Using the SRVC code

Sudheendra Vijayanarasimhan and Kristen Grauman
Department of Computer Sciences
University of Texas at Austin
{svnaras,grauman}@cs.utexas.edu

July 2, 2008

Requirements:

- **Linux** (tested on Ubuntu Fiesty Fawn)
- **bash**
- **perl**
- **matlab**
- **convert** (the ImageMagick convert program)
- **hierarchical clustering code** (bow-trees.out, hierarchical-cluster-point-set.out)

  Binaries are included with this package. In case you need to recompile for a different platform you can get the source code here http://people.csail.mit.edu/jjl/libpmk/
- **smil** (the sparse MIL classifier used in [1])

  The binary is included with this package. In case you need to recompile it for a different platform please contact razvan@cs.utexas.edu
- **extract_features.ln** (for extracting binary features)

  The binary is included with this package. It can be downloaded from http://www.robots.ox.ac.uk/ṽgg/research/affine/detectors.html

1 Getting started

1. To get started, make sure the directory "srvc-code/" is in the system path (for eg: in bash, *export PATH=$PATH:<path to srvc-code>*).

[1] Multiple Instance Learning for Sparse Positive Bags, Bunescu and Mooney, ICML, 2007

2. Make sure all script files in srvc-code have execute permissions (*chmod +x srvc-code/**).

3. Make a directory where all the intermediate images, feature files, etc will be stored (for eg: */home/<user>/robotchallenge*)

4. Make directory */home/<user>/robotchallenge/config/* and create a text file called *categories_en* containing the list of categories for which models need to be built. (see *srvc-code/categories_en* for an example)

5. Type *learn_models.sh /home/<user>/robotchallenge/* to download the images for the categories and build the models.

6. Make a directory called */home/<user>/robotchallenge/test/images/* and copy the test images into this directory.

7. Type *find_objects.sh /home/<user>/robotchallenge/* to find the images containing objects of the learnt categories.

2 Learning the models

This section gives a brief run through of the steps required to build the models. (these steps are in *learn_models.sh*)

2.1 Translating the keywords into multiple languages

The command

translate.pl <input-filename> de <output-filename>

converts all words in the input file into the german language and stores it in the output file. Use 'en' for English, 'de' for German, 'fr' for French.

2.2 Downloading images

The command

get_images.pl <keyword> <number of images> <directory to save in> <language>

downloads images of the given keyword from google, yahoo and msn image search and saves them in the specified directory. *<language>* can be en, fr, de, it, etc.

2.3 Feature extraction and Bag of words

First, all the downloaded images need to be converted to the pgm format. The command

convertall <file containing list of directories> <output directory>

converts all images within the directories specified in the first argument and saves them in the output directory respecting the directory structure.

Next,

extractkeypoints <list of directories> <ouput directory>

extracts sift features from hessian affine regions and saves the results in the output directory again respecting the directory structure.

To randomly sample a list of sift feature vectors for clustering use

selectimages.pl <basedir>/train/keypoints/ 50 sift <basedir>/train/centres/universe.ds 18 300

Here 50 is the number of images to sample from each of the categories. 18 is the number of categories and 300 is an upper limit on the number of features to choose from each image. The output file universe.ds is a text file containing a list of 128-d features.

Running the command,

hier_cluster('<basedir>', 'universe.ds', '<clusters filename>', '<intermediate filename>');

in Matlab does hierarchical clustering on the given input file (universe.ds) and saves the cluster information in the files *<clusters filename>* and *<ptsets filename>*. The number of levels and the branching factor are currently hard-coded in variables NUMLEVELS =4, NUMCLUSTERS = 1000 in files *hier_cluster.m, makebags.m and print_bows.m.* At the end of this stage the directory *<basedir>/train/bags/* will contain one file for each of the directories in *<basedir>/train/images/* which contains the bag of words representation of each image within the directory.

2.4 sMIL models

The command

makebgmodels 0 <number of categories - 1>

creates sMIL models for each of the categories using the feature files in *<basedir>/train/bags/*. It uses the binary "smil" and an RBF kernel with kgamma = 5. Type "smil" at the command prompt for a list of options. The models are saved in *<basedir>/models/*

3 Detecting objects

The test images need to be stored in *<basedir>/test/images/* . The script "find_objects.sh" first scales the test images to 1024x768 and converts them to the 'pgm' format (saved in *<basedir>/test/pgm*). Features are extracted on the from the test images using *extracttestkeypoints* and saved in *<basedir>/test/keypoints/*.

Each feature from each image is then mapped to the cluster centres using *print_bow.m.* Windows of three different scales are extrated from each image

using *extract_windows.m* and the sift features falling in each of the windows are converted to the bag of words representation using the cluster centres found in the training stage.

Finally *find_objects.m* evaluates the windows using the models learnt and saved in *<basedir>/models/* and outputs images for each of the categories with a bounding box around the object.