School of Computing Sciences

Module: CMP-6035B Computer Vision
Assignment: Image categorisation II.

Set by: Michal Mackiewicz M.Mackiewicz@uea.ac.uk
Checked by: Daniel Paredes-Soto d.paredes-soto@uea.ac.uk
Date set: 21st March 2025
Value: 50%

Date due: 14th May 2025 3pm week 12
Returned by: 23rd May 2025
Submission: Blackboard

Learning outcomes

To reinforce material presented in the lectures and to give you practical experience programming computer vision algorithms in Matlab. An in-depth understanding of the related computer vision algorithms. An understanding of the algorithm evaluation process. Improved Matlab programming skills with a focus on creating efficient solutions.

Specification

Overview

In coursework 1 you carried out a literature review of the image classification research and implemented a benchmark image classification algorithms based on tiny image and colour histogram features; and a kNN classifier. In this coursework you will use the knowledge you have gained in coursework 1 and from the lectures to implement further improvements to your algorithm.

You will work on this coursework in pairs.

Description

You will work with the same image dataset. You will use the same functions as a skeleton for your implementation.

There are many ways one can approach this coursework task. In coursework 1 you have found that tiny images and naive colour indexing do not support accurate classification. During the lectures, you have learned about other image features, classifiers and meta-algorithms that could be applied in this problem e.g. SIFT, HOGs, bag of features, spatial pyramid kernels etc. You have also performed your own literature review into these and similar algorithms. You will need to implement some of these to build a better performing solution. You may discover that your literature review in coursework 1 was insufficient and you need to consider other algorithms. You are encouraged to read more.

One of the algorithms you will need to implement during this coursework is bags of quantised SIFT features. First, you will need to establish a dictionary of visual words. This can be

achieved by collecting large number of SIFT features from the training set and then clustering them with *kmeans* algorithm. The number of kmeans clusters is the size of your dictionary and also the size of your features. You might start by clustering many SIFT descriptors into e.g. $k = 50$ clusters. The 50 features will partition a 128 dimensional SIFT feature space into 50 regions. Any new SIFT feature in the test set can be associated with one of these regions by finding the closest region centroid.

Since extracting and clustering SIFT features can be slow, we suggest that you precompute the cluster centroids and store them in a .mat file, which will save you lots of time in subsequent runs of the algorithm when you want to test different settings of the classifier. See `build_vocabulary.m` for more details. You can find the updated skeleton implementation for this coursework in the `coursework2_starter.m`

Once you have the cluster centroids, you can represent images as histograms of visual words. From each image you will extract 100s of SIFT features (use dense sampling provided in `vl_dsift`) and count how many of them fall into each bin in your visual word dictionary. This can be achieved by searching for the nearest dictionary centroid for every SIFT feature in an image. Hence, if you have a dictionary of 50 visual words, and detect 200 SIFT features in an image, your bag of visual words will be a 50 dimensional histogram where each bin counts how many times a SIFT feature was assigned to that cluster and sums to 200. You should normalise this histogram so that it sums to one to ensure that image size does not affect the magnitude of the bag of SIFT feature. See `get_bags_of_sifts.m` for more details. There are a few parameters for the bag of SIFT representation (number of clusters and other SIFT parameters). You will need to tune these parameters. See the documentation for the `vl_dsift` function.

The next task is to train 1-vs-all linear SVM to operate in the bag of SIFT feature space. This classifier has been explained in the lecture notes. Since linear classifiers such as this one are binary, you will need to train 15 binary, 1-vs-all classifiers. 1-vs-all means that each classifier will be trained to recognize 'forest' vs 'everything-but-forest', 'kitchen' vs 'everything-but-kitchen', etc. You will have to evaluate all 15 classifiers on each test feature and the classifier which is most confident about the positive result will determine its class label. SVM has a regularisation parameter called λ or C. You have to tune this parameter over a wide range of values as your results may strongly depend on it. See `svm_classify.m` for more details.

To complete the assignment you should:

1. Implement bag of words feature extraction utilising SIFT features (grayscale). You should use kNN classifier from coursework 1 and you need to implement an additional classifier - linear SVM. Test the algorithm for various parameters and discuss the results.

    - The same as above, but for the colour images. You will need to devise a way of fusing colour information and SIFT features. Look at [1] for some ideas.

2. Repeat 1. for the spatial pyramids with SIFT features, see [2].

3. Implement any other scene recognition algorithm. Here, you can use any algorithm from the literature as well as any classifier. You can also devise your own algorithm or fuse some of your ideas with the existing algorithms.

Relationship to formative assessment

You will receive formative feedback during all subsequent Mon lab sessions.

Deliverables

Marks will be awarded for the oral presentation and the electronically submitted code. The code will be used mostly for the verification of the presented results. This is a group exercise

that should be done in pairs. You should continue to work in the same pair you worked during coursework 1, unless there is a problem, in which case you should contact me as soon as possible.

1. Code hand-in electronically, **15:00 on Wed Week 12** through Blackboard. Both members of the pair must submit the same code. The Matlab source code must be zipped into a file with the name containing Student IDs of both members of the group. **Make sure you test your solutions on the lab machine BEFORE you submit**. You should add a brief commentary describing the individual contributions of both members of the pair and submit it together with your code.

2. The presentations have to be submitted together with the code mentioned above by the same deadline i.e. **15:00 on Wed, Week 12** to the same digital dropbox on Blackboard. **Presentations will take place on Thu and Fri in week 12.** You will be informed about your presentation slot on Mon, week 12 the latest.

3. The functions implementing your algorithm have to be called according to the specified naming scheme and placed in a folder named with your pair registration numbers. Any additional functions must be placed in the same folder.

4. After copying your coursework folder into my machine, I should be able to run your code on my machine as is. The only edit to the code you may expect me to make is changing the `data_path` variable in `coursework2_starter.m`. Therefore, you must not include the image dataset in your submission.

If you have medical or other problems you can seek extensions to coursework deadlines. However, it is essential you obtain proper documentation in such cases (i.e. a medical certificate).

If one of the pair members is granted an extension, the other member of the pair also receives an extension. Note, **self-certified extensions are not valid for presentations and groupwork and hence, they are not valid for this coursework**.

Both members of the pair are equally responsible for the joint work. In case of any breakdown of co-operation, you should complete your work individually and notify me by email.

Installation of the dataset and the starter code.

You need to install a VLFeat MATLAB toolbox [1]. Download the toolbox and run `vl_setup`. The toolbox contains a large selection of computer vision related functions. You must not bundle this toolbox with your submission. You can assume that on my machine the toolbox file path is set in MATLAB.

Resources

You should do the previous labsheets which are available on BB. You should also consult the lecture notes, Weeks 4 and 6 - 8 in particular.

You should attend all lab sessions and discuss your progress with the teaching staff who will be able to provide you with formative feedback.

Marking Scheme

This coursework carries 50% of the module weight, therefore you may receive up to 50 marks for this coursework. Marks will be awarded for the electronically submitted code and the presentation. The presentation should be 15 minutes long (maximum), plus questions. Both members of the pair must contribute to the presentation.

[1] `http://www.vlfeat.org/install-matlab.html`

The marks will be awarded according to the following scheme:

1. Implementation, result presentation and discussion of the Bag of SIFT features:
   (a) kNN and grayscale SIFT. (**5 marks**).
   (b) SVM and grayscale SIFT. (**5 marks**).
   (c) kNN and SIFT+colour. (**5 marks**).
   (d) SVM and SIFT+colour. (**5 marks**).
2. Implementation, result presentation and discussion of the spatial pyramids method with SIFT features:
   (a) kNN and grayscale SIFT. (**4 marks**).
   (b) SVM and grayscale SIFT. (**4 marks**).
   (c) kNN and SIFT+colour. (**3 marks**).
   (d) SVM and SIFT+colour. (**3 marks**).
3. Implementation, result presentation and discussion of the additional algorithm of your choice. (**10 marks**)
   - Here, marks will be awarded according to the complexity, performance and the thoroughness of implementation, testing and reporting.
4. Presentation style. (**6 marks**).
   - Oral presentation, quality of slides, timeliness.

You should add a brief commentary describing the individual contributions of both members of the pair and submit it together with your code.

Plagiarism, collusion, and contract cheating

The University takes academic integrity very seriously. You must not commit plagiarism, collusion, or contract cheating in your submitted work. Our Policy on Plagiarism, Collusion, and Contract Cheating explains:

- what is meant by the terms 'plagiarism', 'collusion', and 'contract cheating'
- how to avoid plagiarism, collusion, and contract cheating
- using a proof reader
- what will happen if we suspect that you have breached the policy.

It is essential that you read this policy and take (or refresh your memory of) our school's training on this. You can find the policy and related guidance here [2]

The policy allows us to make some rules specific to this assessment. Note that in this assessment, you are allowed to work within groups prescribed by the assessment setter. Discussing solutions to tasks between groups, or groups otherwise working together to perform the assessed tasks will be considered as a breach of university regulations. Please pay careful attention to the definitions of contract cheating, plagiarism and collusion in the policy and ask your module organiser if you are unsure about anything.

[2] `https://my.uea.ac.uk/departments/learning-and-teaching/students/academic-cycle/regulations-and-discipline/plagiarism-awareness`

References

[1] K. van de Sande, T. Gevers, and C. Snoek, "Evaluation of color descriptors for object and scene recognition." in *Proceedings of CVPR*, 2008.

[2] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, ser. CVPR '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 2169–2178. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2006.68