



Image recognition

Programming Assignment 2

PA2. Image Recognition (100%)

1. Feature extraction (40%)

1. Bag-of-Words (BoW) feature extraction (10%)
2. BoW with Spatial pyramid feature extraction (10%)
3. VGGNet features from VGG13 (10%)
4. VGGNet features from VGG19 (10%)

2. Classifier (30%)

1. Support-Vector-Machine (SVM) classifier (10%)
2. Random Forest (RF) classifier (10%)
3. 2-layer FC layer (10%)

3. Report (30%)

1. Present qualitative and/or quantitative results for image retrieval using 4 image features (10%)
2. Compare the performance of all 12 combinations of feature extraction and classifiers. (20%)

0. Data loader

1. Implement data-loader and load given Caltech 20 datasets.
2. Divide a dataset into a training set and a test set.

Caltech 101

~~New~~ [Caltech256](#) ~~New~~

[\[Description\]](#) [\[Download\]](#) [\[Discussion\]](#) [\[Other Datasets\]](#)



Description

Pictures of objects belonging to 101 categories. About 40 to 800 images per category. Most categories have about 50 images. Collected in September 2003 by Fei-Fei Li, Marco Andreetto, and Marc 'Aurelio Ranzato. The size of each image is roughly 300 x 200 pixels. We have carefully clicked outlines of each object in these pictures, these are included under the 'Annotations.tar'. There is also a matlab script to view the annotations, 'show_annotations.m'.

How to use the dataset

If you are using the Caltech 101 dataset for testing your recognition algorithm you should try and make your results comparable to the results of others. We suggest training and testing on fixed number of pictures and repeating the experiment with different random selections of pictures in order to obtain error bars. Popular number of training images: 1, 3, 5, 10, 15, 20, 30. Popular numbers of testing images: 20, 30. See also the discussion below. When you report your results please keep track of which images you used and which were misclassified. We will soon publish a more detailed experimental protocol that allows you to report those details. See the Discussion section for more details.

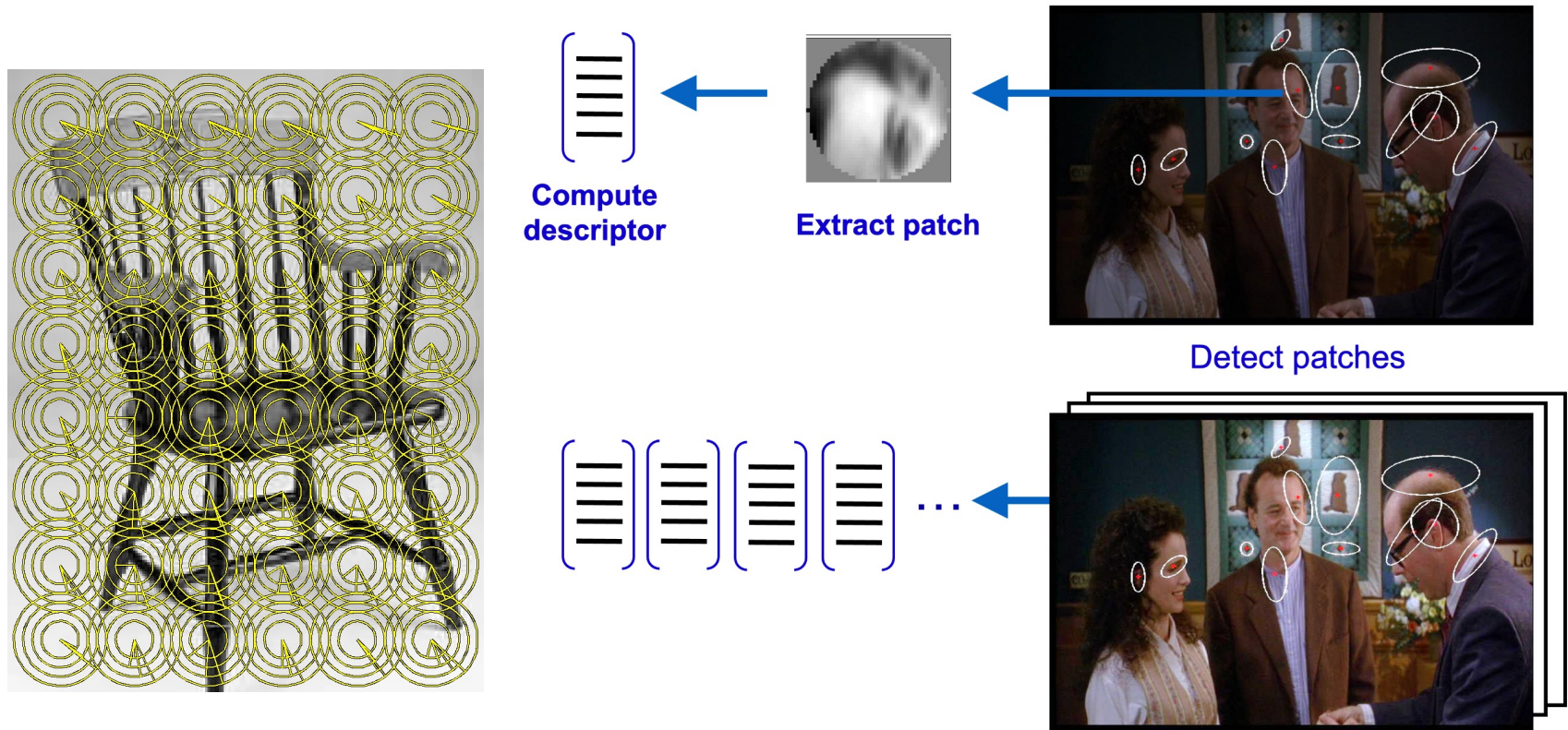
Download

Collection of pictures: [101_ObjectCategories.tar.gz \(131Mbytes\)](#)

Outlines of the objects in the pictures: [1] [Annotations.tar](#) [2] [show_annotation.m](#)

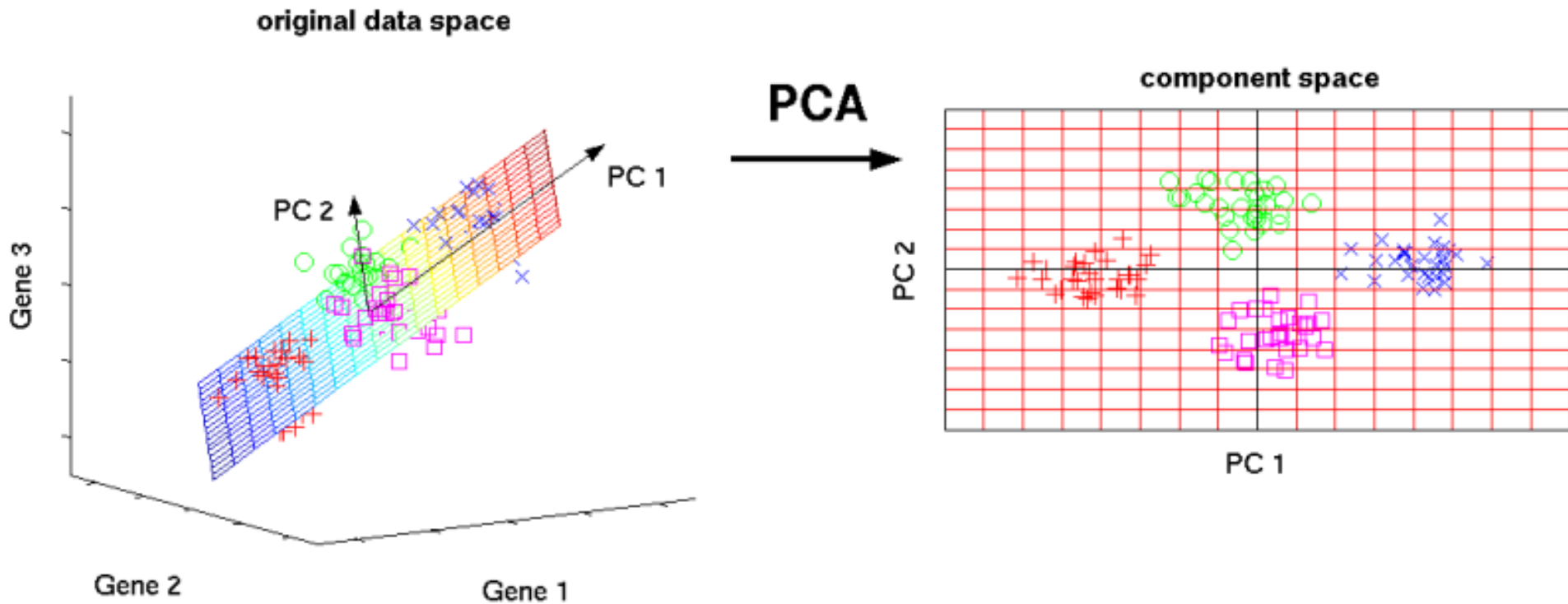
1-(1). Bag-of-Words (BoW) feature extraction

1. Extract Dense SIFT features and descriptors using SIFT algorithm[1]
2. Randomly sample a set of local descriptors from the training set.



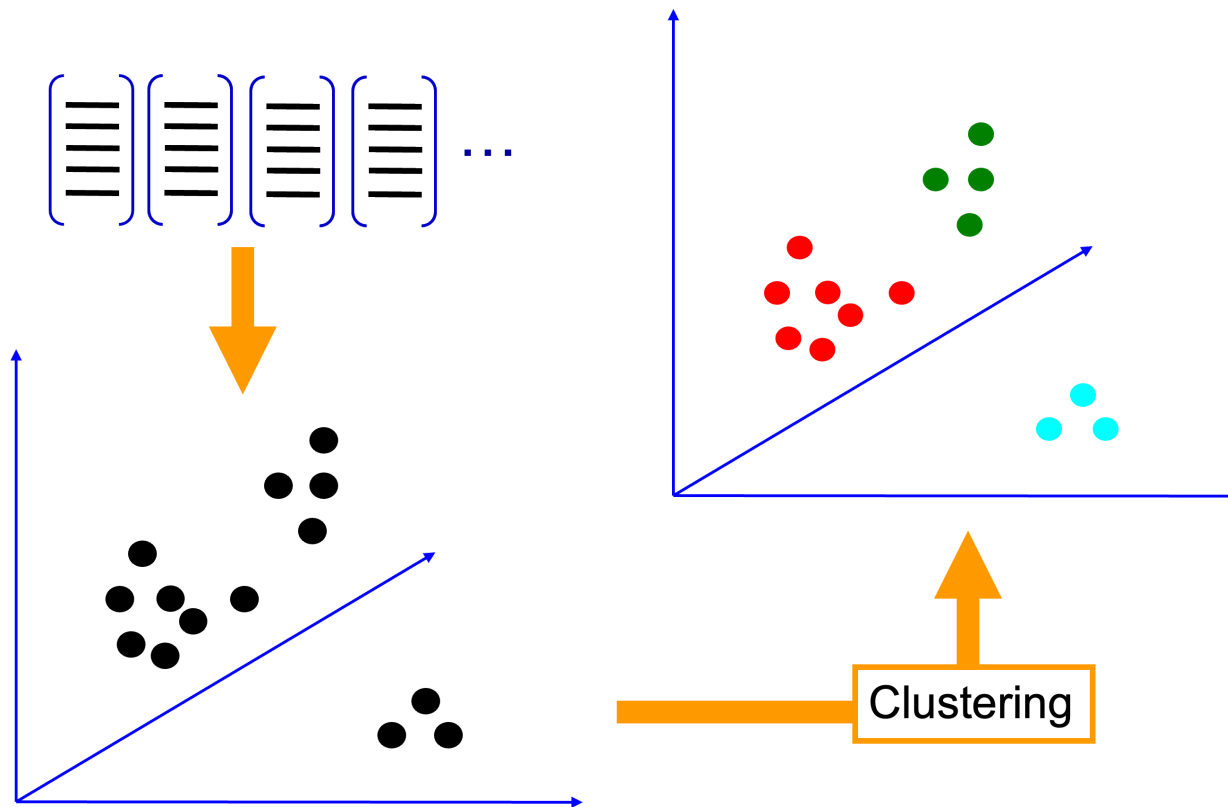
1-(1). Bag-of-Words (BoW) feature extraction

- Reduce the descriptor dimensionality using PCA [2]
 - This step is to reduce the training time



1-(1). Bag-of-Words (BoW) feature extraction

- Train a visual dictionary using K-means clustering [3]



1-(1). Bag-of-Words (BoW) feature extraction

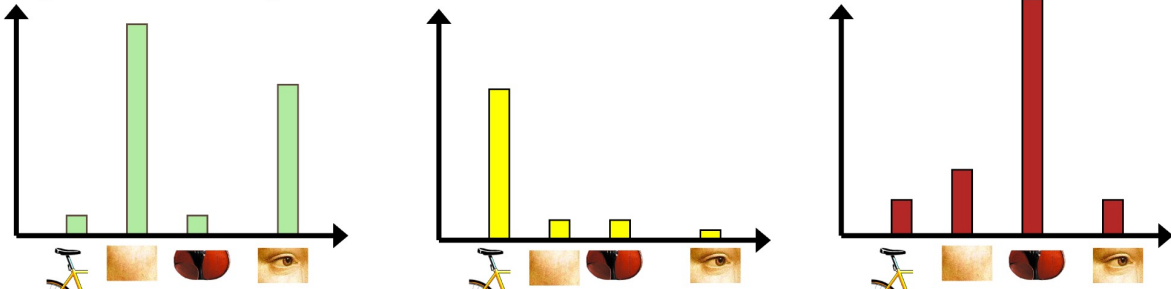
- Encode images to each fixed-length global vector using K-NN algorithm [4]



Visual words

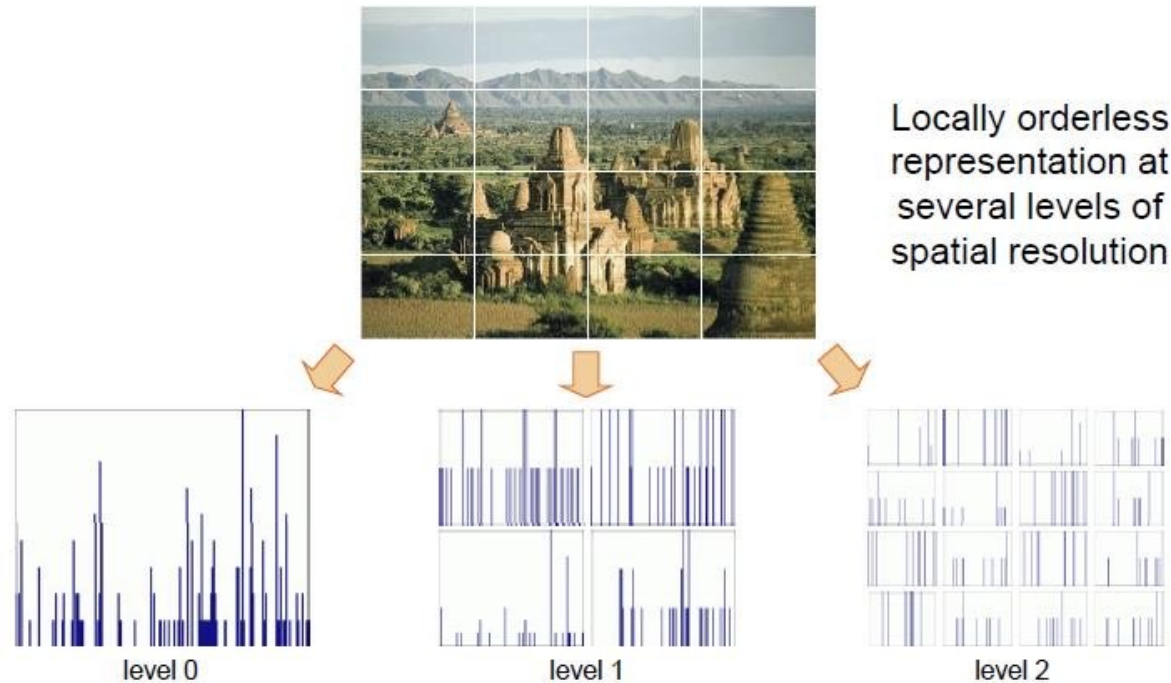


Bag of visual words histograms




1-(2). Bag-of-Words (BoW) with Spatial pyramid


- Implement spatial pyramid kernel to encode spatial information of local features (no given function)



1-(3 & 4). VGG features

- Load pre-trained VGGNet and extract features from the last convolutional layer.

 PyTorch Learn ▾ Ecosystem ▾ Edge ▾ Docs ▾ Blog & News ▾ About ▾ Become a Member 🔔 🔍



VGG-NETS

By Pytorch Team Award winning ConvNets from 2014 ImageNet ILSVRC challenge

[View on Github >](#) [Open on Google Colab >](#) [Open Model Demo >](#)

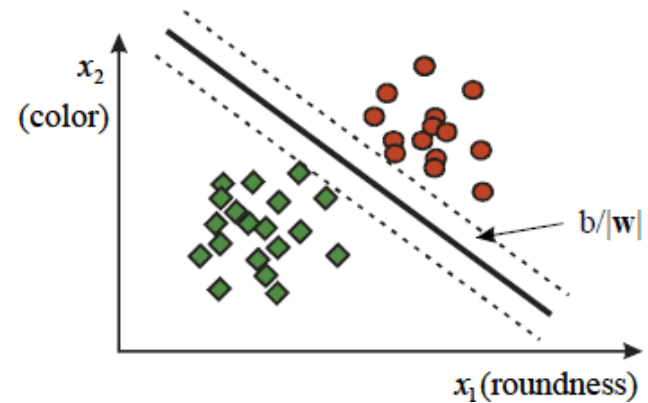
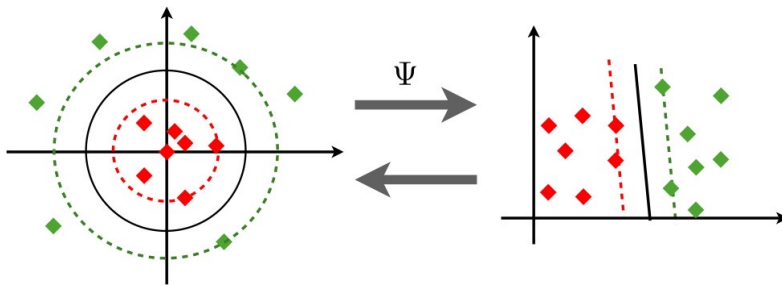
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 x 224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 conv1-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

```
import torch
model = torch.hub.load('pytorch/vision:v0.10.0', 'vgg11', pretrained=True)
# or any of these variants
# model = torch.hub.load('pytorch/vision:v0.10.0', 'vgg11_bn', pretrained=True)
# model = torch.hub.load('pytorch/vision:v0.10.0', 'vgg13', pretrained=True)
# model = torch.hub.load('pytorch/vision:v0.10.0', 'vgg13_bn', pretrained=True)
# model = torch.hub.load('pytorch/vision:v0.10.0', 'vgg16', pretrained=True)
# model = torch.hub.load('pytorch/vision:v0.10.0', 'vgg16_bn', pretrained=True)
# model = torch.hub.load('pytorch/vision:v0.10.0', 'vgg19', pretrained=True)
# model = torch.hub.load('pytorch/vision:v0.10.0', 'vgg19_bn', pretrained=True)
model.eval()
```

All pre-trained models expect input images normalized in the same way, i.e. mini-batches of 3-channel RGB images of shape $(3 \times H \times W)$, where H and W are expected to be at least 224. The images have to be

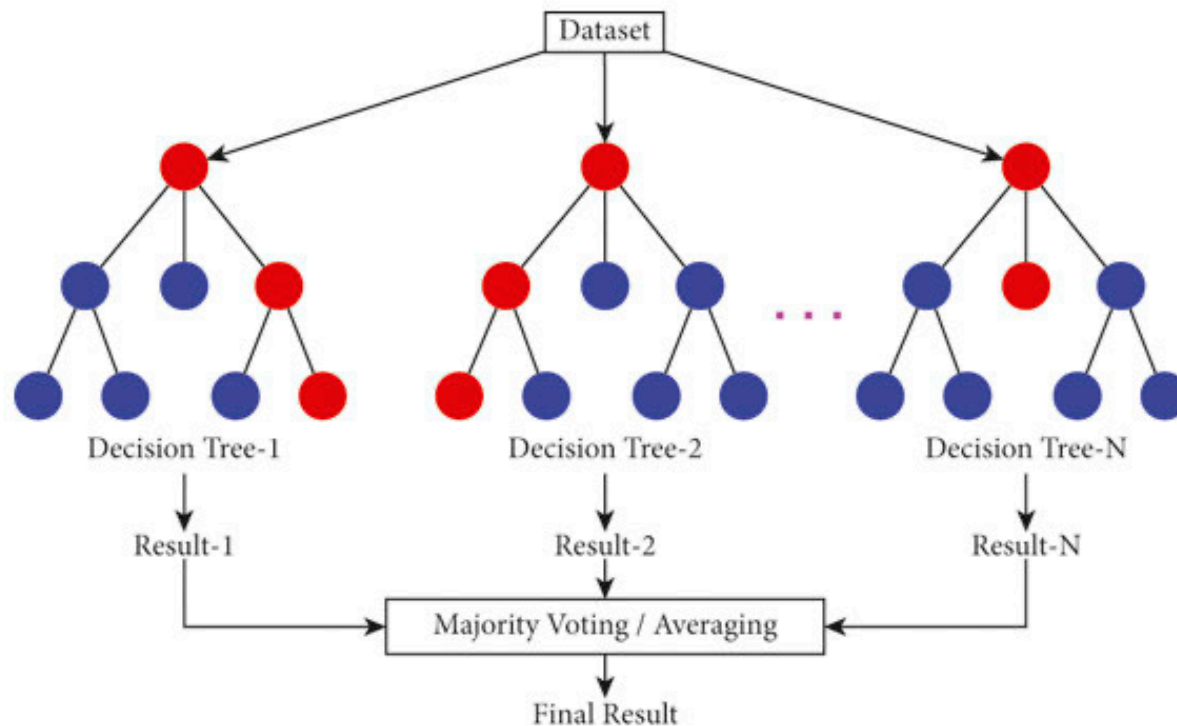
2-(1). Support-Vector-Machine (SVM) training & test

1. Train a decision boundary with the SVM method for four features (BOW, BOW+SP, VGG13, VGG19) using Linear SVM [5]
2. Predict classes of image vectors of a class



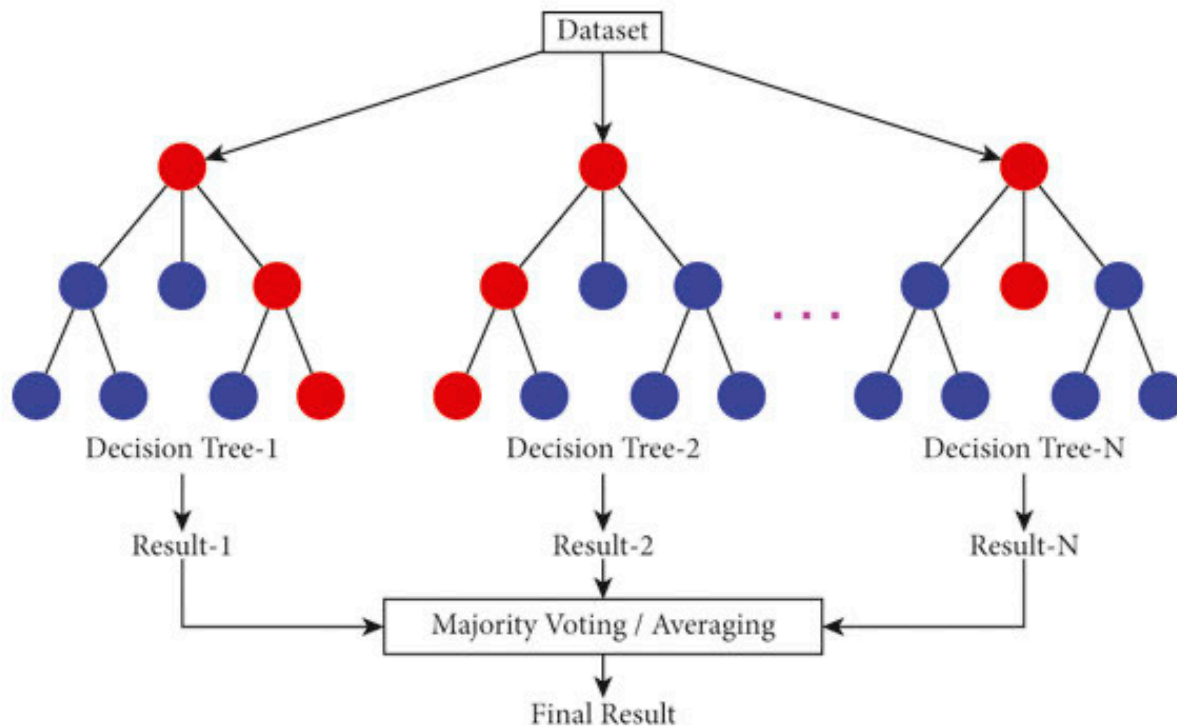
2-(2). Random Forest

1. Train a decision boundary with the Random Forest for four features (BOW, BOW+SP, VGG13, VGG19)
2. Predict classes of image vectors of a class



2-(3). 2-layer FC layer

1. Train a decision boundary with the 2-layer FC layer (use pytorch) for four features (BOW, BOW+SP, VGG13, VGG19).
2. Predict classes of image vectors of a class



3-(1). Image Retrieval with the Encoded Vectors

- Implement the code for image retrieval by comparing the distance of the features
- Report accuracy of retrieval from 4 different features (BOW, BOW+SP, VGG13, VGG19).



100%



97%



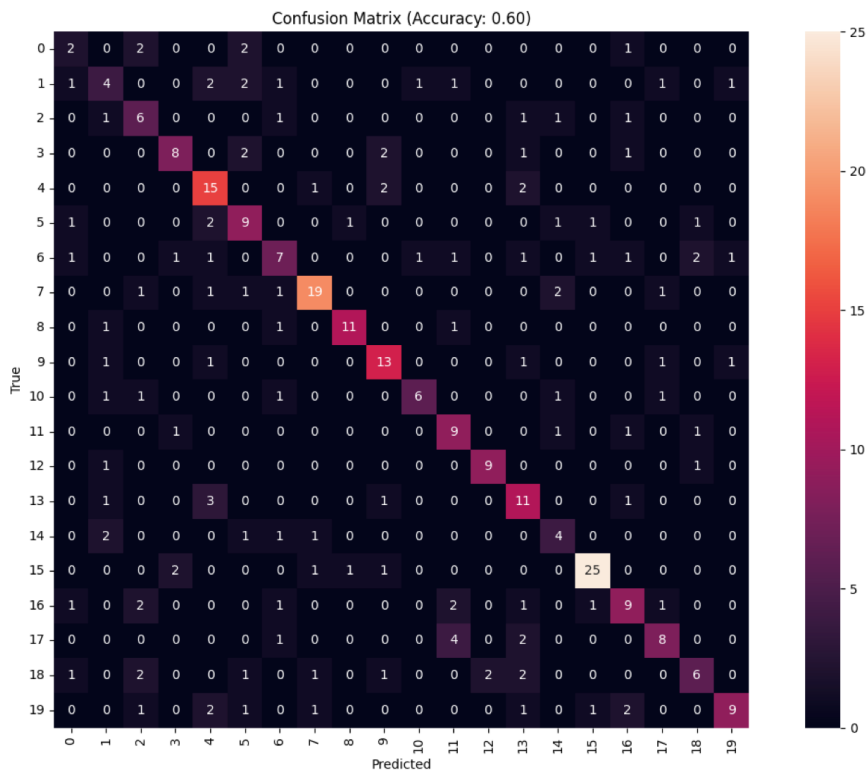
80%



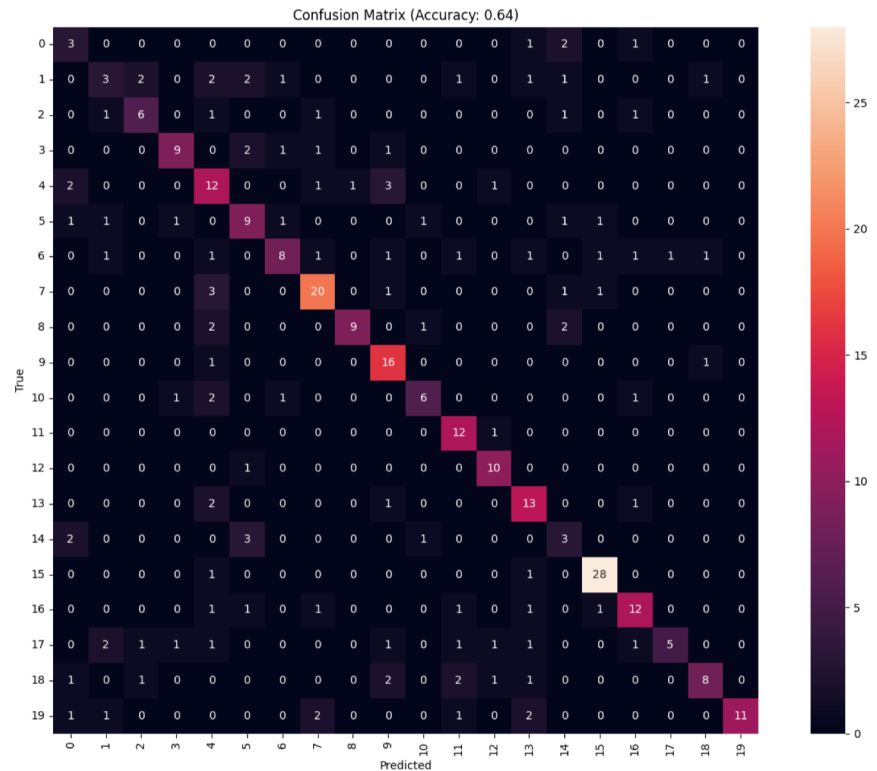
60%

3-(2). Classification accuracy

- Compare the performance of all 12 combinations of feature extraction (4 methods) and classifiers (3 methods) using confusion matrix



<Confusion matrix with BoW>



<Confusion matrix with BoW with Spatial pyramid>

Submission

- **Submission should include...**
 - Source code (Do not include provided datasets)
 - Readme file explaining how to execute the program
 - Report includes
 - The 4 image retrieval results (qualitative and quantitative)
 - The 12 confusion matrices and their average accuracies.
(It will be significantly reflected your grade.)
 - Compare the results of 4 feature extractor and 3 classifiers.
- **Note**
 - Please implement the code using python
 - You can use pytorch or tensorflow for CNN features (you do not need to train the CNN for feature extraction. Just use pre-trained features)
 - You should train the 2-layer FC layer with the frozen weights in Conv layers
 - You can use opencv function & ChatGPT to implement the code.

Notice

Due : May 9, 11:59PM
To : lms.dgist.ac.kr

1. Delayed submission

1. 25% score will be **degraded** every 1-day delay & after 3 days delayed, you will get 10% of the total score & after 1 week delayed, 0%.
* 100% → 75% (1day) → 50% (2days) → 25% (3days) → 10% (3~7days) → 0% (> 7days)

2. Plagiarism

1. **No grade** for copied codes (from friends and the internet)
2. You can refer to sources from the internet, but do not copy and paste.

3. Partial credit

1. Even though you are not successfully designing the network and obtained reasonable results, please send your code.
2. **There will be partial credit** for each module implementation.

Thank you