Homework #1

Deep Learning for Computer Vision

NTU, Fall 2023

FAQ (last updated 10/15 2AM)

1. **Q**: Problem 1, 3 (B) 必須使用 CNN 的架構嗎? 可以用 transformer 嗎?
   A: HW1 是要讓同學熟悉 CNN，所有題目都限制使用 CNN model，禁止使用 transformer
2. **Q**: Problem 1 (A) 中的"from scratch"是指要自己implement CNN中的operation嗎?
   A: 只有限制不能用pre-trained weights，可以使用torch.nn及torchvision.models等
3. **Q**: hw1_*.sh的arguments($1, $2, $3)會是什麼形式?
   A: 會給絕對路徑，且路徑上的directory都會存在
4. **Q**: Problem 3 (A) 中可以使用pre-trained weights嗎?
   A: VGG的部分可以使用pre-trained weights，FCN的部分請同學們自己訓練

FAQ

5. **Q:** Problem 1, 2 生成的csv檔中需要依照 filename 排序嗎? id 與 filename 需有對應關係嗎?
   A: 不需要，只需要確保 filename 和 label 有對應起來即可

6. **Q**: Problem 3 計算 mIoU 時出現 nan (division by zero)
   A: 會出現 nan 應該是因為同學只使用一張圖片計算 mIoU。我們在 evaluate 時不是計算單一圖片的 IoU，我們會使用整個 dataset 中每一個 class 的 TP, FP, FN來計算該 class 的 IoU，然後再平均每個 class 的 IoU 來得到最終用來評分的分數，因此不會有出現 nan 的情況。

FAQ

7. **Q**: Problem 3在inference時input與output的檔名格式?
   A: 在inference時圖片的檔名會與validation data的格式相同，input為xxxx_sat.jpg，output為xxxx_mask.png，請同學按照這個格式將結果存下來。**(原本在42頁上的例子是錯的)**

Outline

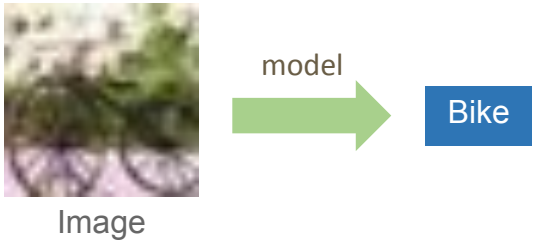
- Problem 1: Image classification (25%)
- Problem 2: Self-supervised pre-training for image classification (40%)
- Problem 3: Semantic segmentation (35%)
- Tools
- Submission
- Homework policy

Outline

- Problem 1: Image classification (25%)
- Problem 2: Self-supervised pre-training for image classification (40%)
- Problem 3: Semantic segmentation (35%)
- Tools
- Submission
- Homework policy

Problem 1: Image Classification

- Image classification - predict a label for each image
  - Input : RGB image
  - Output : classification label



- You need to perform image classification with the following methods:
  - **A.** Train a CNN model **from scratch**
  - **B.** Try alternative models/methods (e.g., fine-tune a pre-trained model)

Dataset

- The dataset consists of 25,000 colored images (32x32 pixels) with 50 classes.
- We split the dataset into
  - **p1_data/train_50/**
    - 22500 images
    - Images are named '{class label}_{***image_id***}.**png**'
  - **p1_data/val_50/**
    - 2500 images
    - Naming rules are the same as p1_data/train_50/
    - You can **NOT** use validation data to train your model in a fully supervised manner,
      but feel free to consider semi-supervised, etc. approaches using both training & validation data

Model Evaluation

- Evaluation metric: Accuracy
  - Accuracy is calculated over all test images.

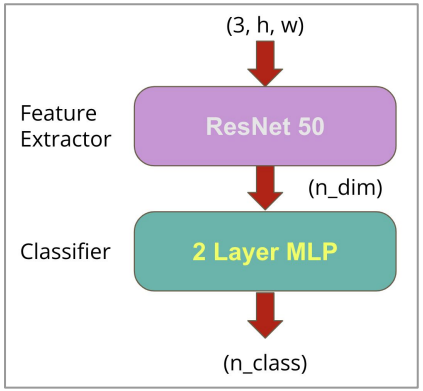
$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

Grading (25%)

- Public Baseline (**5%**) - 2500 validation data
  - simple baseline (2%) - **0.75**
  - strong baseline (3%) - **0.86**
- Private Baseline (**5%**) - 5000 testing data
  - simple baseline (2%)
  - strong baseline (3%)
  - Will be announced after deadline
- You only need to submit one model (either A or B) for the above public/private evaluation.

Grading (25%)

Sample Graph for Q1



Report (**15%**)

1. **(2%)** Draw the network architecture of method A or B.
   - The graph should be brief and clear
   - It would be fine to straight copy the figure from the paper
2. **(1%)** Report accuracy of your models (both A, B) on the validation set.
3. **(2%)** Report your implementation details of model A.
   - Including but not limited to optimizer, loss function, cross validation method, lr scheduling
4. **(3%)** Report your alternative model or method in B, and describe its difference from model A.

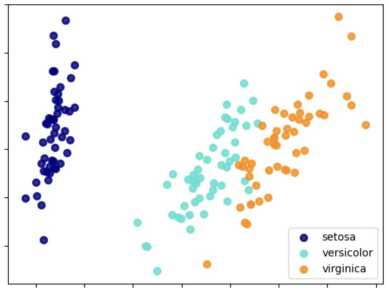
Grading (25%)

Report (**15%**)

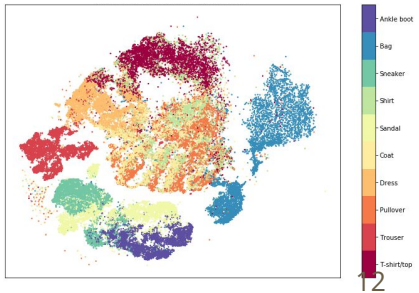
5. **(3%)** Visualize the learned visual representations of **model A** on the **validation set** by implementing **PCA** (Principal Component Analysis) on the output of **the second last layer**. Briefly explain your result of the PCA visualization.

6. **(4%)** Visualize the learned visual representation of **model A**, again on the output of the second last layer, but using **t-SNE** (t-distributed Stochastic Neighbor Embedding) instead. Depict your visualization from **three different epochs** including the first one and the last one. Briefly explain the above results.

PCA



t-SNE



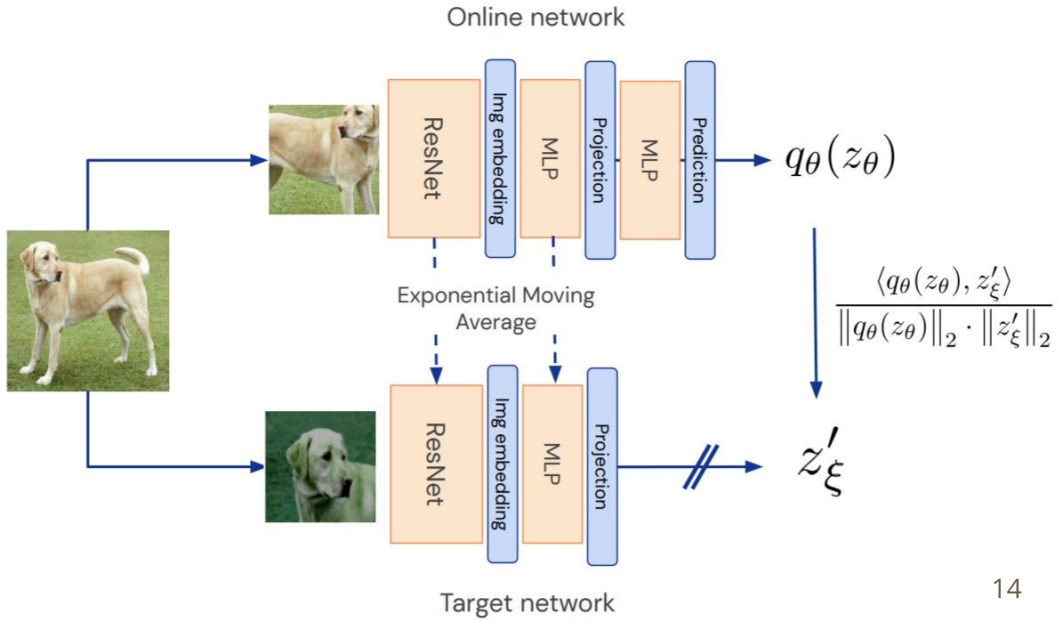
Reference: L. van der Maaten and G. Hinton, "Visualizing Data using t-SNE," Journal of Machine Learning Research, vol. 9, pp. 2579-2605, 2008. [link]

Outline

- Problem 1: Image classification
- Problem 2: Self-supervised pre-training for image classification
- Problem 3: Semantic segmentation
- Tools
- Submission
- Homework policy

Problem 2: Self-Supervised Pre-training for Image Classification

In this problem, you will have to pre-train your own ***ResNet50*** backbone on **Mini-ImageNet** via self-supervised learning methods. After that, you need to conduct image classification on **Office-Home** dataset with different settings to analyze your pre-trained backbone.



Reference: Bootstrap Your own Latent

Grading (40%)

- Grading:
  - Report (25%)
  - Model Performance (15%)

Grading (40%)

Report (25%)

1. (5%) Describe the implementation details of your SSL method for pre-training the ResNet50 backbone. (Include but not limited to the name of the SSL method you used, data augmentation for SSL, learning rate schedule, optimizer, and batch size setting for this pre-training phase)
   - **You have to pre-train the backbone on the Mini-ImageNet without loading default pretrained weights.**
   - **We recommend the following Github repos. (It's easy to understand, use, and train with relatively small batch size.)**
     - **BYOL - LINK (Recommended), Barlow Twins - LINK, etc.**
   - **Since the pre-training phase may take weeks/months to finish under the general SSL setting, we fix the following training setting to reduce the training time. (You MUST follow this setting in the pre-training and fine-tuning phase for fair comparison.)**
     - **Image size: 128*128**
     - **Backbone: ResNet50 (You can choose whether to use the FC layer of ResNet50)**

Grading (40%)

2. (20%) Please conduct the Image classification on **Office-Home** dataset as the downstream task. Also, please complete the following Table, which contains different image classification setting, and **discuss/analyze** the results.
   - **Please report the mean classification accuracy on validation set.**
   - **TAs will run your code to verify the performance of the setting C in the Table below.**
   - **The architecture of the classifier needs to be consistent across all settings.**

Setting	Pre-training (Mini-ImageNet)	Fine-tuning (Office-Home dataset)	Validation accuracy (Office-Home dataset)
A	-	Train full model (backbone + classifier)	*TODO*
B	w/ label (TAs have provided this backbone)	Train full model (backbone + classifier)	*TODO*
C	w/o label (Your SSL pre-trained backbone)	Train full model (backbone + classifier)	*TODO*
D	w/ label (TAs have provided this backbone)	Fix the backbone. Train classifier only	*TODO*
E	w/o label (Your SSL pre-trained backbone)	Fix the backbone. Train classifier only	*TODO*

Grading (40%)

Model Performance (15%)

- Classification accuracy (mean) under **setting C** in Problem 2-2 should be above the baseline score to get points
  - Public baseline (on the validation set):
    - Simple baseline (2.5%): **0.36**
    - Strong baseline (2.5%): **0.40**
  - Private baseline (on the test set):
    - Simple baseline (5%): **TBD**
    - Strong baseline (5%): **TBD**
- TAs will execute your code to check if you pass the private baseline.
- Only TAs have the test data. (testing data is not available for students)

Problem 2: Self-Supervised Pre-training for Image Classification

Provided weights:

- We provide the supervised pre-trained weights, named "**pretrain_model_SL.pt**", of ResNet50 (You can find it in hw1_data/) for you to complete settings B and D. This model is pre-trained on the Mini-ImageNet with class labels as supervision and follows the training setting below.
  - backbone: ResNet50
    - `torchvision.models.resnet50(weights=None)`
  - Image size: 128 * 128
  - Transformation:

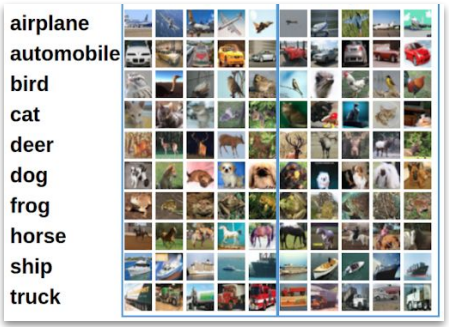
```
TRANSFORM_IMG = transforms.Compose([
    transforms.Resize(128),
    transforms.CenterCrop(128),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406],
                         std=[0.229, 0.224, 0.225] )
    ])
```

Dataset: Mini-ImageNet

- The dataset consists of 38,400 84x84 RGB images of 64 classes.
- Labels are not provided by design

p2_data/mini/

└ train/ # shuffled training images (64 class, each class has 600 images)



Dataset: Office-Home

- The dataset consists of 3,951/406 RGB images in 65 classes for train/valid set.
- In the dataset, you will get

p2_data/office/

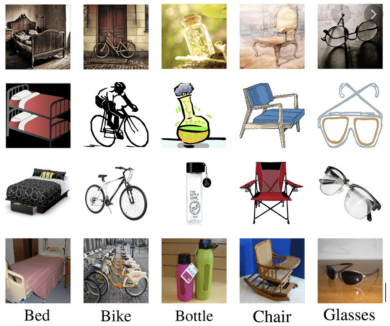
```
train/        # training images directory (65 classes, total 3951 images)
val/          # validation images directory (65 classes, total 406 images)
train.csv     # train image csv file
val.csv       # validation image csv file
```

images are named '{class label}_{image_id}.jpg'

- You **CANNOT** use validation data for training purposes.
- All the testing set files are in the **same** format as the validation set files.

Header in first row: <image_id>, <filename>, <label>

```
id,filename,label
0,13_15.jpg,13
1,27_18.jpg,27
2,48_11.jpg,48
3,0_61.jpg,0
```



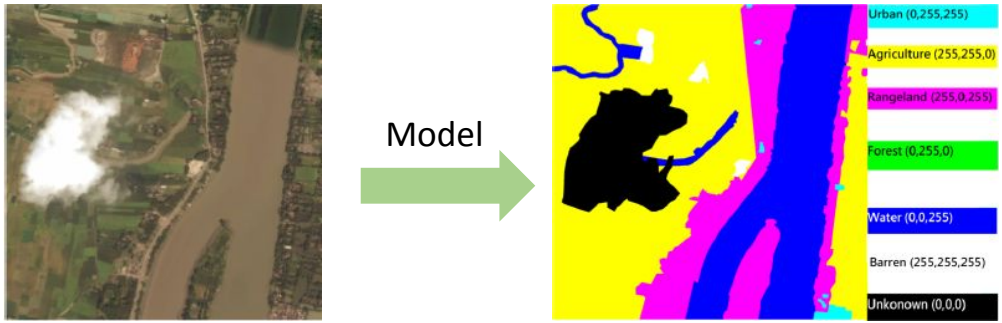
Outline

- Problem 1: Image classification
- Problem 2: Self-supervised pre-training for image classification
- Problem 3: Semantic segmentation
- Tools
- Submission
- Homework policy

Problem 3: Semantic Segmentation

Task Definition

- Semantic segmentation - predict the label for each pixel in an image
  - Input : RGB image
  - Output : Semantic segmentation mask



Image

Segmentation Prediction

Problem 3: Semantic Segmentation

Task Definition

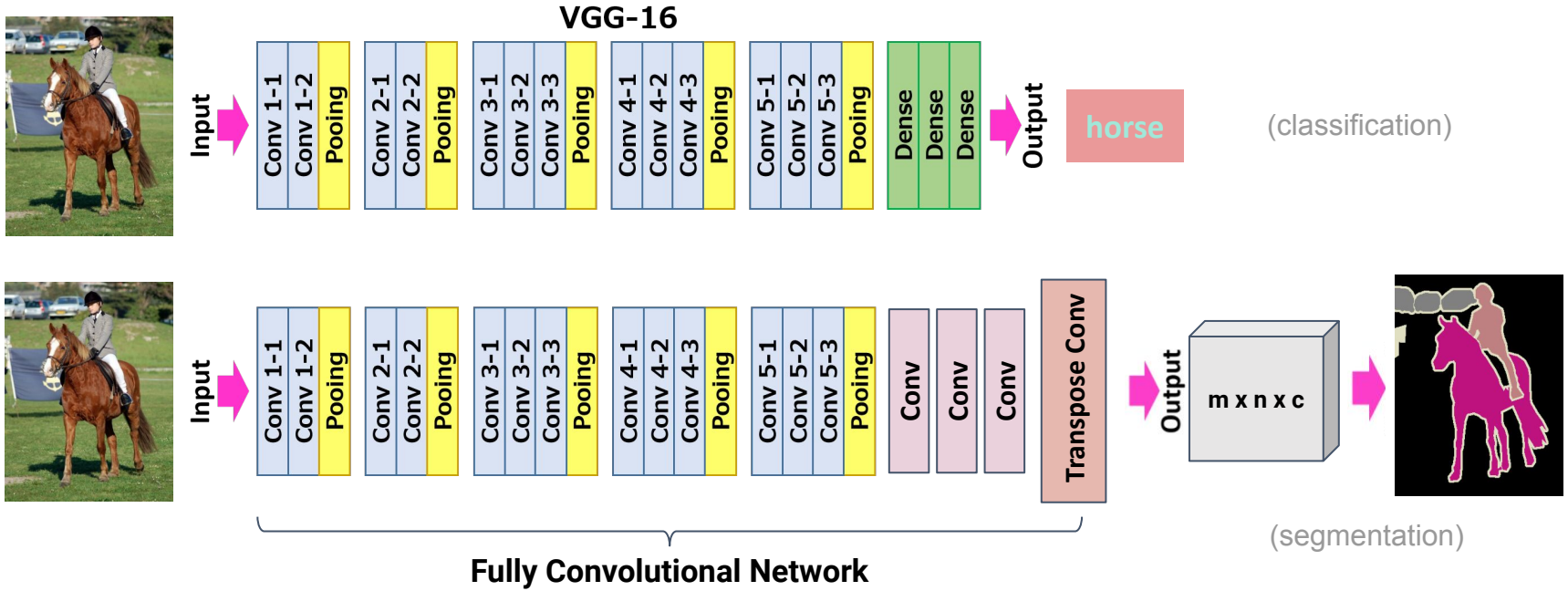
- You need to implement two segmentation models and provide implementation details in the report.
  - A: **VGG16 + FCN32s (baseline model)**

    Implement VGG16-FCN32s model to perform segmentation.
  - B: **An improved model**

    Implement an improved CNN-based model to perform segmentation.
    - You may choose any model different from VGG16-FCN32s, e.g., FCN16s, FCN8s, U-Net, SegNet, etc.)
    - Using pre-trained models is allowed
    - Transformer-based models are not allowed

Semantic Segmentation

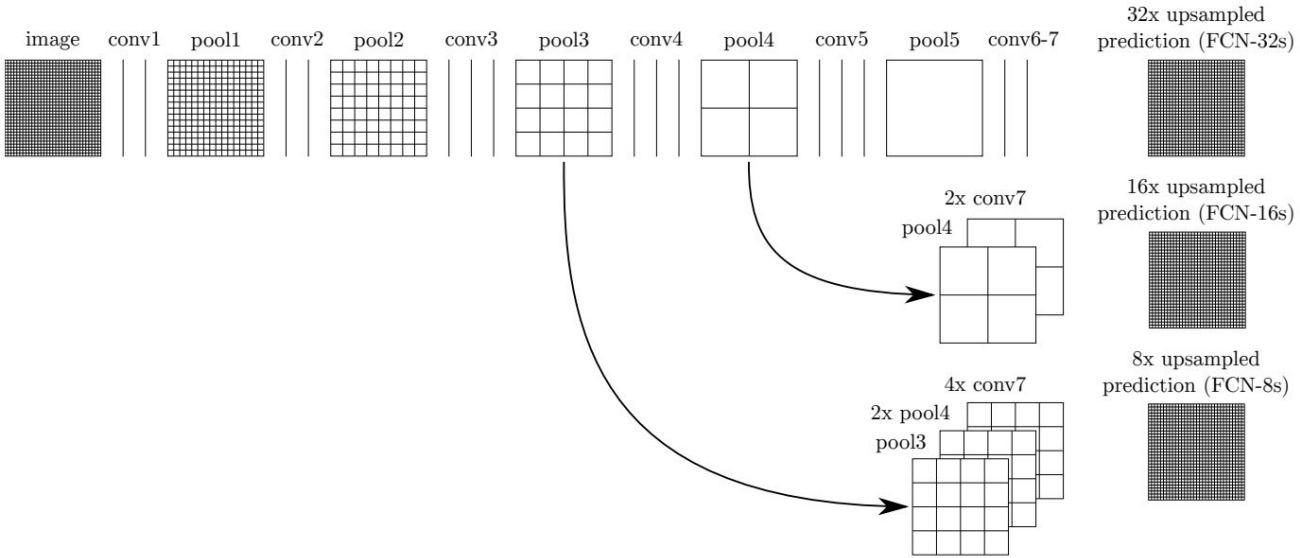
VGG16 + FCN32s



Reference : Long et al., “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015 (Best Paper) [link]

Semantic Segmentation

Fully Convolutional Network - FCN 32s / 16s / 8s

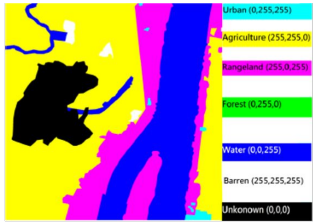


Reference : Long et al., “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015 (Best Paper) [link]

Dataset

Image

CNN



Semantic Segmentation Prediction

- Image size: 512x512
- Mask size: 512x512
- **7** possible class labels

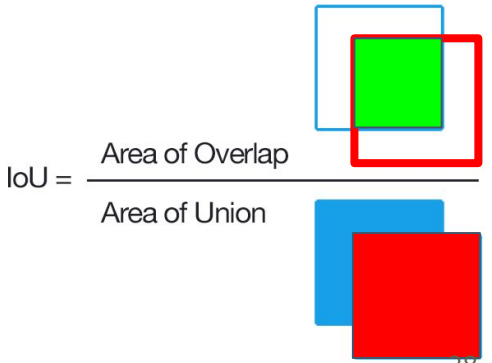
- **p3_data/train/**
  - Contains 2000 image-mask (ground truth) pairs
  - Satellite images are named 'xxxx_sat.**jpg**'
  - Mask images (ground truth) are named 'xxxx_mask.**png**'
- **p3_data/validation/**
  - Contains 257 image-mask pairs
  - Naming rules are the same as p3_data/train/
  - You can **NOT** use validation data to train your model in a fully supervised manner,
    but feel free to consider semi-supervised, etc. approaches using both training & validation data

Metric

- **mean Intersection over Union (mIoU)**
  - For each class, IoU is defined as following:

    $$\textit{True Positive} / (\textit{True Positive} + \textit{False Positive} + \textit{False Negative})$$

  - mean IoU is calculated by averaging over the IoU of all classes **except Unknown(0,0,0)**.
  - mIoU is calculated over **all test images**.



Grading (35%)

Report **(10%)**

1. **(3%)** Draw the network architecture of your VGG16-FCN32s model (model A).
2. **(3%)** Draw the network architecture of the improved model (model B) and explain it differs from your VGG16-FCN32s model.
3. **(1%)** Report mIoUs of two models on the validation set.
4. **(3%)** Show the predicted segmentation mask of “validation/0013_sat.jpg”, “validation/0062_sat.jpg”, “validation/0104_sat.jpg” during the early, middle, and the final stage during the training process of the improved model.
    - Tips: Given n epochs training, you could save the 1st, (n/2)-th, n-th epoch model, and draw the predicted mask by loading these saved models.

Grading (35%)

Model Performance (25%)

- Public Baseline (10%) - 257 validation data
  - simple baseline (5%): **0.69**
  - strong baseline (5%): **0.73**
- Private Baseline (**15%**) - 313 testing data
  - simple baseline (7%)
  - strong baseline (8%)
  - Will be announced after deadline
- You only need to submit one model (either A or B) for the above public/private evaluation.

Outline

- Problem 1: Image classification
- Problem 2: Self-supervised pre-training for image classification
- Problem 3: Semantic segmentation
- Tools
- Submission
- Homework policy

Tool for Dataset

- **Download the dataset**
  - (Option 1) Manually download the dataset here

    https://drive.google.com/file/d/1owVOM3V94bC8v2N8s7n56ciVzVbSzYtd

  - (Option 2) Run the bash script provided in the hw1 repository

    bash ./get_dataset.sh

Tools for Problem 3

- **mIoU**

  python3 mean_iou_evaluate.py -g <ground_truth_directory> -p <prediction_directory>

- **Visualization**

  python3 viz_mask.py <--sat_path xxxx_sat.jpg> <--mask_path xxxx_mask.png>

PyTorch Tutorial

- We provide some resources for someone who is not familiar with **PyTorch**
  - Introduction to PyTorch
  - Basic PyTorch classes
  - Sample colab notebook

Outline

- Problem 1: Image classification
- Problem 2: Self-supervised pre-training for image classification
- Problem 3: Semantic segmentation
- Tools
- Submission
- Homework policy

Submission

- Deadline: **2023/10/17 (Tue.) 23:59 (GMT+8)**
- Click the following link and sign in to your GitHub account to get your submission repository:

  https://classroom.github.com/a/HbabJJaT

  - **You should connect your Github account to the classroom with your student ID**
  - If you cannot find your student ID in the list, please contact us (ntudlcv@gmail.com)
- By default, we will only grade your latest pushed commit **before the deadline** (**NOT** your last submission).
- We will clone the **main** branch of your repository.
- Please send e-mail to TAs if you'd like to submit another version of your repository specifying which commit to grade.

Submission

- Your GitHub repository **DLCV-Fall-2023/hw1-{GitHub_ID}** should include the following files:
  - hw1_1.sh
  - hw1_2.sh
  - hw1_3.sh
  - model checkpoints or hw1_download_ckpt.sh
  - hw1_<studentID>.pdf (e.g. hw1_r09942249.pdf)
  - your python files (Training & Inference code, both required)
- **DO NOT push the dataset to your repo.**

Shell Script

- We will run your shell script from the project root directory.
- You must **NOT** use commands such as **rm**, **sudo**, **CUDA_VISIBLE_DEVICES**, **cp**, **mv**, **mkdir**, **cd**, **pip** or other commands to change the Linux environment.
- In your submitted script, please use the command **python3** to execute your testing python files.
    - For example: python3 test.py $1 $2
- We will execute you code on **Linux** system, so try to make sure you code can be executed on Linux system before submitting your homework.

Shell Script (Problem 1) - hw1_1.sh

- Provide a **script** to test images under the specified directory with your model, and save the classification results in the specified csv file.
- TAs will run your script as shown below:
  - bash hw1_1.sh $1 $2
    - $1: testing images directory with images named 'xxxx.**png**' (e.g., hw1_hiddendata/p1_data/test/)
    - $2: path of output csv file (e.g., output_p1/pred.csv)
- This section must be finished in **10 mins**, otherwise would be considered as a failed run.
- Don't worry about the input/output path, TAs will create appropriate paths for you

Sample CSV Format (Problem 1)

- Predict class labels for all images
  - Output format: .csv file
  - The first row must be: ‘filename,label’ (**NO** space after the comma)
  - You should only output the filename instead of the whole file path (i.e., given the image path ‘hw1_hiddendata/p1_data/test/0_450.png’, you should only output ‘0_450.png’)

```
filename,label
0_450.png,0
0_451.png,0
0_452.png,0
0_453.png,0
0_454.png,0
```

Shell Script (Problem 2) - hw1_2.sh

- Please provide a script to run your model by your setting C, and generate .csv prediction file.
- TA will run your code as shown below
  - bash hw1_2.sh $1 $2 $3
    - $1: path to the images csv file (e.g., hw1_hiddendata/p2_data/office/test.csv)
    - $2: path to the **folder** containing images (e.g., hw1_hiddendata/p2_data/office/test/)
    - $3: path of output **.csv file** (predicted labels) (e.g., output_p2/test_pred.csv)
- Note that you should **NOT** hard code any path in your file or script.
- Your testing code have to be finished in **10 mins.**

id	filename	label
0	0000.jpg	None
1	0001.jpg	None
2	0002.jpg	None

Example of test.csv

id	filename	label
0	0000.jpg	14
1	0001.jpg	2
2	0002.jpg	39

Example of test_pred.csv

Shell Script (Problem 3) - hw1_3.sh

- Provide a **script** to test images under the specified directory with your model, and save the segmentation results as images in the specified output directory.
- TA will run your code as shown below:
  - bash hw1_3.sh $1 $2
    - $1: testing images directory with images named 'xxxx_sat.**jpg**' (e.g., hw1_hiddendata/p3_data/test/)
    - $2: output images directory (e.g., output_p3/pred_dir/)
- You **should** name your output **segmentation mask** as 'xxxx_mask.**png**'
- This section must be finished in **10 mins**, otherwise would be considered as a failed run.

Model Checkpoint

- If your model checkpoints are larger than GitHub's maximum capacity (50 MB), you could download and preprocess (e.g. unzip, tar zxf, etc.) them in hw1_download_ckpt.sh.
  - TAs will run `bash hw1_download_ckpt.sh` prior to any inference if the download script exists, i.e. it is **NOT** necessary to create a blank `hw1_download_ckpt.sh` file.
- Do **NOT** delete your model checkpoints before the TAs release your score and before you have ensured that your score is correct.

Model Checkpoint

- Please use **wget** to download the model checkpoints from cloud drive (e.g. Dropbox) or your working station.
  - You should use **-O argument** to specify the filename of the downloaded checkpoint.
  - Please refer to this Dropbox Guide for a detailed tutorial.
- Google Drive is a widely used cloud drive, so it is allowed to use **gdown** to download your checkpoints from your drive.
  - It is also recommended to use -O argument to specify the filename.
  - Remember to set the permission visible to public, otherwise TAs are unable to grade your submission, resulting in zero point.
  - If you have set the permission correspondingly but failed to download with **gdown** because of Google's policy, TAs will manually download them, no worries!!

Environment

- Ubuntu 20.04.1 LTS
- NVIDIA GeForce RTX 2080 Ti (11 GB)
- GNU bash, version 5.0.17(1)-release
- Python 3.8

Please ensure your scripts can be run under the environment constraint.

Choose your testing batch size wisely to prevent CUDA out of memory error

Packages

These are the only packages TAs will install when running your scripts

- imageio==2.21.2
- matplotlib==3.5.3
- numpy==1.23.1
- Pillow==9.2.0
- scipy==1.9.1
- torch==1.12.1
- torchvision==0.13.1
- gdown, tqdm, glob, yaml
- transformers==4.21.3
- timm==0.6.7
- scikit-learn==1.1.2
- pandas
- einops==0.6.1

If you want to use any other packages, please email TA beforehand.

Packages

- Do not use **imshow()** or **show()** in your code otherwise your code may crash.
- Use **os.path.join** to deal with path as often as possible.
- Only the packages listed above are allowed when TAs run your scripts
  - No points will be given for runtime errors due to dependency or version issues
  - However, we do not care what packages you use during **training**. Feel free to use any tools you might like (e.g., pytorch-lightning, torch-summary, tensorboard, etc.),

Final Check

- Ensure your code can be executed successfully on **Linux** system before your submission.
- Use only **Python3** and **Bash** script conforming to our environment, do not use other languages (e.g., CUDA) and other shell (e.g., zsh, fish) during inference.
  - Use the command `**python3**` to execute your testing python files.
- You must **NOT** use commands such as **sudo, CUDA_VISIBLE_DEVICES** or other commands to interfere with the environment; **any malicious attempt against the environment will lead to zero point in this assignment**.
- You shall **NOT** hardcode any path in your python files or scripts, while the dataset given would be the absolute path to the directory.

Outline

- Problem 1: Image classification
- Problem 2: Self-supervised pre-training for image classification
- Problem 3: Semantic segmentation
- Tools
- Submission
- Homework policy

Deadline and Academic Honesty

- Deadline: **2023/10/17 (Tue.) 23:59 (GMT+8)**
- **Taking any unfair advantages over other class members (or letting anyone do so) is strictly prohibited. Violating university policy would result in F for this course.**
- Students are encouraged to discuss the homework assignments, but you must complete the assignment by yourself. TA will compare the similarity of everyone's homework. **Any form of cheating or plagiarism will not be tolerated and result in an F for students with such misconduct.**.
- Please specify, if any, the **references** for any parts of your HW solution in your report (e.g., your collaborators or the GitHub source code, ChatGPT, Bard, etc.)
- **Using external dataset is forbidden for this homework.**

Late day policy

- We provide a total of **three free late days** for all four homework submissions this semester. After that, late homework will be deducted by **30%** each day.
- If you wish to use your late day quota, or if wish to submit an earlier version, submit this form before **2023/10/20 (Fri.) 23:59 (GMT+8)**. You may resubmit multiple times; TAs will only take the last submission. After this date, you **CAN NOT** change the number of late days to use or the commit hash to grade

https://forms.gle/SMRhZ3kZbzvBz9ou9

Code Modification

- After we grade your assignment, if your code cannot be executed, you have **ONE** chance to make minor modifications to your code. After modifying your code,
  - If we can execute your code, you will receive a **30% penalty** in your model performance score.
  - If we still cannot execute your code, no points will be given.
- TAs will release the log of execution after grading, please check.
  - Email the TAs if something goes wrong in your submission.

How to find help

- Google!
- ask ChatGPT or other LLMs
- Use TA hours (please check course website for time/location)
- It’s highly encouraged to post your questions on NTU COOL discussions page
  - Your questions might benefit others as well
- Contact TAs by e-mail: ntudlcv@gmail.com

Dos and Dont's for the TAs (& Instructor)

- Do NOT send private messages to TAs via Facebook or spam TAs email.
  - TAs are happy to help, but they are not your tutors 24/7.
- TAs will NOT debug for you (e.g., coding, environment, dependencies, etc. issues).
- If you cannot make the TA hours, please email the TAs for an appointment.

Final Reminder

- *Workload for HW1 is heavier than usual this year*
- Please start working on this homework **as early as possible**.
- The training may take a few hours on a GPU or days on CPUs.
- As mentioned in the first class, we **DO NOT** provide any computation resources.
- Violate the aforementioned regulations will incur penalty.
- TAs are here to help. Discussions/Q&As are welcome :)