Towards Cross-task Universal Perturbation against Black-box Object Detectors in Autonomous Driving

Quanxin Zhang[a], Yuhang Zhao[a,d], Yajie Wang[a], Thar Baker[b], Jian Zhang[c], Jingjing Hu[a,*]

[a]*School of Computer Science and Technology, Beijing Institute of Technology, China*
[b]*Department of Computer Science, Liverpool John Moores University, United Kingdom*
[c]*College of Cyber Science, Nankai University, China*
[d]*Institute of Artificial Intelligence and Blockchain, Guangzhou University, China*

Abstract

Deep neural network is the main research branch in artificial intelligence and suitable for many decision-making fields. Autonomous driving and unmanned vehicle often depend on deep neural networks for accurate and reliable detection, classification, and ranging of surrounding objects in real on-road environments, either locally or by swarm intelligence among distributed nodes via 5G channel. But, it has been demonstrated that deep neural networks are vulnerable to well-designed adversarial examples that are imperceptible to human eyes in computer vision tasks. It is valuable to study the vulnerability for enhancing the robustness of neural networks. However, existing adversarial examples against object detection models are image-dependent, so in this paper, we implement adversarial attacks against object detection models using universal perturbations. We find the cross-task, cross-model, and cross-dataset transferability of universal perturbations, we train universal perturbations generator firstly and then add the universal perturbations to the target images in two ways: resizing and pile-up, in order to solve the problem that universal perturbations cannot be directly applied to attack object detection models. We use the transferability of universal perturbations to attack black-box object detection models. In this way, the time cost of generating adversarial examples is reduced. A series of experiments are conducted on PASCAL VOC and MS COCO datasets demonstrating the feasibility of cross-task attacks and proving the effective-

*Corresponding author: Hu Jingjing. email:hujingjing@bit.edu.cn

July 2, 2020

ness of our attack on two representative object detectors: regression-based models like YOLOv3 and proposal-based models like Faster R-CNN.

Keywords:
Adversarial example, Object detection, Universal perturbation

1. Introduction

As one of the most incredibly influential parts of Artificial Intelligence (AI), Deep Neural Network (DNN) is designed to mimic the network of neurons that makes up a human brain so that computers will understand things and make decisions in a human-like manner. It has been widely applied in people's life, such as unmanned vehicles, autonomous driving, monitoring security [1, 2, 3], and so on. An important and challenging application of DNN is performing real-time object detection to help a camera locate instances of semantic objects of a certain class [4, 5, 6, 7], and recognize each physical shape [8] as humans do. Some of the detection decisions are made locally, and the others are shared among distributed vehicles or uploaded to the cloud via broadband 5G channel for coordination and supervision since the detection results are very critical for the safety of autonomous driving. However, recent studies have shown that deep learning models are highly susceptible to adversarial perturbations [9]. The adversarial images can cause serious traffic accidents. For example, if the adversarial stop sign is not recognized correctly, the autonomous driving vehicle will not stop. So, it is worthy studying the attack technologies deeply for enhancing the robustness and security of neural networks and autonomous driving. Szegedy et al. [10] first propose the vulnerability of deep learning models in the field of image classification, that is, adding carefully created perturbations can cause the image classifier to misclassify input images with extremely high confidence, while the same perturbation can fool multiple image classifiers. Then, FGSM [11], C&W [12], and Decision-Based Attack [13] are proposed to attack DNNs based image classifiers. To save the time cost in training adversarial examples, Moosavi-Dezfooli et al. [14] and Mopuri et al. [15] propose universal adversarial perturbations against image classifiers. Meanwhile, more and more methods are presented to attack other tasks, for example, Chen et al. [16], Xie et al. [17], and Wei et al. [18] design attacks on object detections.

Object detection is a fundamental task of AI in computer vision and thus has broad applications like traffic sign recognition, face retrieval, and so on.

In addition, the object detection algorithm has made great breakthroughs in recent years. The most popular algorithms can be divided into two categories. One is proposal-based algorithms (R-CNN [19], Fast R-CNN [20], Faster R-CNN [21], etc.). They are two-stage models and need to generate candidate box first, then classify and return the candidate box. The other is regression-based methods such as YOLO [22], YOLO9000 [23], YOLOv3 [24] and SSD [25], which use a convolutional neural network to directly predict the category and location of different objects. Proposal-based models are accurate but slow, regression-based models are faster but the accuracy is lower.

The operation of DNN of AI depends on the training process and it has vulnerability as mentioned above. The object detection models are also vulnerable to adversarial examples. At present, some attack methods against object detection models have been proposed. Xie et al. [17] propose optimization-based Dense Adversary Generation (DAG) and successfully attack Faster R-CNN. Wei et al. [18] propose Unified and Efficient Adversary (UEA) to attack both proposal-based models and regression-based models. Xin et al. [26] propose a DPATCH which is added to the top left corner of the input images to deceive object detectors. However, the adversarial examples generated by DAG and UEA are image-dependent, so it will spend a lot of time training adversarial examples, and DPATH is so obvious for defenders to find out that it is a modified image.

To address these issues, in this paper, we propose to use the transferability of universal perturbations generated against image classification tasks to attack both proposal-based and regression-based object detection models. We leverage generic representations learned by VGG16 [27], VGG19 and Densenet169 [28] and a generator trained on ImageNet dataset to construct transferable universal adversarial perturbations. Then add the universal perturbations by resizing and pile-up to attack object detectors.

This paper is an extended version of a previous conference paper [29]. In our previous work, we used ResNet architecture generator to train universal perturbations against the VGG16 model and used them to attack YOLOv3 and Faster R-CNN detectors in order to verify the transferability of the universal perturbations, and we used 40 images to evaluate the attack effect. Compared to the conference version, we change the quantity and architecture of generators, the structure and depth of the target models used in the training process, and the test dataset, etc. In summary, the main contributions (novel contributions highlighted in bold) of this work can be summarized as follows:

- We discover the cross-task, cross-model, and cross-dataset transferability of universal perturbations and use them to attack object detection systems. Universal perturbations trained against classification models can be used to attack object detection models.
- We succeed in implementing cross-task, cross-model, and cross-dataset black-box attacks using the transferability of universal perturbations.
- We use two different methods, resizing and pile-up, to solve the problem of perturbations and target images size mismatch, successfully attack both proposal-based and regression-based object detectors.
- **We explore the attack effect of the universal perturbations generated by different architecture generators like ResNet and recursive U-Net.**
- **We explore the attack effect of universal perturbations generated against different structure target models like VGG model and Densenet model and target models with the same structure but not the same depth like VGG16 and VGG19.**
- **We significantly increase the number of target images to test the attack effect of the universal perturbations and replace a more appropriate measurement.**

The rest of our paper is organized as follows: we introduce some related works for object detection models and adversarial attack methods in Section 2. In Section 3, we propose our approaches to generate universal perturbations and attack object detectors. After that, the setup and results of our experiments are presented in Section 4. Finally, we draw the conclusions of our entire work.

2. Related Work

In some aspects, we usually don't know the target model's architecture, hyper-parameters, and dataset used during training, so we hope to find a

universal perturbation that can fool object detectors. In addition, We not only hope that we can generate perturbations fast but also hope that the generated perturbations have good attack effects. The related work comes from three aspects: object detection models, universal perturbations, and adversarial attacks for object detectors.

2.1. Object Detection Models

Object detection is an important branch of computer vision. It is widely used in intelligent driving, traffic sign recognition, face retrieval, and behavior recognition. In simple terms, object detection is object recognition and object location. At present, object detection models are mainly divided into two categories: one-stage models like YOLOv3 and SSD, two-stage models like R-CNN, Fast R-CNN, and Faster R-CNN.

2.1.1. Faster R-CNN

In R-CNN, the selective search is used to pre-extract a series of candidate regions that are more likely to be objects. Then use CNN to extract features on these candidate regions for detection. R-CNN needs to extract features for each region proposal, which takes too much time. Fast R-CNN adds RoI pooling layers to the network and uses a multi-task loss function to speed up the detection speed. Faster R-CNN uses region proposal network instead of selective search, and the entire network can share the feature information extracted by the convolutional neural network, which saves computational cost and solves the problem that the Fast R-CNN algorithm generates candidate frames in a slow way.

2.1.2. YOLOv3

YOLO means "you only look once", that is, object location and classification are completed in one step. YOLO uses the entire graph as the input of the network and directly returns the position of the bounding box and its category at the output layer. YOLOv2 adds batch normalization to the network and removes the fully connected layer, improving the accuracy of recognition. YOLOv3 uses multi-scale prediction to improve the detection accuracy of small objects.

2.2. Universal perturbations

2.2.1. Universal Adversarial perturbations

Moosavi-Dezfooli et al. [14] first propose the concept of universal perturbations. They have shown that adding a fixed-size general perturbation to

the original image can have a deceptive effect on the pre-trained image classifier. However, this method actually superimposes multiple image-dependent perturbations and then crops them to obtain universal perturbations. During the perturbation generation, the original image still needs to be used as input, so their method cannot be used in a black-box environment.

2.2.2. Fast Feature Fool

Mopuri et al. [15] propose a method that do not rely on the original images to generate perturbations. They add perturbations to the input to affect the feature extraction of the next layer, and the cumulative effect will lead to a wrong prediction in the last layer. However, it can't be applied to attack object detectors directly.

2.3. Attacks Against Object Detectors

2.3.1. DAG

Currently, adversarial attacks for object detection models are rare. Xie et al. [17] propose an optimization-based method called Dense Adversary Generation (DAG) to attack object detectors. They choose Faster R-CNN as the target model, and specify the object class on the proposal regions and then generate adversarial examples by optimizing the loss function. Therefore, DAG needs many iterations to generate adversarial examples. In addition, adversarial examples generated using the DAG method are image-dependent.

2.3.2. DPATCH

Xin et al. [26] propose an attack method called DPATCH to cripple the object detectors by adding an adversarial patch in the original images. Firstly, they add a randomly-initialized DPATCH to the image, and they update the pixels of DPATH during back-propagation. DPATCH's attack effect is independent of where it is placed. It can deceive Faster R-CNN and YOLO, but its patch is too obvious to be imperceptible, and it needs a very large number of iterations.

To solve the above problems, we add universal perturbations to object detection datasets in two ways: resizing and pile-up, and use the transferability of universal perturbations to attack both proposal-based Faster R-CNN model and regression-based YOLOv3 model in a black-box environment in this paper.

3. Approaches

3.1. Problem Definition

In this paper, we implement black-box adversarial attacks against object detectors, hoping to successfully attack the object detectors and reduce the mean average precision (mAP). mAP is a measure of the detection result of the object detectors. The higher the value, the better the performance of the detectors. mAP is the average value of APs of various classes of objects in the datasets, and AP is the average value of precision of a class on different recalls. Precision measures the accuracy of the detector, that is, the probability that the actual positive example is detected as a positive example by the detector, and recall measures the recall of the detector, that is, the probability that the positive sample is actually detected as a positive example. mAP can measure the detection effect of all examples in a large dataset, so it is usually used as the detection measure of object detectors. The black-box means that we can only obtain the input and output results of the target models, but not the internal network structure and weights of the target models, the training set used in the training process, the setting of parameters during the training process and other internal information. White-box means that the attacker can obtain all the information of the target models. The differences between black-box attacks and white-box attacks are shown in Table 1:

For the attacks of the black-box models, we cannot use the gradient of the neural network to perform iterative calculations to generate adversarial examples, so we consider using the transferability of perturbations to attack the black-box object detectors. Image-dependent adversarial perturbations consume a lot of time in the calculation process, so in order to reduce the attack time and generate a large number of adversarial examples in a short time, we consider using universal perturbations to complete the attack process. Because the size of the target images is different, the universal perturbations cannot be directly used to attack the object detectors. Therefore, we propose to use two methods: resizing and pile-up, to solve the problem that the size of universal perturbations does not match the size of the target images. In the next, we will introduce the generation process of universal perturbations and how to use the transferability of the universal perturbations to attack object detectors.

Table 1: The differences between black-box attacks and white-box attacks

Attributes	black-box attack	white-box attack
Network structure	Unable to get the network structure of the target model	The network structure of the target model can be obtained
Internal parameters	Unable to get the parameters and weights of the neural network of the target model	The parameters and weights of the neural network of the target model can be obtained
Training dataset	Unable to get the dataset used during the training process of the target model	The dataset used during the training process of the target model can be obtained
Network gradient	Unable to generate adversarial examples by calculating the gradient of the neural network	The adversarial examples can be generated by calculating the gradient of the neural network
Adversarial example generation	Can only obtain the input and output of the target model, and adversarial examples are generated by querying	The internal information of the target model can be obtained, and the adversarial examples can be generated by iterative calculation

3.2. Universal perturbations

Moosavi-Dezfooli et al. [14] discovered the existence of universal perturbations. They used an iterative method to generate adversarial perturbations that misclassified the classifier for each class in the target dataset, and then superimposed the perturbations and clipped the universal perturbations according to the limitation of l_p norm. Therefore, the universal perturbation is a vector that causes all categories to cross their corresponding decision boundaries. The existence of the universal adversarial perturbations reveals the geometric correlation of the classifier's high-dimensional decision boundaries, and it is a great threat to AI. When attackers use universal perturbations to implement adversarial attacks, they usually want to train only one perturbation image to make the classifier misclassify all the classes in the target images. The universal perturbations can make a classifier misclassify all classes without affecting the true distribution of the original images, reducing the time to calculate and generate adversarial examples for each class in the target images. And the universal perturbations have a good generalization ability, which is not only across datasets but also across models, and they are

suitable for adversarial attacks in black-box situations. However, compared with the image-dependent perturbations, the L_∞ norm of the universal perturbations is larger because of their universality. Generally, the algorithm of universal perturbations is shown in Algorithm 1:

Algorithm 1 Calculation process of universal perturbations

Input: adversarial perturbation v, image classifier k, target images distribution X, perturbation limitation ϵ

Output: universal perturbation v

Initial $v \leftarrow 0$
for each $x_i \in X$ **do**
  if $k(x_i + v) = k(x_i)$ **then**
    Calculate the minimum v that makes $x_i + V$ reach the decision boundary
    Update v
  end if
end for
return v

3.3. Target Models

In this section, we choose Faster R-CNN and YOLOv3 as our target models, which are representative detectors of proposal-based (two-stage) models and regression-based (one-stage) models, respectively. Faster R-CNN receives the entire image as the input of a shared convolutional neural network to obtain the feature map. The regional proposal network (RPN) completes the segmentation of the feature map and identifies each foreground and background of each segmented anchor. Then, it corrects the coordination of the anchors belonging to the foreground for the first time. The RoI Pooling Layers select the features corresponding to each RoI on the feature map according to the output of the RPN and set the dimension to a fixed value. Finally, the proposals are classified by a fully connected layer (FC Layer), and the anchors are modified for the second time. The detection process of Faster R-CNN is shown in Figure 1:

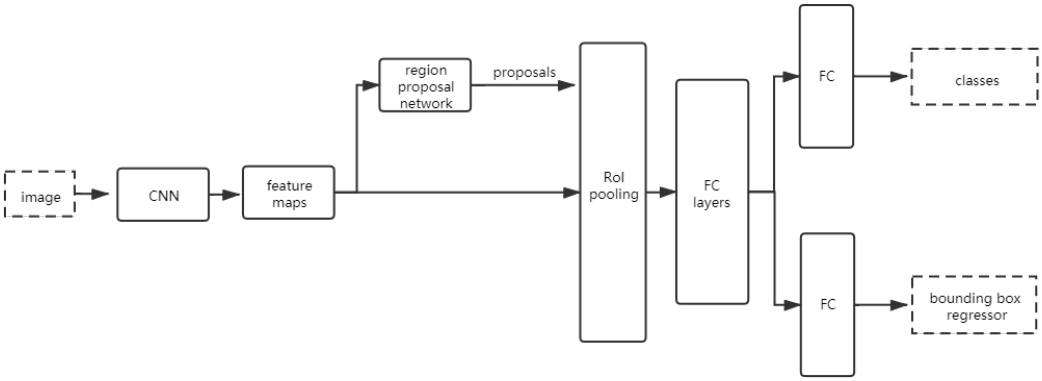

Figure 1: The two-stage detection process of proposal-based Faster R-CNN.

YOLOv3 is different from the two-stage idea of Faster R-CNN that object classification and object location are performed in two stages. It transforms the object detection task into a regression problem. Using the whole image as the input of the network, it directly returns the object frame and the class of the object in multiple positions of the image. YOLOv3 uses Darknet-53 as the backbone network, and it uses the shortcut-connection in the backbone network to ensure that the network can converge normally. It predicts from three different feature scales, feature maps at different scales are fused through upsampling and concatenation operations to improve small objects detection effect. The detection process of YOLOv3 is shown in Figure 2:

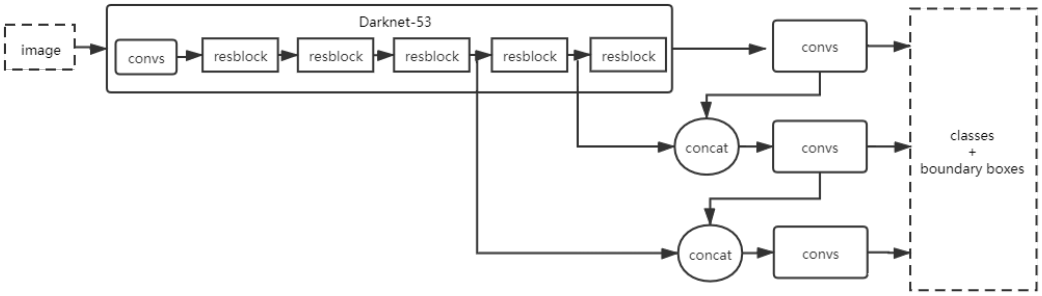

Figure 2: The one-stage detection process of regression-based YOLOv3.

Both Faster R-CNN and YOLOv3 can be used in traffic sign recognition,

face detection, and other scenarios. In this paper, we choose pre-trained Faster R-CNN and pre-trained YOLOv3 as the target models. In order to meet the black-box scene during the attack process, we do not consider obtaining the network structure and parameters of these two pre-trained object detection models, so they are black-box models for us.

3.4. Generation Process of Universal Perturbations

We expect to attack both the proposal-based detector and the regression-based detector at the same time, so in this paper, we propose to use the transferability of the universal perturbations proposed by Poursaeed et al. to attack Faster R-CNN and YOLOv3.

The generation process of the universal perturbations mainly includes a generator G, a loss function $Loss$, and a target classification model F. First, we sample noise z from a random vector that satisfies a uniform distribution. Then, the generator G takes it as input and outputs perturbation p. If the perturbation p is larger than the l_∞ limitation, the perturbation will be scaled at $p* = min(1, \frac{L_{\infty maximum}}{||G(z)||_\infty})$. Then add the scaled perturbation to the original image x and get the adversarial sample $x' = x + p$. The image classifier F predicts the adversarial sample x' and outputs the prediction label l. Finally, the cross-entropy loss function $H(F(x'), l_t)$ is used to measure the distance between the predicted label l and the least-likely label l_t, and the parameters of the generator model are continuously updated by back-propagation so that we can get a universal perturbation generator. The loss function is defined as follows:

$$Loss = log(H(F(x'), l_t)). \tag{1}$$

3.4.1. Generator Architecture

For the choice of the perturbation generators, in this paper, we use ResNet and recursive U-Net as the generator architecture to generate universal perturbations.

The ResNet generator uses a residual module. The shortcut-connection structure in the residual block connects the input layer to the subsequent layers, so the subsequent layers only need to learn the residuals. The convolutional layers and the fully connected layers have the problems of information loss during the information transfer process. After using the shortcut-connection structure, it can deepen the neural network and avoid the decrease of accuracy caused by the increase of the number of neural network layers.

The U-Net generator uses a network that includes upsampling and downsampling. There is a skip-connection in the U-Net network, which realizes feature fusion under different sizes and improves the accuracy of the model.

3.4.2. Perturbation Addition

We add the universal perturbations generated by the generator to the images in Pascal VOC 2007 test set and MS COCO 2014 validation set to attack the object detectors. Since the size of perturbation images is smaller than the size of target images, in order to solve the problem that the universal perturbations cannot be directly applied to the attack object detectors, we use two methods to add the perturbation images to the target images: resizing and pile-up. Resizing means that the perturbation images are adjusted to the size of the target images, and pile-up means that the perturbation images are copied and stacked, and then clipped to the size of the target images. The process is shown in Figure 3.

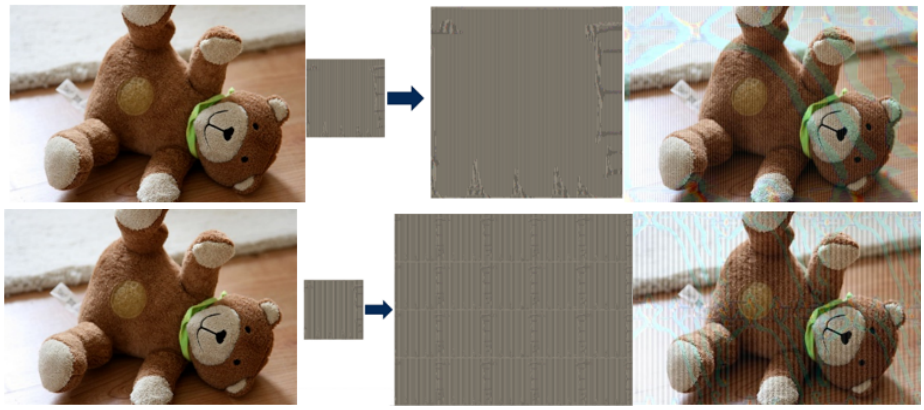

Figure 3: The top row adding the perturbation to the original images in a resizing way, and the bottom row adding the perturbation to the original images in a pile-up way.

The attack process algorithm is shown in Algorithm 2.

Algorithm 2 Attack process of universal perturbations

Input: random noise z, generator G, original image x, perturbation limitation ϵ

Output: two different adversarial examples x'_1 and x'_2

1: The generator generate perturbation: $p = G(z)$
2: Resize the perturbation to the size of the target image: $p \to p_1$
3: Add the perturbation to the original image: $x'_1 = x + p_1$
4: Copy and stack the perturbation, then clip it to the size of the target image: $p \to p_2$
5: Add the perturbation to the original image: $x'_2 = x + p_2$
6: **return** x'_1 and x'_2

3.5. Adversary model

In order to verify the transferability of the universal perturbations, firstly, we use generators of ResNet architecture and recursive U-Net architecture to generate the universal perturbations on the first 10 classes of images in the ImageNet training set. Second, we train the generators with a different number of training set images. In this paper, we use the parameter f to represent the number of images classes used in the training process. When f = 10, it means that the first ten classes in the training set of the ImageNet dataset are used. Then, we let f = 50, f = 100, f = 200, f = 500, and f = 1000 respectively to train the universal perturbation generators of the two architectures, and observe the attack effect of generated universal perturbations against the two object detectors. In addition, in Figure 4, we describe the flowchart of the attack process in detail.

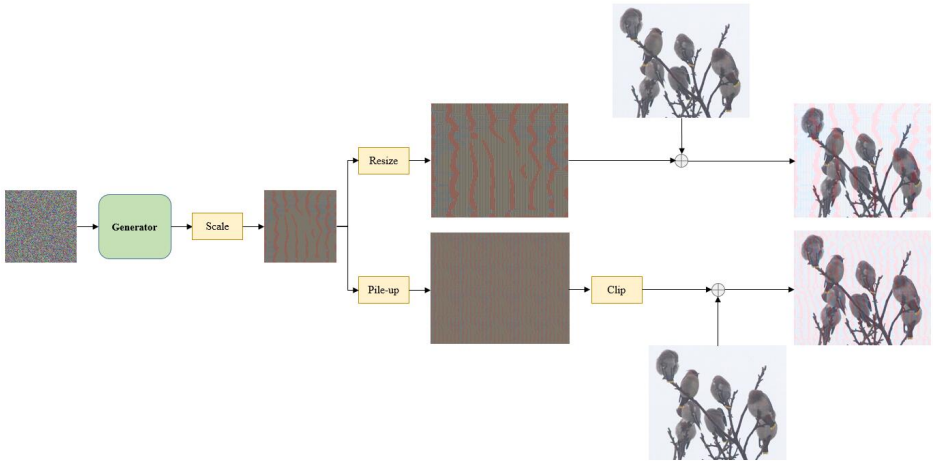

Figure 4: Flowchart of the universal perturbation generation process. We first use a generator to generate universal perturbation and then add the universal perturbation to the target image in two ways. On the one hand, we resize the universal perturbation to the size of the target image and add it to the target image. On the other hand, we pile up the universal perturbation and crop it to the size of the target image, then add it to the target image.

Therefore, the inputs of the target models Faster R-CNN and YOLOv3 are adversarial examples that universal perturbations added by resizing and pile-up, and the outputs are the detection results of the adversarial examples. The universal perturbations can make the above object detectors misclassifies or hides some or all of the objects in the original images, thereby reducing the mAP of the object detectors.

4. Experiments and Evaluation

We conduct two sets of experiments against Faster R-CNN and YOLOv3. In the next subsections, we will show our experimental results and evaluate the proposed methods.

4.1. Experiments

4.1.1. Datasets

We use ILSVRC 2012, Pascal VOC 2007, and MS COCO 2014 datasets for our experiments. ImageNet Large Scale Visual Recognition Challenge (ILSVRC) evaluates algorithms for object detection and image classification

on a large scale. ILSVRC 2012 dataset is a subset of the ImageNet dataset, which is a large scale hierarchical image database. MS COCO dataset and PASCAL VOC dataset are large scale objection detection, segmentation, and captioning datasets. In our experiment, we chose the MS COCO 2014 validation set as the attack dataset against YOLOv3 and PASCAL VOC 2007 test set as the attack dataset against Faster R-CNN. PASCAL VOC contains 20 classes of objects, including person, vehicle, animal, accessory, sports, kitchen, food, furniture, electronic, appliance, the outdoor, and the indoor. There are 4952 images in the PASCAL VOC 2007 test set. The size of images in PASCAL VOC is about 500*375 or 375*500, and each image has its own ".xml" format annotation file. MS COCO contains 80 classes of objects, which is more comprehensive than PASCAL VOC. There are 40504 images in MS COCO 2014 validation set, and the size of images in MS COCO is usually larger than that in PASCAL VOC. The annotation of all images is in a ".json" format file.

4.1.2. Generator Architecture

In our experiment, we use neural networks as generators. For the architecture of the generators, we use recursive U-Net [30] and ResNet [31] to explore the effects of different structures in generating perturbations. The U-Net architecture is an encoder-decoder network with ship connections between the encoder and decoder. And ResNet architecture consists of several downsampling layers, residual blocks, and upsampling layers.

4.1.3. Target Models

In our experiments, we choose Densenet169, VGG16, and VGG19 as our based models to generate perturbations because they are powerful and widely used models. And, we choose two representative models: Faster R-CNN and YOLOv3 as our target models. Faster R-CNN and YOLOv3 are the two most widely-used object detectors, and they are representatives of the one-stage and two-stage object detectors, respectively. In the field of autonomous driving, Faster R-CNN and YOLOv3 are usually used for object detection. Especially, Faster R-CNN is trained on the PASCAL VOC 2007 training set and tested on the PASCAL VOC 2007 testing set. YOLOv3 is trained on the MS COCO 2014 training set and tested on the MS COCO 2014 validation set. In addition, we aim at Faster R-CNN and YOLOv3 to verify the transferability of the perturbations.

4.1.4. Experiment Environment

We use the PyTorch 0.4.1 framework on the Linux system to conduct all the experiments. In addition, we perform our experiments across two Nvidia GeForce GTX 1080 GPUs with 8 GB memory each.

4.1.5. Evaluation Metrics

We mainly evaluate the performance of perturbations from two aspects. One is the degradation of the mAP, the other is the norm of the perturbations added to the original images.

The mAP is mean average precision, the meaning of "mean" is to average the APs of each class, and AP is the area under the precision-recall curve. In addition, the precision and the recall are defined as formula 2 and 3 respectively. Generally speaking, the higher the AP value is, the better the detector is. The size of the mAP must be in the range of [0, 1], the bigger the mAP is, the better the performance of the object detector is. This indicator is one of the most important object detection performance assessment methods.

$$precision = \frac{TP}{TP + FP} \tag{2}$$

$$recall = \frac{TP}{TP + FN} \tag{3}$$

We use p-norm to measure the size of the adversarial perturbations, where p-norm is defined as:

$$\|x\|_p = (\sum_{i=1}^{n} |x_i|^p)^{\frac{1}{p}} \tag{4}$$

In this paper, We use L_∞, which is the absolute value of the maximum changing value of the pixel in the adversarial examples, and use mAP degradation amount between original images and adversarial examples to evaluate our proposed method.

4.2. Results

4.2.1. Results on YOLOv3

We use ResNet and recursive U-Net as the generator architecture. Most of the time is spent in training universal perturbations. The training time varies according to the number of images used in the training process. It varies from two hours to seven days. When we complete the training process

of universal perturbations, we add them to the target images to generate adversarial examples, which takes 0.4 seconds to generate each adversarial example on average. Firstly, we resize and pile up the perturbation generated using the first ten classes of the ImageNet dataset to the size of the input images. And during the training process of the perturbation generators, we set $L_\infty = 20$.

In the next experiment in the case of $L_\infty = 20$, we use the first 50, first 100, first 200, first 500, and 1000 classes of the ImageNet dataset respectively to train the perturbation generators. Subsequently, we perform resizing operation and pile-up operation on the generated perturbations to add them to the input images. And the attack effect is shown in Table 2 and Table 3. The perturbation parameter f represents the number of classes of images used in the generator training process. Figure 5 depicts the perturbations added to different input images.

Table 2: mAP degradation by ResNet generated universal perturbations attack effect against YOLOv3

	Densenet169		VGG16		VGG19	
perturbation parameter(f)	resize	pile-up	resize	pile-up	resize	pile-up
f=10	**22.1**%	29.8%	22.4%	23.2%	**22.4**%	23.7%
f=50	21.8%	**30.9**%	21.6%	**28.3**%	12.7%	13.0%
f=100	19.6%	29.8%	**23.4**%	26.9%	19.4%	27.1%
f=200	20.5%	27.9%	12.9%	13.4%	13.0%	13.9%
f=500	17.9%	28.6%	21.9%	25.3%	19.0%	23.2%
f=1000	12.1%	19.1%	17.5%	25.4%	15.1%	**27.8**%

Table 3: mAP degradation by Recursive U-Net generated universal perturbations attack effect against YOLOv3

	Densenet169		VGG16		VGG19	
perturbation parameter(f)	resize	pile-up	resize	pile-up	resize	pile-up
f=10	19.3%	17.9%	19.3%	17.9%	19.1%	17.7%
f=50	19.3%	17.9%	19.3%	17.9%	19.1%	17.7%
f=100	19.3%	17.9%	19.3%	17.9%	19.1%	17.7%
f=200	19.3%	17.9%	19.3%	17.9%	19.1%	17.7%
f=500	19.3%	17.9%	19.3%	17.9%	19.1%	17.7%
f=1000	19.3%	17.9%	19.3%	17.9%	19.1%	17.7%

Table 2 and Table 3 show the quantitative attacking performance of the transferability of universal perturbations on MS COCO 2014 validation set. We use YOLOv3 as the target object detector here, and the original mAP is 40.9%. From Table 2, we can see the attack performance of universal perturbations generated by ResNet generator is different when the generator is trained on different numbers of classes of images. When we use Densenet169 as the target model to train the generator on the first 50 classes of images in ImageNet and add the perturbations to MS COCO validation set through pile-up, we can reduce the mAP by 30.9%, and at this time, the mAP of YOLOv3 is only 10%. At the same time we found an interesting phenomenon, from Table 3 we can see that when we use recursive U-Net network as the generator, the attack performance of universal perturbations are the same, although the number of classes of images used during the training process of the generator is not the same, we speculate that this is due to the recursive architecture of the generator. It can be seen from Figure 5 that using the transferability of universal perturbations to attack YOLOv3 can make it misclassify or hide the objects in the original images. The left column is original images, and we can see that all the images are detected correctly. The middle column is adversarial examples with resized perturbations, "teddy bear" is misclassified as "dog", "bed" and "bird" are hidden. The right column is adversarial examples with pile-up perturbations, "teddy bear", "bed", and "bird" are hidden. From the results, we can see that our attack method is more suitable for object hiding.

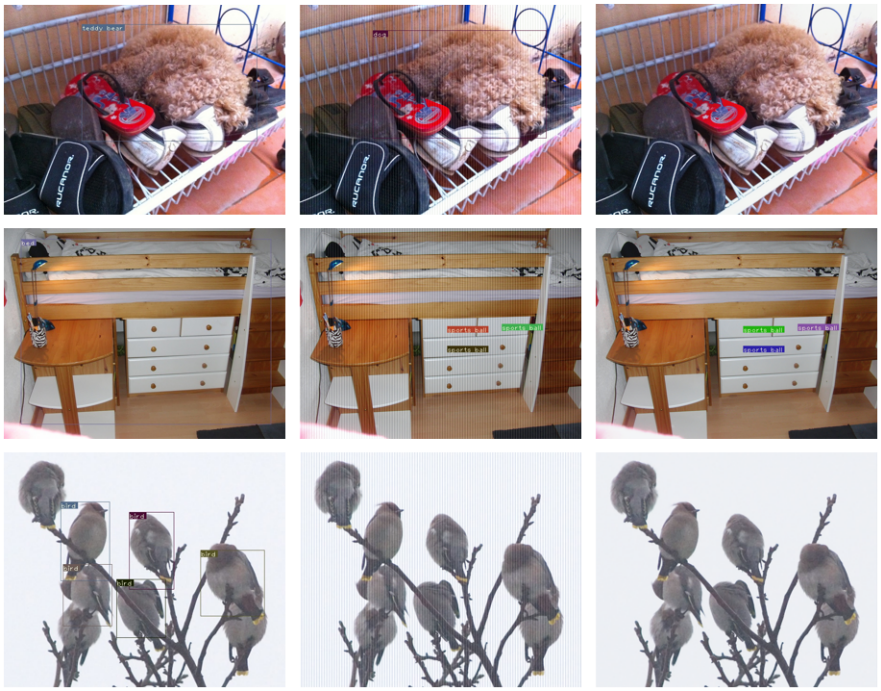

Figure 5: Attacks against YOLOv3 model when $L_\infty = 20$. The left column is original images, the middle column is adversarial examples with resized perturbations, and the right column is adversarial examples with pile-up perturbations.

4.2.2. Results on Faster R-CNN

We also consider ResNet and recursive U-Net generator to attack Faster R-CNN. We attack Faster R-CNN on Pascal VOC 2007 test set, and the generation process of the universal perturbations is the same as the attack on YOLOv3. Table 4 and Table 5 show the attack effect and Figure 6 depicts the perturbations added to different input images.

Table 4: mAP degradation by ResNet generated universal perturbations attack effect against Faster R-CNN

	Densenet169		VGG16		VGG19	
perturbation parameter(f)	resize	pile-up	resize	pile-up	resize	pile-up
f=10	28.0%	42.9%	28.2%	38.3%	**33.2**%	**38.6**%
f=50	29.4%	**45.4**%	26.9%	42.1%	23.5%	35.5%
f=100	**31.1**%	44.3%	**29.8**%	**43.2**%	25.3%	37.5%
f=200	30.8%	40.9%	24.2%	36.2%	23.7%	36.5%
f=500	25.2%	40.9%	27.8%	38.8%	28.5%	33.3%
f=1000	14.0%	21.1%	26.0%	35.1%	21.6%	31.6%

Table 5: mAP degradation by Recursive U-Net generated universal perturbations attack effect against Faster R-CNN

	Densenet169		VGG16		VGG19	
perturbation parameter(f)	resize	pile-up	resize	pile-up	resize	pile-up
f=10	26.2%	34.7%	26.2%	34.7%	26.4%	34.1%
f=50	26.2%	34.7%	26.2%	34.7%	26.4%	34.1%
f=100	26.2%	34.7%	26.2%	34.7%	26.4%	34.1%
f=200	26.2%	34.7%	26.2%	34.7%	26.4%	34.1%
f=500	26.2%	34.7%	26.2%	34.7%	26.4%	34.1%
f=1000	26.2%	34.7%	26.2%	34.7%	26.4%	34.1%

Table 4 and Table 5 show the quantitative attacking performance of the transferability of universal perturbations on PASCAL VOC 2007 test set. We use the Faster R-CNN as target object detector here, and the original mAP is 69.5%. From Table 4 we can see that when we use Densenet169 as target model to train the generators on the first 50 classes of images in ImageNet and add the perturbations to PASCAL VOC 2007 test set through pile-up, we can reduce the mAP by 45.4%, and at this time, the mAP of Faster R-CNN is only 24.1%. From Table 5 we can see that when we use recursive U-Net network as the generator, the attack performance of universal perturbations are the same, although the number of classes of images used during the training process of the generator is not the same, we speculate that this is due to the recursive architecture of the generator. And it can be seen from Figure 6 that using the transferability of universal perturbations

to attack Faster R-CNN can make it misclassify or hide the objects in the original images. The left column is original images, and we can see that all the images are detected correctly. The middle column is adversarial examples with resized perturbations, "bird" and "boat" are hidden. The right column is adversarial examples with pile-up perturbations, "bird" is misclassified as "person", "boat" is hidden. From the results, we can see that our attack method is more suitable for object hiding.

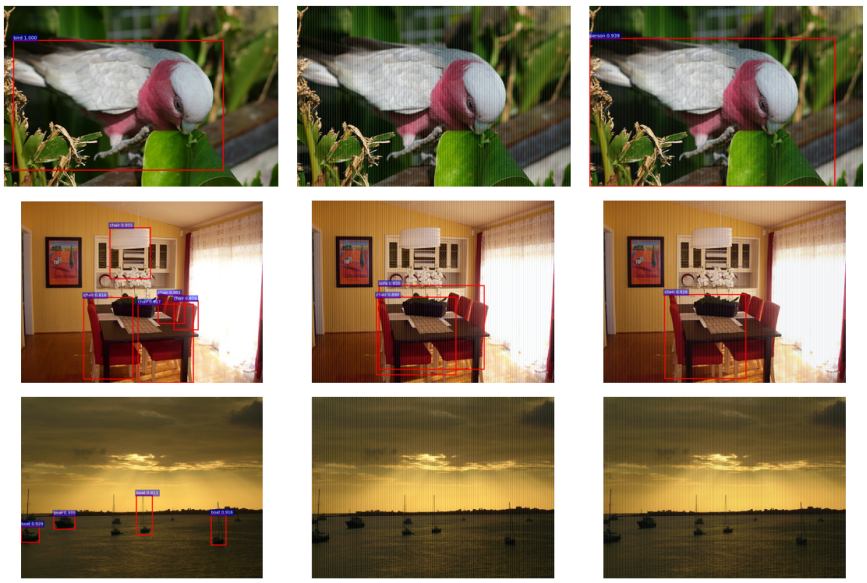

Figure 6: Attacks against Faster R-CNN model when $L_\infty = 20$. The left column is original images, the middle column is adversarial examples with resized perturbations, and the right column is adversarial examples with pile-up perturbations.

Table 6: mAP degradation of YOLOv3 and Faster R-CNN

	Densenet169		VGG16		VGG19	
	resize	pile-up	resize	pile-up	resize	pile-up
YOLOv3	22.1%	**30.9**%	23.4%	28.3%	22.4%	27.8%
Faster R-CNN	31.1%	**45.4**%	29.8%	43.2%	33.2%	38.6%

In this section, we attack YOLOv3 and Faster R-CNN. As can be seen from Table 6, when we use Densenet169 as the target model to train universal perturbation generators and add the universal perturbations to the target

images through pile-up, we can get the strongest attack effect. Universal perturbations in our method are more suitable for large size images. The mAP degradation has verified the powerful attacking ability of the transferability of universal perturbations both against the proposal-based detector (Faster R-CNN) and the regression-based detector (YOLOv3). We believe that if we use the advanced object detectors, the mAP will also degrade.

5. Conclusion

In this paper, we find that the universal perturbations generated against image classification tasks is transferable, and the adversarial attack is extended from image classification to object detection using the transferability of universal perturbations. We successfully solve the problem that universal perturbations cannot be directly applied to attack object detection models through two methods: resizing and pile-up. Then, we attack the two representative object detectors: regression-based YOLOv3 model and proposal-based Faster R-CNN model, and reduce their mAP by 30.9% and 45.4% respectively, demonstrating the great transferability of universal perturbations between different object detector architectures. Our experiments imply that using the transferability of universal perturbations can perform effective attacks under black-box setup, i.e., even without the knowledge of the targeted network's architectures and parameters. It reveals the vulnerability of the object detection models and reveals an interesting topic for future research. The research works and proposed methods in this paper can be powerful vulnerability testing means for object detection in AI. They can enhance the robustness and security of AI, especially the safety of autonomous driving.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant No. 61876019.

References

[1] J. Li, Y. Huang, Y. Wei, S. Lv, Z. Liu, C. Dong, W. Lou, Searchable symmetric encryption with forward search privacy, IEEE Transactions on Dependable and Secure Computing (2019).

[2] Z. Guan, Y. Zhang, L. Zhu, L. Wu, S. Yu, Effect: An efficient flexible privacy-preserving data aggregation scheme with authentication in smart grid, Science China Information Sciences 62 (2019) 32103.

[3] Z. Guan, X. Liu, L. Wu, J. Wu, R. Xu, J. Zhang, Y. Li, Cross-lingual multi-keyword rank search with semantic extension over encrypted data, Information Sciences 514 (2020) 523–540.

[4] W. Rawat, Z. Wang, Deep convolutional neural networks for image classification: A comprehensive review, Neural computation 29 (2017) 2352–2449.

[5] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3431–3440.

[6] Y. Huang, B. Li, Z. Liu, J. Li, S. M. Yiu, T. Baker, B. B. Gupta, Thinoram: Towards practical oblivious data access in fog computing environment, IEEE Transactions on Services Computing (2019).

[7] Z. Liu, B. Li, Y. Huang, J. Li, Y. Xiang, W. Pedrycz, Newmcos: Towards a practical multi-cloud oblivious storage scheme, IEEE Transactions on Knowledge and Data Engineering (2019).

[8] R. Girshick, J. Donahue, T. Darrell, J. Malik, Region-based convolutional networks for accurate object detection and segmentation, IEEE transactions on pattern analysis and machine intelligence 38 (2015) 142–158.

[9] Y. Wang, Y. an Tan, W. Zhang, Y. Zhao, X. Kuang, An adversarial attack on dnn-based black-box object detectors, Journal of Network and Computer Applications 161 (2020) 102634.

[10] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, arXiv preprint arXiv:1312.6199 (2013).

[11] I. J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, arXiv preprint arXiv:1412.6572 (2014).

[12] N. Carlini, D. Wagner, Towards evaluating the robustness of neural networks, in: 2017 IEEE Symposium on Security and Privacy (SP), IEEE, 2017, pp. 39–57.

[13] W. Brendel, J. Rauber, M. Bethge, Decision-based adversarial attacks: Reliable attacks against black-box machine learning models, arXiv preprint arXiv:1712.04248 (2017).

[14] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, Universal adversarial perturbations, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1765–1773.

[15] K. R. Mopuri, U. Garg, R. V. Babu, Fast feature fool: A data independent approach to universal adversarial perturbations (2017).

[16] S.-T. Chen, C. Cornelius, J. Martin, D. H. P. Chau, Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2018, pp. 52–68.

[17] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, A. Yuille, Adversarial examples for semantic segmentation and object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1369–1378.

[18] X. Wei, S. Liang, X. Cao, J. Zhu, Transferable adversarial attacks for image and video object detection, arXiv preprint arXiv:1811.12641 (2018).

[19] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587.

[20] R. Girshick, Fast r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448.

[21] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, in: Advances in neural information processing systems, 2015, pp. 91–99.

[22] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[23] J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7263–7271.

[24] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, arXiv preprint arXiv:1804.02767 (2018).

[25] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, Ssd: Single shot multibox detector, in: European conference on computer vision, Springer, 2016, pp. 21–37.

[26] X. Liu, H. Yang, Z. Liu, L. Song, H. Li, Y. Chen, Dpatch: An adversarial patch attack on object detectors, arXiv preprint arXiv:1806.02299 (2018).

[27] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).

[28] G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.

[29] Y. Zhao, K. Wang, Y. Xue, Q. Zhang, X. Zhang, An universal perturbation generator for black-box attacks against object detectors, in: M. Qiu (Ed.), Smart Computing and Communication, Springer International Publishing, Cham, 2019, pp. 63–72.

[30] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical image computing and computer-assisted intervention, Springer, 2015, pp. 234–241.

[31] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

Quanxin Zhang received the Ph.D. degree from the School of Computer Science, Beijing Institute of Technology, in 2003. From 2011 to 2012, he was a Visiting Scholar with the University of Connecticut. He is currently a Faculty Member with the School of Computer Science, Beijing Institute of Technology. His research interests include mobile computing and neural network security.

Yuhang Zhao received the B.S. degree from the School of Electrical and Information, Northeast Agricultural University in 2019, and now is a postgraduate student majoring in School of Computer Science, Beijing Institute of Technology. His research interest is neural network security.

Yajie Wang received the master's degree of computer and information systems in France in 2018, and now he is a Ph.D. candidate in the School of Computer Science, Beijing Institute of Technology. His main research interests are adversarial example generation under complex black-box conditions, the robustness of deep neural networks.

Thar Baker is a Reader in Cloud Engineering in the Department of Computer Science / Faculty of Engineering and Technology at Liverpool John Moores University (LJMU). He has received his PhD in Autonomic Cloud Applications from LJMU in 2010. Dr Baker has published numerous refereed research papers in multidisciplinary research areas including: Cloud Computing, Big Data, Algorithm Design, Intelligent Transportation Systems, and Autonomic Routing. He has been actively involved as a member of editorial board and review committee for a number peer-reviewed international journals, and is on programme committee for a number of international conferences. Dr. Baker was appointed as Expert Evaluator in the European FP7 Connected Communities CONFINE project (2012-2015).

Jian Zhang, Ph.D., professor of college of cyber science, Nankai University, deputy secretary-general of Cybersecurity Association of China (CASC) and vice chairman of competition and evaluation working committee of CASC. He ever served as the executive deputy director of the National Computer Virus Emergency Response Center and vice president of Association of anti-Virus Asia Researchers (AVAR) who has long engaged in research on China's anti-virus work, monitoring and pre-warning and computer emergency technology as well as the combat and prevention on cybercrime. He has published almost 45 technical papers.

Jingjing Hu received the Ph.D. degree in computer science from Beijing Institute of Technology, Beijing, China. She is currently an associate professor in the School of Computer Science, Beijing Institute of Technology. Her research interests are in the areas of service computing, web intelligence, and information security.

 Quanxin Zhang

 Yuhang Zhao

 Yajie Wang

 Thar Baker

Jian Zhang

Jingjing Hu