

National University of Singapore

Department of Mathematics

04/2025 DSA5203 Visual Information Processing and Interpretation Project 3

Goal

The objective of this project is to apply image classification techniques learned in the course. You are required to create an application designed for the categorization of scenes.



Group project.

Each team should comprise no more than two members. Please take the initiative to form your team. If you are unable to find a teammate, you may also choose to work independently.

Implementation option

You have two options for implementing the categorization task. Please choose **one and only one** of the following two approaches.

1. The system based on a Bag-of-Words feature model and a non-linear SVM classifier.
2. The system utilizing a convolutional neural network (CNN).

Academic Integrity

You and your group member are required to independently develop the project. While discussing ideas with classmates from other groups is permitted, code sharing is strictly forbidden. You must not use code from other groups' projects.

Using third-party code for additional refinement is acceptable, provided that you explicitly acknowledge this in your report, clarifying that it is not your original work. Failure to disclose the use of third-party code from online resources in your implementation will be considered plagiarism.

Should you have any doubts regarding the use of such resources, it is imperative to seek clarification beforehand. Your code will undergo examination through a standard code plagiarism detection system. Any code flagged by the system that lacks proper attribution in the report will be subject to further scrutiny. Violating these guidelines constitutes a breach of the Honor Code and will be reported to the university for appropriate disciplinary action.

Training dataset

The training dataset comprises a total of 1,500 images. These images are evenly distributed across 15 classes, with each class containing 100 images. This dataset is freely accessible to you. You may choose to divide it into training and validation datasets when training a classifier.

Remark: When training a classifier, the category labels are quantified as per the table below.

Warning: Do not alter the quantitative labels assigned to the scene categories as specified in the table. Modifying these labels may result in incorrect results when your model is evaluated on testing samples according to the labels listed in the table.

category	label	category	label	category	label
bedroom	1	Coast	2	Forest	3
Highway	4	industrial	5	Inside city	6
kitchen	7	livingroom	8	Mountain	9
Office	10	OpenCountry	11	store	12
Street	13	Suburb	14	TallBuilding	15

Testing dataset for evaluating your result

Your trained classifier will be assessed based on its performance on the test dataset,

which comprises 225 images, with 15 images per category. This dataset will not be accessible to you.

Grading policy

The grade will be determined by the quality of your code, the performance of your model on the test data, and adherence to the submission instructions.

1 The Guideline on BoW+SVM based implementation

Bag-of-words model for scene: Bag of words model is about building feature vector using a histogram of the frequency of visual words, which "vocabulary" is established by clustering a large corpus of local features. The basic procedure is as follows

- Uniformly (or randomly) sampling interesting points
- Computing SIFT feature for each point
- Using K-means clustering to construct codebook (dictionary)
- Based on codebook, constructing histogram of visual words
- Training an one-vs-all SVM-based classifier on training dataset.

Guideline of code

The main function is *scene_recog_bow.py*, where there are at least two functions: **train()** and **test()**:

train(train_data_dir, **kwargs)

Parameter: **train_data_dir**:str the directory of training data.

****kwargs** optional. Other necessary variables with default values.

The **train**(train_data_dir) is the function for training the model. It is supposed to call at least three functions in the training step: **build_vocabulary()**, **get_hist()**, **svm_classify()**. Specifically, **build_vocabulary()** builds the vocabulary from the training data, **get_hist()** represents the training data as the histograms of visual words and **svm_classify()** trains a SVM-based classifier according to the histogram of visual words.

The functions `build_vocabulary()`, `get_hist()`, `svm_classify()` are strongly encouraged to be in separate Python files from `scene_recog_bow.py`. Also, saving the trained SVM classifier with the file "`trained_svm.pkl`"¹.

`test(test_data_dir,trained_svm_dir,**kwargs)`

Parameter: `test_data_dir:str` the directory of test data.

`trained_svm_dir:str` the directory of trained SVM file ("`trained_svm.pkl`").

`**kwargs` optional. Other necessary variables with default values.

`test(train_data_dir)` is the main function for testing the model. It should incorporate at least three steps: (1) processing the incoming test images, (2) loading the pre-trained SVM to do the test and (3) calculate and print the accuracy.

Remark 1: It is a good practice to comment the meaning of inputs and outputs, under the definition of each function.

Remark 2: There are many implementation options to be determined to have optimal performance. Taking bag-of-word for example, How to sample interesting points; how many clusters are used for constructing codebook; are you using linear SVM or SVM with Gaussian kernel. These are left to you to discover.

Forbidden function: You may NOT directly call the routine from existing library or online-source for calculating bag-of-word of images. You are allowed to use the routines including calculating SIFT, k-means, calculating histogram, linear SVM, and SVM with Gaussian kernel. If you are unsure about a function, please ask in advance.

Submissions

You are required to submit the following:

1. *The python package contains all necessary codes.*
2. *Example code for how to run your code on sample image for classification.*
3. *A technical report on the contributions of each team member, the discussion of your method and the implementation tricks.*

Important: Please check the `guideline.py` for an sample for preparing your submission.

¹other type of saved model is also applicable

2 The guideline on CNN-based implementation

CNN-based method: *Convolutional neural network is about training a CNN on the training dataset for classifying scenes. The basic procedure is as follows.*

- *Design and implement an CNN*
- *Splitting the training dataset into training data and validation data*
- *Training the CNN on the training and validation data*

Guideline of code

*Main function is scene_recog_cnn.py. In the scene_recog_cnn.py, there are at least two functions: **train()** and **test()**:*

train(train_data_dir, **kwargs)

Parameter: *train_data_dir:str* the directory of training data.

***kwargs optional. Other necessary variables with default values.*

train(train_data_dir) is main function for training the model. It should involve at least three steps: (1) processing the incoming data, (2) construction the CNN model (saved in model.py) and (3) the complete training procedures. Saving the trained CNN model in the file "trained_cnn.".*

Remark 1: *The suffix of such file depends on your framework. For example, TensorFlow ending with ".ckpt", and Pytorch ending with ".pth".*

test(test_data_dir, trained_cnn_dir, **kwargs)

Parameter: *test_data_dir:str* the directory of test data.

trained_cnn_dir:str the directory of trained CNN model ("trained_cnn.*").

***kwargs optional. Other necessary variables with default values.*

test(train_data_dir) is main function for testing the model. It should involve at least three steps: (1) processing the incoming test images, (2) loading the pre-trained CNN to run the test and (3) calculating and printing out the accuracy.

Remark 2: *There are many implementation options to be determined to have optimal performance. What architecture you will use, what is the size of NN weights, what training technique to be used. These are left to you to discover.*

Forbidden function: *You are allowed to use standard routines in pytorch for building the model and training the model. If you are unsure about a function, please ask in advance.*

Submissions

You are required to submit the following:

1. The python package contains all codes and trained models.
2. Sample code for how to run your model on testing images
3. A technical report on the discussion your method and useful implementation trick that you like to point out.

Submission guidelines

- Please package all your files in a single zip file named by assignment id and IDs of your team (e.g. hw3-u839393a-e9405050b.zip for two-member team or hw3-u0919929b.zip for single-memeber team).
- For each team, only one team member needs to submit the file. The members of the team need to be clarified clearly in both the name of the submitted file, as well as inside the report.
- Upload the zip file to assignment 3 in Canvas. **DO NOT email the files to my email address.** The deadline for submission is **FRIDAY, 25-April-2025**. Any submission after the deadline will lead to some penalty on grade.

If you have any question, please contact me by email: matjh@nus.edu.sg for help.