Intro to Image Understanding (CSC420)

Projects

Posted Oct 5, 2017;

Submission Deadlines: Proposal: Oct 24th 11.59pm;

Report Nov. 25th 11.59pm; Presentations Nov. 28, 30, Dec 5, 2017

Instructions for submission: Detailed instructions below. Please write a document and submit your proposal and report as **PDFs**. Include your code as a set of separate files, and submit through **MarkUS**.

Max points: 40

This document is long, but read it carefully and in full.

Projects can be done individually or in groups of up to three students. Most projects are quite big, so pair work is actually encouraged. Projects are calibrated for two students, for groups of three, justify what work each member will carry out and how you will expand upon the project in comparison to a group of two.

As a ballpark each student is expected to do roughly as much work as 2.5 of the regular term assignments on the project, and one assignment's worth of work on the report.

If a project is done in a group, each student should still hand in his/her own report and must be able to defend the project on his/her own. From the report it should be clear what each student contributed to the project. By November 25th you will need to hand in the project report including code. Make the code organized and documented, ideally including scripts that run your pipeline.

In the oral defense (Q & A during the presentation) you'll need to run some of your code and be able to defend it. The presentations are tentatively scheduled to be between Nov 28-Dec 5 (depending on how many project groups we have). The grade will evaluate the project proposal (5% of the grade), the project report (20% of the grade), and an oral presentation (15% of the grade).

Whenever you use ideas, code or data from a paper, forum, webpage, etc, you need to cite it in your report. Whenever you use code available from some paper or method, you need to include a short description of the method showing your understanding of the technique. If there were multiple options for techniques or code, please also explain why you chose a particular one (See below for report outline and details).

The grade will take into account the following factors:

- Your demonstrable work, what you've learned, and what knowledge you've applied.

- Problem solving. Your ideas to tackle a problem: how appropriate the techniques you chose are for the problem. Coming up with novel ideas is obviously a big plus.
- Whether you implemented a technique yourself or used code available online
- How you present your results in the report: ideally you would show your results for each task by including a few pictures in your report. Even more ideally, you would show good cases where your method works and also bad cases where it doesn't. Providing some intuitive explanation of the failure cases is a plus.
- Thoroughness of your report: How well you describe the problem and the techniques you chose, how well you analyzed your results and whether your citations to the techniques you used are appropriate. See below for report outline and details.
- Your implementation: the accuracy of the solution, speed, and partly also how organized the code is (scripts that run the full pipeline or specific subtasks, documentation). The grade will also take into account the accuracy of your solution in comparison to other students results.

It may be that the project has too many questions and you might not be able to solve them all. Don't worry. The goal is to teach you how to solve cool problems and hopefully you have fun doing it. We will evaluate relative to how everyone does. Think of it as a competition.

1 Project Proposal: Due Oct 24th 11:59pm

A project proposal is your commitment to a particular project. Write down what you chose, who your project partners are (if any), and a few sentences describing your plan how to tackle the problem. For groups of three, include the justification, what work each member will do, and how you will expand on the two-person project.

If you are proposing a non-listed project, please write a longer project proposal, outlining your general ideas and difficulties you think the project has, and reference the scope of work (in similar steps taken) to one of the included proposals. It should be clear that work is proportional, and we reserve the right to refuse a proposal (refusal will not affect your proposal mark).

2 Project Report: Due Nov 25th 11:59pm

Your project report is the basis for showing your work and demonstrating what you've learned in this course. It should be 4-6 pages in length (if single spaced) plus figures and references. Reference your code, but do not include it in the report. Build it as you go, be clear and concise. Use the following section breakdown. Notice that it approximately follows the structure of an academic paper.

Each student, in each group, prepares and submits his/her own individual report. You may share figures, but credit them to the student who created them accordingly.

2.1 Introduction and related work

Approximately 0.5-1.5 pages.

Give an introduction to your problem. Why you picked it, and what previous work you found and read (at a high level, referencing back to concepts/algorithms we learned in the course).

2.2 Methods

Approximately 1.5-2 pages.

Describe each step of your method. If you use an algorithm, e.g. SIFT, summarize in your own words how it works. I suggest using pictures and diagrams to show the modular/algorithmic flow, and results of intermediary steps of your algorithm to show your processing pipeline. Make clear references/associations to your code so we can follow along.

Make it clear which parts you did, which work your group did, and what was obtained online.

2.3 Main challenges

Approximately 0.5-1 pages.

Describe the main challenges you faced, and how you solved them.

2.4 Results and discussions

Approximately 1-2 pages.

Describe and demonstrate your results. Compare various steps of your process, e.g. how much additional accuracy is gained by each step. Show example images and/or videos.

Explain why your method works when it does, and why it fails. Show examples and justify (i.e. if it a is problem with the algorithm or a problem with your implementation).

2.5 Conclusion and future work

Approximately 0.25 pages

Summarize what you did, how it worked, and what you would do in the future to improve your results.

3 Sample Project 1

This project is about analysis of news broadcast. You will be given a news video clip. Here are the tasks to solve:

1. Download three sets of news broadcasts: Lego Space man;Clowns.. Parliament
2. Download a set of male and female faces with gender labels: link
3. Detect shots in the videos. A shot is a set of consecutive frames with a smooth camera motion.
4. (Manually) Annotate shot boundaries in the video. How would you evaluate how well you are detecting the shots? Compute your performance.
5. Detect the news companys logo (without prior knowledge of location).
6. Detect faces in the video.
7. Perform face tracking by correctly associating a face detection in the previous frame to a face detection in the current frame.
8. Train a classifier that can predict whether a face is female or male. For each face track in the news video predict whether it is female or male. To do this you will take a few images of faces, compute image features and train a male-vs-female classifier, e.g., SVM, NN. Once trained, you will predict the gender of each face detection in the video. That is, youll take each face detection (a crop in the image specified by the face box), compute appropriate image features, and use your classifier to predict the gender of the face. How would you decide whether a full face track is female or male?
9. Visualize your results: produce a video in which you show a bounding box around the detected company logo, and bounding boxes around the detected faces. Each face bounding box should have text indicated whether the face is male or female.
10. **Bonus:** Can you detect the clowns as a 3rd or class?

4 Sample Project 2

This is a project for a single student (no group work allowed).

Imagine a phone app where you take a picture of a DVD and the app tells you which movie it is. In this project, the goal is to localize and recognize a DVD cover in an input image given a database of a large set of DVD covers. You will need to implement the following paper:

Nister, Stewenius, Scalable Recognition with a Vocabulary Tree, CVPR 2006,http://www-inst.eecs.berkeley.edu/ cs294-6/fa06/papers/nister_stewenius_cvpr2006.pdf

You will be download a set of images of DVD covers containing training images of DVD cover templates, and testing images each containing one DVD in a non-frontal viewpoint taken from a phone camera. Here are the tasks:

1. Download the Stanford Mobile Visual Search Dataset database: link. The **Reference** covers are your training data, and the rest is for testing.

2. Perform homography estimation with RANSAC to match one DVD cover image to a test image.

3. Implement the efficient retrieval approach by Nister and Stewenius including:

   (a) Keypoint detection

   (b) Keypoint description with rotation and scale in-variance

   (c) Building and using the vocabulary tree (i.e. quantization)

   (d) Hierarchical scoring

   (e) Retrieval

4. For a test image, retrieve top 10 matches (DVD cover images from the database) returned via the implemented approach. Compute a homography with your implementation from (2) for each retrieved DVD cover image and the test image.

5. Find the DVD cover image from (4) with the highest number of inliers. Plot the test image with the localized DVD cover as well as the best retrieved DVD cover.

6. This is a 2006 paper, come up with and implement your own improvements to address the miss-classifications results based on what we've learned or other resources (see bottom of this document).

5 Sample Project 3

This is a project on segmentation and object recognition in 2D medical images using deep learning. You will build an automated system to help characterize histopathology images of cancerous cells.

There are two main parts to this project, detecting the cells and then classifying them.

A key question in cancer research is why two people with the same broad cancer type (e.g. colon cancer) have different outcomes from treatment. One patient will receive treatment and be healthy, and another will receive the same treatment and have a cancer recurrence in a few years. Recent work with genetics and imaging data has begun to break broad classes of cancer into smaller subsets involving different phenotypes of tumors; that in turn respond differently to treatment. One of the barriers to this analysis is characterizing the cell distribution within the tumor to analyze its spatial arrangement (e.g. tightly clustered, vs disperse) or its composition in terms of different types of cell nuclei (see the manuscript for details).

Sirinukunwattana, Korsuk, et al. "Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images." IEEE transactions on medical imaging 35.5 (2016): 1196-1206, link.

1. Download the data set: link.
2. Detect patches with a nucleus at the centre:
   (a) Use techniques learned in the course to build an approach to detect nucleus centres; much like you would faces, cars, etc.
   (b) Experiment with different approaches using both grayscale and colour information; Compare and contrast.
   (c) Reproduce (approximately) their baseline CP-CNN result. Take a look at MatConvNet (link).
   (d) Setup cross-validation and evaluate your results. Compare to those in the paper.
3. Classify the detected nuclei patches in the 4 classes (epithelial, inflammatory, fibroblast, and miscellaneous):
   (a) Build your own patch-feature descriptor
   (b) Classify your detected patches into one of the 4 classes, and visualize your results.
   (c) Experiment with different approaches using both grayscale and colour information; Compare and contrast.
   (d) Implement their NEP classification step (this can be combined with any type of classifier in practice) and try in conjunction with your own approach

neighbouring ensemble predictor

6 Sample Project 4

Autonomous driving is one of the major research venues these days. A lot of effort is devoted to it by both academics and industry. In this project youll familiarize yourself with some of the most important problems that arise in the field of autonomous driving.

The input to your algorithm is a stereo image pair and the camera parameters. Here are the tasks to solve:

1. Part 1 road detection:
   (a) Download the left and right colour images and calibration data for the road recognition dataset: Info and link. You will find training and testing data, do not mix them up.
   (b) Compute disparity between the two stereo images. We do not mind if you use existing code as long as you include a description of the algorithm you used, showing you understand what it is doing.
   (c) Compute depth of each pixel. Compute 3D location of each pixel.
   (d) Train a road classifier on a set of annotated images, and compute road pixels in your image. Which features would you use? Try to use both 2D and 3D features.
   (e) Train your algorithm on the training data, and show example outputs on the testing images.
   (f) Fit a plane in 3D to the road pixels by using the depth of the pixels. Make sure your algorithm is robust to outliers.
   (g) Plot each pixel in 3D (we call this a 3D point cloud). On the same plot, show also the estimated ground plane.
2. Part 2 object detection:
   (a) Download the left and right colour images for the object recognition dataset and the camera calibration matrices: Info and link. You will find training and testing data, do not mix them up.
   (b) Detect cars in the image. You can use the pre-trained models available here: link.
   (c) Train a classifier that predicts viewpoint for each car. The viewpoint labels are in 30°increments, thus train 12 classifiers. Which features would you use?
   (d) Show a test image with the detected car bounding boxes and show the estimated viewpoints by plotting an arrow in the appropriate direction.
   (e) Given the ground plane, estimated depth, and the location of the cars bounding box, how would you compute a 3D bounding box around each detected car? Add the 3D bounding boxes to your plot from 1.g.
3. Part 3: Make a result video of your algorithm running on the road recognition test set, see **getFrame.m** and **videowriter.m**.

7 Building Your Own

1. Prepare a more detailed proposal.
2. Download data, there are more links to data sets at the end of this document, or build your own.
3. Your project must integrate at least 3 of the courses main subjects: Image processing, Features and Matching, Geometry, and Recognition
   (a) The fewer the number of subject categories used, the more depth, detail, experimentation, and implementation expected in each.
4. More examples
   (a) Build a 3D motion capture system link.
   (b) Build a pedestrian detector and/or tracker link.
   (c) Build an activity recognition system link.

8 Useful Code and Techniques

- **Shot Detection:** A simple way of detecting shot boundaries is to look at differences in color histograms between two consecutive frames. An even better way is to look at Displaced Frame Distances, described in Section 2.1 of this paper:

  1. Makarand Tapaswi, Martin Baeuml and Rainer Stiefelhagen, Knock! Knock! Who is it Probabilistic Person Identification in TV Series, CVPR 2012, link
  2. More options are discussed in this paper: Y. Yusoff, W. Christmas, and J. Kittler, A Study on Automatic Shot Change Detection. Multimedia Applications and Services, 1998, link.

- **Face Detection:** A few options:

  1. Paul Viola and Michael Jones, Rapid object detection using a boosted cascade of simple features, CVPR, 2001, link.
  2. P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object Detection with Discriminatively Trained Part Based Models IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 32, No. 9, Sep. 2010 link. Code link with a trained model called dpm_baseline.mat available here:link. This detector may work a little better. You can also check a detailed analysis and code for different face detectors here: link.
  3. X. Zhu, D. Ramanan. Face detection, pose estimation and landmark localization in the wild, CVPR 2012, link. This detector however also gives you the keypoints for the facial landmarks.

- **Tracking:**

  1. Barbu, Andrei, et al. "Simultaneous object detection, tracking, and event recognition." arXiv preprint arXiv:1204.2741 (2012) link.
  2. Geiger, Andreas, et al. "3d traffic scene understanding from movable platforms." IEEE transactions on pattern analysis and machine intelligence 36.5 (2014): 1012-1025, link. First 3 paragraphs of Section 4.1
  3. Tracking via Kalman filter: lots of tutorials online
  4. H. Pirsiavash, D. Ramanan, C. Fowlkes. Globally-Optimal Greedy Algorithms for Tracking a Variable Number of Objects, Computer Vision and Pattern Recognition (CVPR), 2011 Paper and code.
  5. http://www.cvlibs.net/software/trackbydet/
  6. http://research.milanton.de/dctracking/

- **Stereo:** For autonomous driving, its best to check the stereo challenge of the KITTI dataset (link). It has links to code and gives you running time of each algorithm. Some options:

  1. Large scale semi-global matching: link.

2. Yamaguchi, Koichiro, David McAllester, and Raquel Urtasun. "Efficient joint segmentation, occlusion labeling, stereo and flow estimation." European Conference on Computer Vision. Springer International Publishing, 2014 Paper and code.

- **Optical Flow:**
  1. Code here: link.
  2. Matlab has some functions to compute flow: **opticalFlow.m**
  3. OpenCV has flow and trackers, and other useful stuff: link.
- **Other:**
  1. Many libraries in vlfeat link.
- **More data sets:**
  1. Deep learning data sets: link
  2. Machine learning sets: link
  3. Computer vision sets: link