiu, D. Bacciu, and B. Biggio, "FADER: Fast adversarial example rejection," *Neurocomputing*, vol. 470, pp. 257–268, Jan. 2022.
[17] J. Lu, T. Issaranon, and D. Forsyth, "SafetyNet: Detecting and rejecting adversarial examples robustly," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 446–454.
[18] D. Hendrycks and K. Gimpel, "Early methods for detecting adversarial images," 2016, *arXiv:1608.00530*.
[19] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, "On the (statistical) detection of adversarial examples," 2017, *arXiv:1702.06280*.
[20] J. Hendrik Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," 2017, *arXiv:1702.04267*.
[21] A. Aldahdooh, W. Hamidouche, S. A. Fezza, and O. Déforges, "Adversarial example detection for DNN models: A review and experimental comparison," *Artif. Intell. Rev.*, vol. 55, no. 6, pp. 4403–4462, Aug. 2022.
[22] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2016, pp. 582–597.
[23] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2017, *arXiv:1706.06083*.
[24] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 656–672.
[25] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *Proc. ICML*, 2019, pp. 1310–1320.
[26] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.
[27] A. Ilyas, S. Santurkar, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–12.
[28] S. Bubeck, Y. T. Lee, E. Price, and I. Razenshteyn, "Adversarial examples from computational constraints," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 831–840.
[29] T. Tanay and L. Griffin, "A boundary tilting persepective on the phenomenon of adversarial examples," 2016, *arXiv:1608.07690*.
[30] S. Han, C. Lin, C. Shen, Q. Wang, and X. Guan, "Interpreting adversarial examples in deep learning: A review," *ACM Comput. Surv.*, vol. 55, no. 14s, pp. 1–38, Dec. 2023.
[31] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," 2016, *arXiv:1611.01236*.
[32] S. Ye, K. Xu, S. Liu, H. Cheng, J.-H. Lambrechts, H. Zhang, A. Zhou, K. Ma, Y. Wang, and X. Lin, "Adversarial robustness vs. model compression, or both?" in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 111–120.
[33] D. Rodriguez, T. Nayak, Y. Chen, R. Krishnan, and Y. Huang, "On the role of deep learning model complexity in adversarial robustness for medical images," *BMC Med. Informat. Decis. Making*, vol. 22, no. S2, pp. 1–15, Dec. 2022.
[34] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: From phenomena to black-box attacks using adversarial samples," 2016, *arXiv:1605.07277*.
[35] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "Robustness may be at odds with accuracy," in *Proc. ICLR*, 2019, pp. 1–24.
[36] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. E. Ghaoui, and M. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7472–7482.
[37] A. Raghunathan, S. Michael Xie, F. Yang, J. Duchi, and P. Liang, "Understanding and mitigating the tradeoff between robustness and accuracy," 2020, *arXiv:2002.10716*.
[38] C. Mao, A. Gupta, V. Nitin, B. Ray, S. Song, J. Yang, and C. Vondrick, "Multitask learning strengthens adversarial robustness," in *Proc. Eur. Comput. Vis.*, 2020, pp. 158–174.
[39] A. Shafahi, M. Najibi, M. A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, and T. Goldstein, "Adversarial training for free!" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.
[40] H. Wang, A. Zhang, S. Zheng, X. Shi, M. Li, and Z. Wang, "Removing batch normalization boosts adversarial training," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2022, pp. 23433–23445.
[41] R. Caruana, "Multitask learning: A knowledge-based source of inductive bias," in *Proc. Int. Conf. Mach. Learn.*, 1993, pp. 41–48.
[42] T. Standley, A. Zamir, D. Chen, L. Guibas, J. Malik, and S. Savarese, "Which tasks should be learned together in multi-task learning?" in *Proc. Int. Conf. Mach. Learn.*, Jul. 2020.
[43] S. Ruder, "An overview of multi-task learning in deep neural networks," 2017, *arXiv:1706.05098*.
[44] S. Vandenhende, S. Georgoulis, W. Van Gansbeke, M. Proesmans, D. Dai, and L. Van Gool, "Multi-task learning for dense prediction tasks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3614–3633, Jul. 2022.
[45] Y. Zhang and Q. Yang, "A survey on multi-task learning," 2017, *arXiv:1707.08114*.
[46] S. Ghamizi, M. Cordy, M. Papadakis, and Y. Le Traon, "Adversarial robustness in multi-task learning: Promises and illusions," in *Proc. AAAI Conf. Artif. Intell.*, no. 1, 2022, pp. 697–705.
[47] M. Klingner, V. R. Kumar, S. Yogamani, A. Bär, and T. Fingscheidt, "Detecting adversarial perturbations in multi-task perception," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 13050–13057.
[48] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. 25th Int. Conf. Mach. Learn.*, Helsinki, Finland, 2008, pp. 160–167.
[49] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Secur. Privacy*, Mar. 2016, pp. 372–387.
[50] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Artificial Intelligence Safety and Security*. Boca Raton, FL, USA: CRC Press, 2018, pp. 99–112.
[51] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 274–283.
[52] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, Nov. 2017, pp. 3–14.
[53] F. Tramer, N. Carlini, W. Brendel, and A. Madry, "On adaptive attacks to adversarial example defenses," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1633–1645.
[54] N. Carlini and D. Wagner, "Defensive distillation is not robust to adversarial examples," 2016, *arXiv:1607.04311*.
[55] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.
[56] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, Apr. 2017, pp. 506–519.
[57] W. He, J. Wei, X. Chen, N. Carlini, and D. Song, "Adversarial example defense: Ensembles of weak defenses are not strong," in *Proc. 11th USENIX Workshop Offensive Technologies*, 2017.
[58] Z. Liu, "Feature distillation: DNN-oriented JPEG compression against adversarial examples," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 860–868.
[59] A. Graese, A. Rozsa, and T. E. Boult, "Assessing threat of adversarial examples on deep neural networks," in *Proc. 15th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2016, pp. 69–74.
[60] A. Prakash, N. Moran, S. Garber, A. DiLillo, and J. Storer, "Deflecting adversarial attacks with pixel deflection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8571–8580.
[61] G. Weiguang Ding, L. Wang, and X. Jin, "Advertorch v0.1: An adversarial robustness toolbox based on PyTorch," 2019, *arXiv:1902.07623*.
[62] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Mar. 1998.

[63] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, 2011.
[64] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Netw.*, vol. 32, pp. 323–332, Aug. 2012.
[65] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.
[66] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[67] L. Rice, E. Wong, and Z. Kolter, "Overfitting in adversarially robust deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 8093–8104.
[68] C. Yu, B. Han, L. Shen, J. Yu, C. Gong, M. Gong, and T. Liu, "Understanding robust overfitting of adversarial training and beyond," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2022, pp. 25595–25610.
[69] Y. Le and X. Yang, "Tiny ImageNet visual recognition challenge," *CS 231N*, vol. 7, no. 7, p. 3, 2015.



CHANGHUN HYUN received the B.S. and M.S. degrees from the School of Computer Science and Engineering, Kyungpook National University, Daegu, South Korea, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree. His current research interests include deep learning, computer vision, and adversarial robustness of DNNs.

HYEYOUNG PARK received the B.S. (summa cum laude), M.S., and Ph.D. degrees in computer science from Yonsei University, Seoul, South Korea, in 1994, 1996, and 2000, respectively. She was a member of Research Staff with the Brain Science Institute, RIKEN, Japan, from 2000 to 2004. She is currently a Professor with the School of Computer Science and Engineering, Kyungpook National University, Daegu, South Korea. Her current research interests include computational learning theory, machine learning theory, and their application to various fields, such as pattern recognition, image processing, and data mining.