Sibgrapi 2016 - Tutorial

Image Operator Learning and Applications

October / 2016

Igor S. Montagner, Nina S. T. Hirata and Roberto Hirata Jr.

Laboratory session

In the next exercises you will use Trioslib, a framework to design image operators using machine learning. Follow the instructions bellow if you have not installed the library previously to this section.

Step 1 Install Anaconda Python in your notebook. This is an open data science platform by Continuum Analytics and you can download it here:

`https://www.continuum.io/downloads`

Step 2 **Windows and Linux** Once you have installed Anaconda Python, open a terminal and type:

```
conda install -c igordsm trios
```

MacOSX Open a terminal and type:

```
pip install trios
```

You will need to install gcc, since trios requires OpenMP to run.

- If the installation is sucessful, you will see some messages as bellow:

```
Successfully built trios
Installing collected packages:  trios
Successfully installed trios-2.0.6
```

Step 3 Download images and scripts at `http://vision.ime.usp.br/projects/trios/tutorial/practical-exercises.tar.gz`

Exercise 4 In this exercise we will learn how to build image sets and windows, train WOperators and evaluate their performance on a test set.

(a) Open a terminal.

(b) List the contents of the script, `ex1.py`, that we prepared for this exercise. Can you identify the size of the window? Write down the size of the window.

(c) Display and examine the training images. What kind of processing they represent ?

(d) Run the script `ex1.py` and write down the error of the WOperator. To run the script, just type `python ex1.py` in your terminal.

(e) Modify the script to try some other window sizes. For instance, try the following sizes and write down the error of each operator.

    3×3 square;

    7×7 square;

    9×7 rectangle;

(f) Which one has the smallest error?

(g) Repeat the previous exercise using 2, 3 and 5 images for training. How does this affect the performance of the different window sizes?

Tips:

- Windows are represented by numpy arrays of type `uint8`. Nonzero elements belong to the window.
- Use `for` loops to change the window sizes and number of images used in items 1 and 2.
- The messages `Testing 0 ...` in eval are printed to `stderr`. Redirect the output of the script to capture only the errors.
- Add the image pairs `jung/jung-(s-)3a.png`, `jung/jung-(s-)5a.png`, `jung/jung-(s-)7a.png` an `jung/jung-(s-)9a.png` to the training set. Use the input image as mask.

Exercise 5 In exercise 4 we used ISI as the classifier. In this exercise we will use classifiers from scikit-learn in. Open `ex2.py` and complete the code to train operators using the models Logistic Regression and Decision Tree. Answer the following questions:

(a) which one is faster to train?

(b) And to evaluate?

(c) Which one has the best accuracy?

(d) Interpret the use of Logistic Regression in a WOperator. What image processing operation does it implement?

Tips:

- Logistic Regression in scikit-learn: `http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`.
- Decision Tree in scikit-learn: `http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html`
- Use an instance of SKClassifier to wrap the scikit-learn models. Use the RAWFeatureExtractor class to extract the features.
- The arguments of the constructor of WOperator are (1) window, (2) classifier and (3) feature extractor.

Exercise 6 Now we will train a two-level operator using a combination of operators trained using windows with simple shapes. Open `ex3.py`.

- Train operators using the following windows (in a 9×9 domain):

  Horizontal line with height 3

  Vertical line with width 3

  Square of size 5 in the top left corner

  Square of size 5 in the top right corner

  Square of size 5 in the bottom left corner

  Square of size 5 in the bottom right corner
- Create an instance of `CombinationPattern` and pass as arguments the operators created above.
- Train a new WOperator using a model from scikit-learn as classifier and the object created in the previous item as feature extractor.
- Evaluate the performance. How it compares to the previously trained operators? How does it compare with the error of the combined operators?
- (Optional) Try other window shapes.

Tips:

- Instances of `CombinationPattern` have an attribute `window`. Pass it to the WOperator class when creating two level operators.

Exercise 7 Train operators using whichever technique you like on the images in the `veja` folder. Training images are available in `level1.set` and `level2.set` and test images are at `test1.set` and `test2.set`. See below some of our most recent results.

Domain	Method	*Error*
9×7	NILC	**0.029**
	KA	0.034
	WER	0.037
	Manual	0.046
11×11	NILC	0.024
	KA	**0.021**
	IT	0.031
	Manual	0.031

Table 1: Recent results for the `veja` dataset.