DSC 180B Checkpoint Report

Students: Ayush More, Amy Nguyen
Mentor: Lily Weng

Introduction

Deep fusion models benefit from multiple input sources as they provide both complementary information. This enriches the model's feature space, and shared information, which compensates for the shortcomings of different sources. Robustness in deep fusion models is primarily associated with shared information as the input sources face varying corruption and it is important for the model to be balanced in its performance regardless of the corruption. Kim, Taewan, and Joydeep Ghosh's [1] work highlights the motivation of their research through showcasing unbalanced robustness in an error-free model for linear fusion data. To address this issue, they provide three different approaches, two of which were training algorithms and the other implemented a structural change to the model's feature fusion. By focusing on autonomous driving, the authors experimented with a deep fusion object detection model, AVOD, and found their methods to be effective in developing robustness. In the current setting, the noise is generated randomly from a Gaussian distribution rather than a targeted attack, which does not account for intentional corruption from third parties, such as hackers. We plan on expanding on their research by incorporating adversarial examples, which are images with noise masked over each pixel that causes a model to make a mistake in labeling. These are slightly perturbed into the training algorithm rather than use images perturbed by random noise. The AVOD model uses two inputs to create its inferences, images and LIDAR point clouds. By creating subsets of adversarial examples from the image data, we will train the model on the corrupted data and evaluate its robustness against single source noise. We believe that this is relevant given the risks that autonomous vehicles pose to pedestrians - it is important that we ensure the inferences and decisions made by the model are robust against corruption, especially if it is intentional from outside threats.

Methods

Many of the techniques that are used for our project stem from Kim, Taewan, and Joydeep Ghosh's [1] work because, as mentioned in the background, we hope to build off of their work by incorporating adversarial examples.

The authors proposed three different methods involved in training a pre-existing object detection model, AVOD: two algorithms implementing their loss function. They then compared these solutions with a defined evaluation metric across varying difficulties of the Car class within the

dataset. Similar to their procedure, we plan on using the same object detection model, feature fusion method, and evaluation metrics. Differently, we plan on training the AVOD model on our proposed adversarial training algorithm and afterwards, comparing our results with Kim, Taewan, and Joydeep Ghosh's [1] work.

Object Detection Model

AVOD (Aggregate View Object Detection) is a neural network that uses point clouds and RGB images to deliver real-time object detection in the form of bounding boxes and labels for objects in an image. It has state of the art results on the KITTI object detection benchmark, making it a great candidate for our baseline model. Using the same setup as Kim, Taewan, and Joydeep Ghosh [1], we will train the model solely on the car class for the object detection tasks and use the feature pyramid network for feature extraction.

Adversarial Training

To train a model against adversarial attacks, we create adversarial examples and include them into the training set. The loss for predicting all the adversarial examples would need to be minimized. Formally, this optimization problem can be written as

$$\underset{\theta}{minimize}\, \frac{1}{|S|} \sum_{x,y \in S} \underset{\hat{x}}{maximize}\, \ell(h_\theta(\hat{x}), y)$$

where *S* represents the input and output pairs and the inner maximization is the same as the previous section.

This optimization problem, also known as the outer minimization problem, can be solved using standard gradient descent. We include further details on developing the adversarial examples below, specifically focusing on the outer minimization problem.

Adversarial Examples

Since an adversarial example's predicted output should fail to match the true label, creating an adversarial example differs from the typical way of training a classifier. The usual way would be to minimize the loss of the input's predicted output to the true output as represented by

$$\underset{\theta}{minimize}\, \ell(h_\theta(x), y)$$

where ℓ is the loss function h_θ is the model, and x is the image.

However, to create an adversarial example, we instead want to maximize the loss. The optimization problem to solve would be

$$\underset{\hat{x}}{maximize}\, \ell(h_\theta(\hat{x}), y)$$

where $\hat{x}$ is the adversarial example we want to maximize the loss of.

The adversarial example is an image x with noise δ added to it. This noise, more formally referred to as perturbation, is a mask of values over each pixel value and is specific to a given image. Rewriting the optimization problem again to include δ, we get

$$\underset{\delta \in \Delta}{maximize}\, \ell(h_\theta(x+\delta), y)$$

where Δ indicates the valid set of perturbations to add. We keep δ within Δ to ensure the perturbed image is still recognizable to the human-eye. Solving this equation yields an adversarial example to use for an untargeted attack.

Set of Valid Perturbations

For our project, we plan to use the l_{inf} norm as our Δ since it's considered a common perturbation set to use.

Finding the best perturbation is an optimization problem. One way to solve it is to use gradient descent to find the lower bound on the optimization objective.

Solving the Optimization Problem - Creating an adversarial example

We will implement the Fast Gradient Sign Method (FGSM) as our primary method of solving for the optimization objective. Gradient ascent is performed on δ to maximize the optimization objective. The update to delta can be represented by this assignment

$$\delta := \delta + \alpha g$$

where α is the step size and g is the gradient.

For FGSM, we take the largest step possible to maximize the loss so that δ is updated to be as large as possible. To ensure δ is still a valid perturbation, δ is constrained to be within $\pm\epsilon$. This means the magnitude of δ is maxed to be ϵ in FGSM. With this constraint, the equation for updating δ in FGSM becomes

$$\delta := \epsilon \cdot \text{sign}(g).$$

This process creates an adversarial example on one image.

For computation speed, we propose using FGSM to produce an adversarial example.
To train a model adversarially, multiple examples need to be created and the overall loss on the prediction of all these images need to be minimized.

Our Training Algorithm

We will implement our adversarial training algorithm through developing adversarial examples from the input sources and optimizing the maximum perturbation added to the data. Specifically,

we plan on perturbing the image input of our model due to difficulties in translating the noise over to the LIDAR input. [2] Although we recognize that we can add perturbation to both input sources, we understand that there would be a different range of perturbations added. For instance, the noise added to the LIDAR input would be of a different magnitude and for our exploration, we plan on only focusing on images.

First, we calculate the best perturbation to add by using FGSM to the image input. Then, we plan on repeating this procedure for a defined percentage of the dataset so that we can populate our training set with these adversarial examples. Ultimately, this should make the model more robust against adversarial attacks in one of its many input sources.

Our optimization problem will be

$$\underset{\theta}{minimize} \frac{1}{|S|} \sum_{x,y \in S} \underset{\delta \in \Delta}{maximize}\, \ell(h_\theta(x+\delta), y)$$

where x_j represents our image input source.

Overall, the pseudocode for our algorithm is the following:

Proposed Algorithm

Repeat:

1. Select mini-batch B of size n_s
2. If the iteration is an odd number:
    a. For every (x_j, y_j) in B:
        i. Find best attack perturbation δ^* on x_{j^*}
            1. $\delta^* = argmax_\delta \ell(f(x_{j^*} + \delta), y)$
3. If the iteration is an even number:
    a. For every (x_j, y_j) in B:
        i. $\ell = \ell(f(x_j), y)$
4. Compute and add the gradient at δ^*
    a. g := g - $\alpha \nabla \ell(f(x + \delta^*), y)$
5. Perform backpropagation to update weights

Certifying Robustness

After training our model adversarially, we would need to formally certify robustness to ensure the model is robust against adversarial examples.

To certify the robustness of our model, we look to Chiang, Ping-yeh, et al.'s Certified Object Detection [3] approach for verifying robustness of object detectors. Their method involves smoothing based on Gaussian medians as opposed to Gaussian means and can ensure model robustness against ℓ_2 norm attacks. Chiang, Ping-yeh, et al. certifies the robustness across the two categories of object detection: bounding-box and label.

For bounding box certification, the authors certify that a box is correct if the IoU (Intersection over Union) between the actual and worse-case box is above a specified threshold. For label certification, a label is certified if the predicted label matches the actual label exactly.

For our project, we plan on passing our robustly trained AVOD model into Chiang, Ping-yeh, et al.'s certification pipeline on the default certification configurations. We plan to continuously improve on the model to reach certification.

Results

After performing inference on a model trained on clean data, we run inference to see the bounding boxes, confidence scores, and IoU scores of all interesting objects in each sample image.

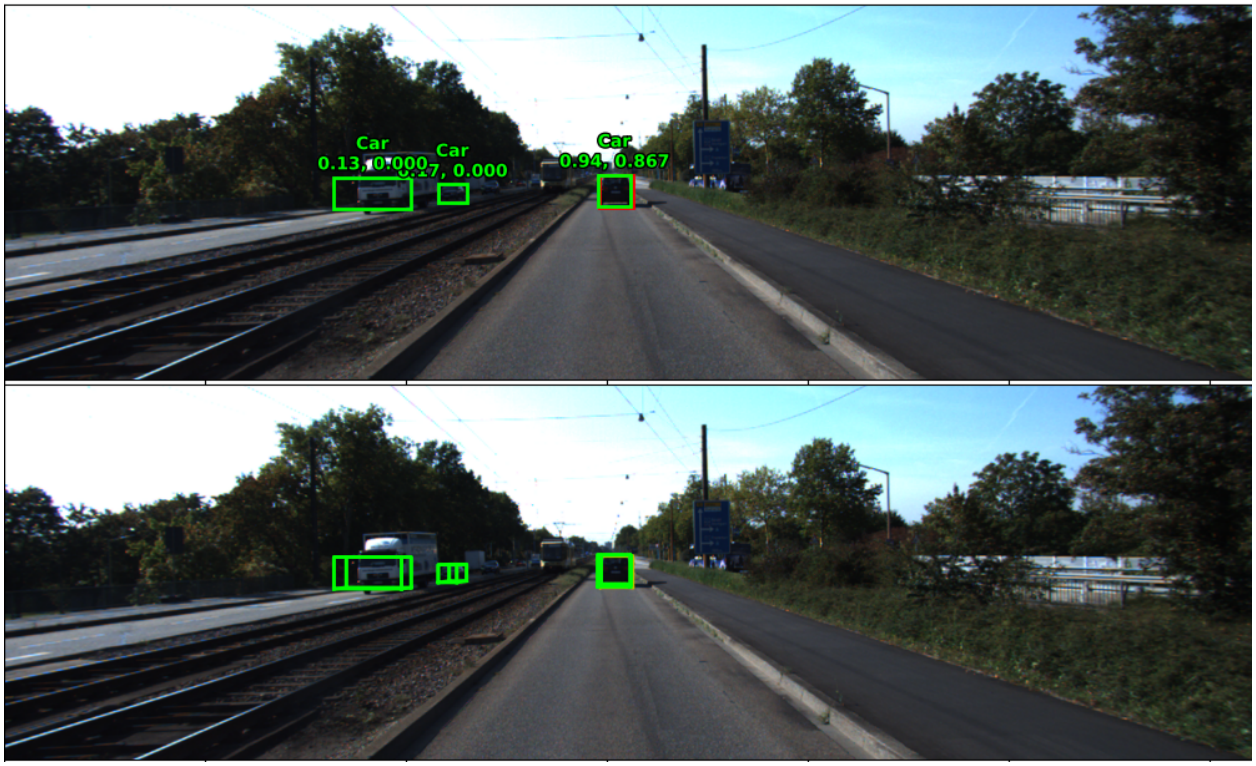

Figure 1: Bounding boxes with confidence and IoU scores for sample 630.

For sample 630, we see that the rightmost car has a IoU score of 0.86 with a confidence of 0.94.

References

[1] Kim, Taewan, and Joydeep Ghosh. "On single source robustness in deep fusion models." arXiv preprint arXiv:1906.04691 (2019). https://arxiv.org/pdf/1906.04691.pdf

[2] Park, Won, and Chen, Qi Alfred. "Crafting Adversarial Examples on 3D Object Detection Sensor Fusion Models." arXiv preprint arXiv:2109.06363 (2021). https://arxiv.org/pdf/2109.06363.pdf

[3] Chiang, Ping-yeh, et al. "Detection as Regression: Certified Object Detection by Median Smoothing." arXiv preprint arXiv:2007.03730 (2020). https://arxiv.org/pdf/2007.03730.pdf

Appendix

Preliminary ideas for the methods used in our project are originally discussed in our project proposal, which can be found here: https://docs.google.com/document/d/1Bgs7Imq6swV6FdzWi7xMLcEBQPIgxPxYTfv1dqjEyYM/edit?usp=sharing