ECE740 Course Project Information

In this semester, we provide 4 different projects. Please discuss with your teammates and select one topic from the 4 problems. As described in our course syllabus, we will evaluate your project following the 4 aspects.

- **Project literature survey & methodology design** (10%), due on Friday at noon (12:00pm), Apr. 1. Note, you are given 4 weeks to complete your project literature survey and methodology design. This is to train your skills to search and study related materials. Note, there is no page limitation on your project literature survey methodology design.
- **10-min Recorded Project presentation** (10%), due on Friday at noon (12:00pm), Apr.22. Your classmates will watch/evaluate your presentation.
- **Project report** (15%), due on Friday at noon (12:00pm), Apr.29. Please prepare your report using ECCV latex template (which can be downloaded from https://eccv2022.ecva.net/submission/call-for-papers/.) and limit your report within 16 pages including figures and tables. Additional pages containing only cited references are allowed.
- **Project implementation** (10%), due on Friday at noon (12:00pm), Apr.29. Reproduction of your method is very important. Please submit your code with clear comments. Our TA will run your code and verify your results. In this regard, please provide a detailed README file in your submission. One may loss the mark if code doesn't work.

Project 1:
Adversarial training for image classification

Xingyu Li (xingyu@ualberta.ca)

1. Problem Description

Adversarial training(AT) incorporates adversarial samples in model optimization and is a very effective method to boost model robustness. AT can be formulated as a MinMax optimization problem [1],

$$\mathcal{L}(\theta) = \mathbb{E}_{(x,y)\sim D}[\max_{\delta\in S} L(f_\theta(x+\delta), y)], \tag{1}$$

Where δ is the adversarial perturbation superimposed on the input, and f_θ denotes the training model parameterized by θ. The inner maximization tries to find adversarial samples that maximize the loss within the allowed permutation range, while the outer minimization tries to minimize the adversarial loss.

Despite its success to improve neural network's robustness against adversarial attacks, adversarial training is observed to hurt standard accuracy badly [2]. Here, adversarial robustness refers to the image classification rate on adversarial samples, and the standard accuracy refers to the image classification rate on clean (original) data. Recently, Raghunathan *et al.* introduce robust self-training (RST) [3] to balance the vanilla (standard) and adversarial loss by a regularization hyper-parameter $\beta > 0$.

$$\mathcal{L}(\theta, \beta) = \mathbb{E}_{(x,y)\sim D}[L(f_\theta(x), y) + \beta \max_{\delta\in S} L(f_\theta(x+\delta), y)]. \tag{2}$$

Another regularized adversarial training strategy, TRADES [4], is proposed to boost model robustness following the Locally-Lipschitz smoothness constraint,

$$\mathcal{L}(\theta, \lambda) = \mathbb{E}_{(x,y)\sim D}[L(f_\theta(x), y) + 1/\beta \cdot \max_{\delta\in S} L(f_\theta(x+\delta), f_\theta(x))], \tag{3}$$

where the hyper-parameter β is used to balance the adversarial robustness and standard accuracy in training. However, we still witness standard accuracy loss in these methods. For instance, TRADES losses about 10% standard accuracy for a 50% adversarial robustness improvement on CIFAR-10 image set.

It should be noted that a big loss of standard performance on clean data is unacceptable in many applications that might cause severe consequences. Instead, the standard performance is more valued than model robustness against the extremely malicious adversarial attacks. Such applications include medical diagnosis, autonomous surgical robotics, etc. This leads to the question: **To what extent can we boost model robustness without sacrificing standard performance?** To answer this question, we formulate the problem as

$$\mathcal{L}(\theta) = \mathbb{E}_{(x,y)\sim D}[\max_{\delta\in S} L(f_\theta(x+\delta), y)] \quad \text{s.t.} \quad \mathbb{E}_{(x,y)\sim D}[L(f_\theta(x), y)] \leq \mathbb{E}_{(x,y)\sim D}[L(f_{\theta_{std}}(x), y)], \tag{4}$$

where $\theta_{std} = \arg\min_\theta \mathbb{E}_{(x,y)\sim D}[L(f_\theta(x), y)]$ represents the model with parameter θ_{std} that yields the minimized standard loss. Comparing the new problem in (4) with previous adversarisal training methods in (1-3), the new

regularization term in (4) explicitly defines the behavior of the model: to improve adversarial robustness without the loss of model's standard performance. In this project, you will investigate novel AT algorithms to tackle the problem described in (4). You may follow any of the following directions to tackle this problem described as follows:

Direction 1: Convert the constrained optimisation problem in (2) into an unconstrained problem using the Karush-Kuhn-Tucker (KKT) approach with a KKT coefficient λ,

$$\mathcal{L}(\theta, \lambda) = \mathbb{E}_{(x,y)\sim D}[\max_{\delta \in S} L(f_\theta(x+\delta), y)] + \lambda\{\mathbb{E}_{(x,y)\sim D}[L(f_\theta(x), y)] - \mathbb{E}_{(x,y)\sim D}[L(f_{\theta_{std}}(x), y)]\}. \tag{5}$$

Therefore, the solution to the dual problem formulated in (5)

$$\min_{\theta} \max_{\lambda, \lambda > 0} \mathcal{L}(\theta, \lambda) \tag{6}$$

is identical to the solution of the primal problem in (4). Note that λ in (5-6) is a trainable parameters, rather than a pre-fixed hyper-parameter. In addition, to solve the problem, we need a pretrained model $f_{\theta_{std}}$, which is fixed in the adversarial training. ***If you try to tackle the problem in this direction, you task is to code an optimization algorithm to solve the problem described in (6)***.

Direction 2: Instead of approaching the solution using the KKT approach, ***you may also try to tackle the problem by proposing heuristic AT algorithms***. For instance, you may want to introduce extra regularization terms in the original AT algorithm in (1) so that the loss of standard performance is punished somehow. The algorithm of RST [3] and TRADES [4] are examples for adding regularization terms in the original AT algorithm in (1).

Direction 3: Different from the above two directions that mix clean data and adversarial samples in training, you may tackle this problem into two phases. The first phase is to detect adversarial samples. For detected adversarial samples, you can either remove the adversarial noise from adversarial samples and pass the purified data to a vanilla trained classfier, or directly train different classifiers for clean data and adversarial data respectively. In this direction, detection of adversarial samples is the key. In addition, adversarial samples denoising is also important if your solution relies on it. You may find many publications in top-tier CV/ML conferences for adversarial sample detection and purification. I encourage you to study SOTA methods and improve their performance.

2. Datasets and Experimental Protocol

MNIST and CIFAR-10 are the widely used datasets to test AT algorithms. Please follow the experimental settings in the study of TRADSES [4] and evaluate the accuracy-robustness performance of your proposed adversarial training algorithm on MNIST and CIFAR-10 datasets. You can refer to the official TRADES codes in github for detail. Note, since this project focuses on improving model's robustness against white-box PGD attacks only, ***your AT algorithm will use PGD samples in training.*** Please report your adversarial robustness and standard accuracy under following settings:

- MNIST setting 1: The CNN model has two convolutional layers, followed by two fully-connected layers. The specific model structure is defined by net_mnist.py in the official TRADES github link. In adversarial training, please set perturbation $\epsilon = 0.1$, perturbation step size $\eta_1 = 0.01$, number of iterations $K = 20$,

learning rate $\eta_2 = 0.01$, batch size $m = 128$, and run 50 epochs on the training dataset. To evaluate the robust performance and standard performance, please apply PGD (white-box) attack with 40 iterations and 0.005 step size.

- MNIST setting 2: The CNN model has four convolutional layers, followed by three fully-connected layers. The specific model structure is defined by small_cnn.py in the official TRADES github link. In adversarial training, please set perturbation $\epsilon = 0.3$, perturbation step size $\eta_1 = 0.01$, number of iterations $K = 40$, learning rate $\eta_2 = 0.01$, batch size $m = 128$, and run 100 epochs on the training dataset. To evaluate the robust error, we apply PGD (white-box) attack with 40 iterations and the step size is 0.01.

- CIFAR-10 setting 1: A ResNet-18 is trained for image classification. The specific model structure is defined by resnet.py in the official TRADES github link. In adversarial training, please set perturbation $\epsilon = 0.031$, perturbation step size $\eta_1 = 0.007$, number of iterations $K = 10$, learning rate $\eta_2 = 0.1$, batch size $m = 128$, and run 100 epochs on the training dataset. To evaluate the robust error, please apply PGD (white-box) attack with 20 iterations and the step size is 0.003.

- CIFAR-10 setting 2: A wide residual network, WRN-34-10, is trained for image classification. The specific model structure is defined by wideresnet.py in the official TRADES github link. In adversarial training, please set perturbation $\epsilon = 0.031$, perturbation step size $\eta_1 = 0.007$, number of iterations $K = 10$, learning rate $\eta_2 = 0.1$, batch size $m = 128$, and run 100 epochs on the training dataset. To evaluate the robust error, please apply PGD (white-box) attack with 20 iterations and the step size is 0.003.

Note,
1. You need GPU resource in this project. Since AT requires to generate advesarial samples for each iteratio during training, it takes time.
2. If the batch size is too large for your GPU, you may reduce it. Please keep other settings unchanged.
3. TRADES official github codes provides a very good code base for this project. You can reuse the code for your convenience.
4. You need to report the adversarial robustness and standard accuracy of RST [3] and TRADES [4] under the same settings. They serve as the comparison baselines in this project,

3. Reference

[1] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2017.

[2] T. Bai, J. Luo, J. Zhao, B. Wen, and Q. Wang, "Recent Advances in Adversarial Training for Adversarial Robustness", 2021.

[3] A. Raghunathan, S. M. Xie, F. Yang, J. Duchi, and P. Liang, "Understanding and mitigating the tradeoff between robustness and accuracy," 2020.

[4] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, "Theoretically principled trade-off between ro- bustness and accuracy," 2019. (TRADES official github: https://github.com/yaodongyu/TRADES)

Project 2:
Weak supervision for semantic segmentation

WSSS4LUAD challenge: https://wsss4luad.grand-challenge.org

This project is all about the WSSS4LUAD challenge. Though the challenge is complete, the dataset and mask ground-truth in WSSS4LUAD provide a good opportunity to hone your deep learning skills towards a practical medical problem.

1. Problem Description by WSSS4LUAD challenge

Histopathology slide is the gold standard of cancer diagnosis. It delivers massive information on tumor microenvironment (TME), which not only plays a vital role in interpreting tumor initiation and progression, but also influences therapeutic effect and prognosis of cancer patients. The crosstalk between different types of tissues are highly related to tumor progression. Therefore, it is urgent to segment and differentiate different tissues for further clinical researches.

Lung cancer is the leading cause of cancer death worldwide . In this challenge, we aim to perform tissue semantic segmentation in H&E stained Whole Slide Image (WSI) for lung adenocarcinoma. The current challenge is that obtaining pixel-level annotations of tissue semantic segmentation is extremely difficult and time-consuming. Inspired by Weakly Supervised Semantic Segmentation (WSSS) in computer vision, we decided to provide only image-level annotations to perform tissue semantic segmentation.

In this challenge, we scanned 67 H&E stained slides from Guangdong Provincial People' Hospital (GDPH) and collected 20 WSIs from The Cancer Genome Atlas (TCGA). Only one WSI was extracted per patient. ***The goal of this challenge is to use only image-level annotations to achieve pixel-level prediction of three common and meaningful tissue types, tumor epithelial tissue, tumor stromal tissue and normal tissue.*** Participants are only given image-level annotations (3-digit one-hot encoding) for machine learning algorithm training, and pixel-level ground truth for validation and testing.

2. Datasets

The WSSS4LUAD dataset is partitioned into training, validation, and testing. You can download the whole dataset from google drive, https://drive.google.com/drive/folders/1qTTTaHAp8HOnvxnKi1RXp-bC7sito9DF. The training set is used to train your weakly-supervised semantic segmentation algorithm. The validation set is used for hypermerameter tuning locally. The test set is for performance comparison and should not be used in training and hyper-parameter tuning. Note, (1) mask images in the mask.zip folder in the above Google drive don't align with any images in either training.zip or test.zip folders. (2) The mask ground-truth of the test set can be downloaded from https://drive.google.com/file/d/1Y5KT9vqSDBg5ec0c1VU8kmzubxF5N2W7/view. Following is the dataset description from the WSSS4LUAD challenge.

Patch Cropping and File Naming Conventions

All the patches were cropped under 10x magnification.

 # Training data

Filename:

- 'patient_ID'+'x_axis'+'y_axis'+'3-digit one-hot labels'.png

Each image (WSI patch) in the training set was cropped from a WSI image at a random anchor point (x, y), with random height (150~300) and width (150~300), where (x, y) is the top-left corner of the patch.

one-hot labels: [Tumor, Stroma, Normal]

 # Validation and testing data

In validation and test sets, we provide patches under two different resolution ranges, large patches and small patches.

- Large patches: (3000~4000)*(3000~4000)
- Small patches: (150~300)*(150~300)

Here are the semantic segmentation labels of each type of tissue in the RGB color space.

- Tumor epithelial tissue : (0, 64, 128), (one-hot label: 0)
- Tumor stromal tisse: (64, 128, 0), (one-hot label: 1)
- Normal tissue: (243, 152, 0), (one-hot label: 2)
- White background: (255, 255, 255), (one-hot label: 3)

3. Evaluation metric

Please use mean Intersection over Union (mIOU) for model evaluation. The white background inside the alveoli will be excluded when calculating mIOU. The background mask in the validation and testing data are provided for quantitative evaluation. Please directly overlay the white background mask on the prediction results.

You can find the challenge leaderboard at https://wsss4luad.grand-challenge.org/evaluation/test-phase/leaderboard/. Please try your best to compete against the best performance of 0.8413 in the leaderboard.

4. Possible Solutions

You may find many publications in top-tier CV/ML conferences to tackle this problem. I encourage you to study SOTA methods and improve their performance on this dataset. Briefly, some studies consider this problem as image classification and takes the classification salience maps or activation maps via XAI techniques as the segmentation results. On the other had, recently studies take advantage of supervised/unsupervised contrastive learning to train a good representation encoder from the training set for downstream segmentation.

Project 3:
Object Tracking in Video with SiamFC model based on MindSpore

Description

Object Tracking is essential in video analysis, and has many applications in surveillance and robotics.

SiamFC model is the SOTA work for object trakcing. It solves the challenge of tracking an arbitrary object in video, where the object is identified solely by a rectangle in the first frame. The goal of this project is understand the SiamFC model and train it yourself in MindSpore and Ascend NPU.

To be able to complete the project, you need to

1. Have a very good understanding of the SiamFC model and codes (in PyTorch).
2. Learn to rewrite/transfer the codes from PyTorch to MindSpore, which takes good understand of both frameworks.
3. Learn how to tune the training hyper-parameters to reach the target accuracy.

Through this project, you will have practical experience of the deep learning frameworks, in-depth knowledge of the SiamFC model and object tracking methods, as well as model training techniques.

Requirements

- The accuracy should be at least the same with accuracy claimed in the paper

Training & Test Dataset website

- Training dataset
  - GOT-10k: http://got-10k.aitestunion.com/downloads
- Test dataset
  - VOT2018: https://www.votchallenge.net/vot2018/dataset.html
- Other related website
  - https://github.com/got-10k/toolkit

Support:

- Free Training Resources (Huawei Ascend Training Server)
- Technical Support on using MindSpore

Some hints

- reference paper
  - https://arxiv.org/pdf/1606.09549.pdf
- Pytorch version
  - https://github.com/huanglianghua/siamfc-pytorch

Project 4:
Handwritten Chinese Character Recognition based on MindSpore

Description

Optical character recognition (OCR) or optical character reader is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo or from subtitle text superimposed on an image. It has been widely used as a form of data entry from printed paper data records – whether passport documents, invoices, bank statements, computerized receipts, business cards, mail, printouts of static-data, or any suitable documentation (refer to wikipedia).

The goal of this project is using MindSpore to implement a Handwritten Chinese Character Recognition model. Actually, it includes two parts, one is Handwritten Chinese Character Detection, and the other is Handwritten Chinese Character Recognition.

To be able to complete the project, you need to

1. Have a very good understanding of the deep learning model and codes.
2. Learn to rewrite/transfer the codes from the original framework (e.g. PyTorh) to MindSpore, which takes good understand of both frameworks.
3. Learn how to tune the training hyper-parameters to reach the target accuracy.

Through this project, you will have practical experience of the deep learning frameworks, in-depth knowledge of the deep learning based OCR, as well as model training techniques.

Requirements

- It can accurately recognize the handwritten characters in standard font: the accuracy in word >=80%

Training & Test Dataset website

- http://www.nlpr.ia.ac.cn/databases/handwriting/Home.html

Support:

- Free Training Resources (Huawei Ascend Training Server)
- Technical Support on using MindSpore

Some hints

- you can choose your own detection & recognition model, also you can use
    - a) CTPN (detection model https://github.com/CrazySummerday/ctpn.pytorch)
    - b) CRNN+CTCLoss (recognition model https://github.com/Holmeyoung/crnn-pytorch)