5a. ML 1: Digits recognition

The machine learning task has two parts - **symbol classification** and **determination of the optimal classifier parameter**. Check the Upload system [https://cw.felk.cvut.cz/upload/] for the due date and notice that both tasks are submitted **separately**. It is expected that the classifiers will be implemented by students, usage of advanced pattern recognition libraries is forbidden. If unsure, ask the TA.

Data provided by Eyedea Recognition [http://www.eyedea.cz], some data are from public resources.

Problem

The task is to design a classifier / character recognition program. The input is a small grayscale image of one handwritten character - a letter or number - the output is a class decision, i.e. the recognition of the character in the image.

You are given training data, a set of images with the information on the correct classification. This is usually all that the customer provides. After you prepare the code, the customer, in this case represented by the instructor, will use different test data on which to evaluate your work. We recommend dividing the provided data into a training and test set.

Your resulting code will be tested on the new data within the AE system.

Data

The images are in the png format in one folder, where we also provide the file `truth.dsv` (dsv format [https://en.wikipedia.org/wiki/Delimiter-separated_values]). The file names are not related to the file content. The file truth.dsv has on each line `file_name.png:character` , e.g. `img_3124.png:A` . The separator character is `:` , which is never in the name of the file. The names of the files contain only characters, numbers or underscores (_).

- train_1000_10.zip [/wiki/_media/courses/b3b33kui/cviceni/strojove_uceni/train_1000_10.zip] images 10×10, 20 classes, 50 exemplars for each.
- train_1000_28.zip [/wiki/_media/courses/b3b33kui/cviceni/strojove_uceni/train_1000_28.zip] images 28×28, 10 classes, 100 exemplars for each
- train_700_28.zip [/wiki/_media/courses/b3b33kui/cviceni/strojove_uceni/train_700_28.zip] images 28×28, 10 classes, varying number of exemplars

Specifications

You have to upload two files, `knn.py` and `naive_bayes.py` , to be run with parameters, see below:

```
>> python3 knn.py -h
usage: knn.py [-h] (-k K) [-o filepath] train_path test_path

Learn and classify image data with a k-NN classifier.

positional arguments:
  train_path    path to the training data directory
  test_path     path to the testing data directory

optional arguments:
  -h, --help    show this help message and exit
  -k K          number of neighbours (if k is 0 the code may decide about proper K by i
  -o filepath   path (including the filename) of the output .dsv file with the results
```

So, for example:

```
python3 knn.py -k 3 -o classification.dsv ./train_data ./test_data
```

runs the 3-NN classifier.

Similarly,

```
>> python3 naive_bayes.py -h
usage: naive_bayes.py [-h] [-o filepath] train_path test_path

Learn and classify image data with a naive bayes classifier.

positional arguments:
  train_path    path to the training data directory
  test_path     path to the testing data directory

optional arguments:
  -h, --help    show this help message and exit
  -o filepath   path (including the filename) of the output .dsv file with the results
```

The output of the classifier is the file specified by the `-o` parameter (`classification.dsv` by default), which should have the same format as the `truth.dsv` .

Restrictions

Classifiers must not:

- Use the Internet.

- Spy on disk or elsewhere.
- We expect that you will use the `numpy` and pillow (`PIL`) libraries. Using other third-party libraries like `scikit-learn` , `pandas` , etc. is not allowed because they may not be installed on the evaluation machine. Just use the standard Python library. If you really need external libraries for something, discuss it with your teacher as soon as possible, who will assess whether it is beneficial from the point of view of the subject. If your intention is justified, we will try to comply with your wish, but we do not guarantee that it will be successful.

What to submit?

You will submit a ZIP archive with your modules `knn.py` , `naive_bayes.py` and any extra modules you wrote that these modules import (if any). **These files must be in the root of the archive; the archive must not contain any directories!** You upload the ZIP archive (and only this file) to BRUTE [https://cw.felk.cvut.cz/BRUTE/]

Solution structure

If you are not sure how to solve this task, we offer the following tips. Your solution will probably use the following partial steps:

- Command-line arguments processing [/wiki/courses/be5b33kui/semtasks/05_ml1/argparse] (including basic solution skeleton)
- Listing a directory content [/wiki/courses/be5b33kui/semtasks/05_ml1/listdir]
- Reading .dsv file [/wiki/courses/be5b33kui/semtasks/05_ml1/readcsv]
- Reading .png image in the form of numerical vector [/wiki/courses/be5b33kui/semtasks/05_ml1/image]
- Distance of two images [/wiki/courses/be5b33kui/semtasks/05_ml1/distance]
- For the work with image data, we suggest to use numpy.array [https://numpy.org/doc/stable/reference/generated/numpy.array.html]. If you would like to use other libraries, do not forget to test your solution in BRUTE early, or ask your lab instructor if your library is not too exotic.

Symbol classification - Evaluation

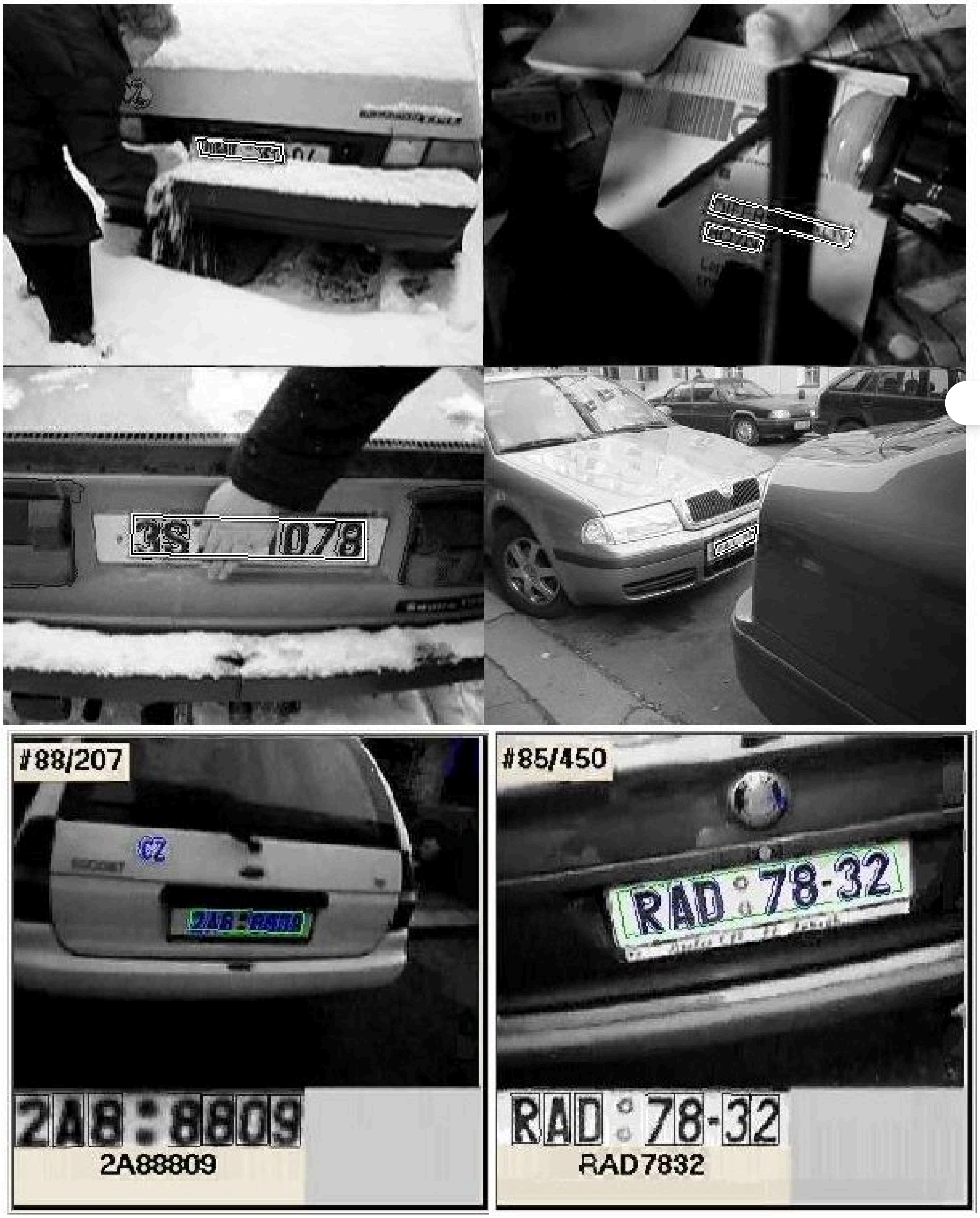
Automated Evaluation (AE) will only check if your code works well and show you the correctly classified ratio on the small AE dataset. However, the actual points will be awarded in a later ("tournament") run on a large dataset.

- Nearest neighbor classifier (1-NN) is evaluated according to the table below. [0–3 points]
- The Naive Bayes classifier follows also the table below: [0–5 points]
- Auto-evaluation: [0–1 points]
- Code quality: [0–1 points]

1-NN	
correctly classified	points
>93%	3
>80%	2
>60%	1
=<60%	0

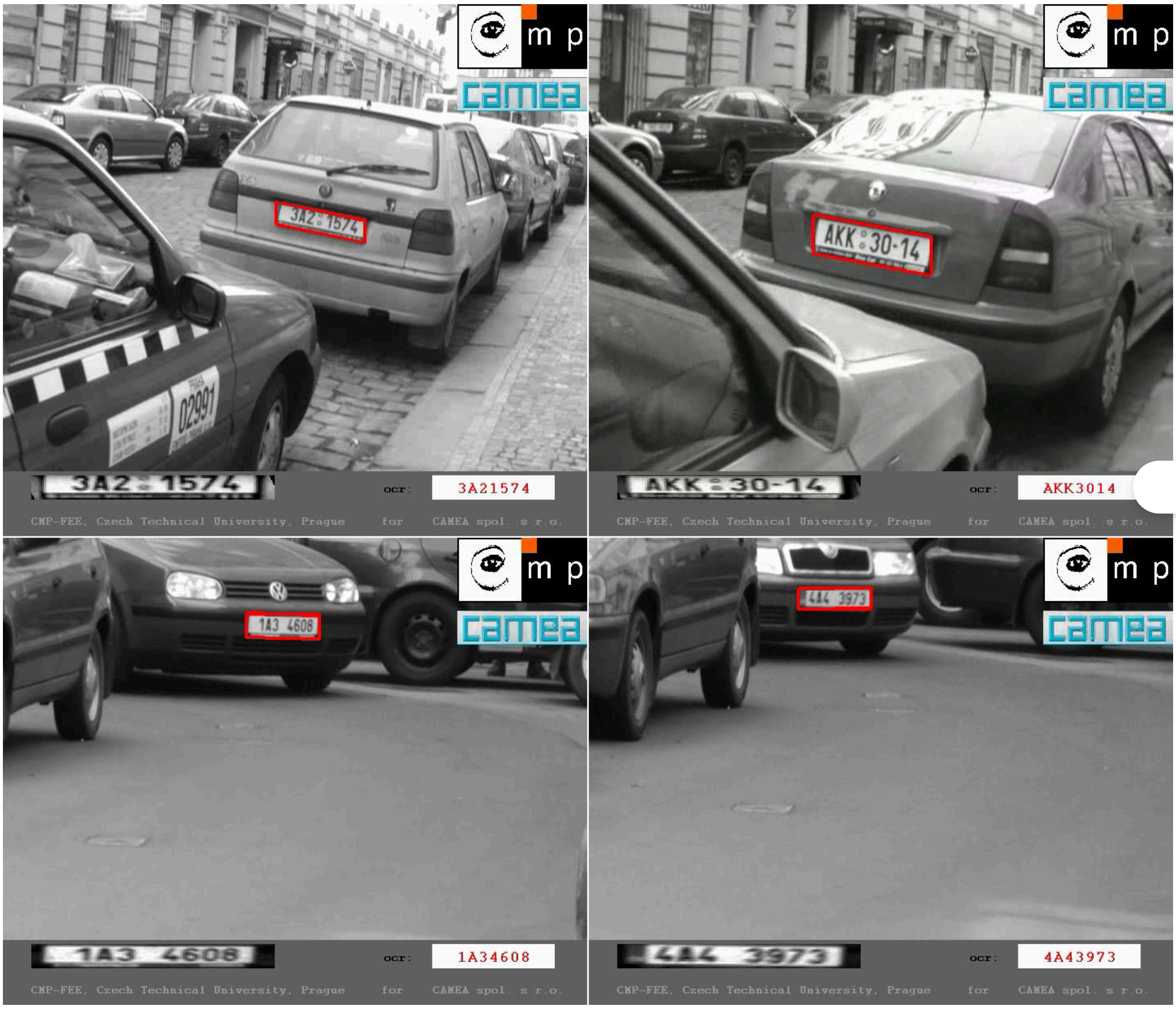
Naive Bayes classifier	
correctly classified	points
>82%	5
>75%	4
>70%	3
>65%	2
>60%	1
>55%	0.5
=<55%	0

Examples of use



[/wiki/_detail/courses/be5b33kui/labs/machine_learning/02_03_04_obr5.png?id=courses%3Abe5b33kui%3Asemtasks%3A05_ml1%3Astart]

Fig. 3: *Automatic text localization from pictures. More information available at http://cmp.felk.cvut.cz/~zimmerk/lpd/index.html* [http://cmp.felk.cvut.cz/~zimmerk/lpd/index.html].



[/wiki/_detail/courses/be5b33kui/labs/machine_learning/02_03_04_obr6.png?id=courses%3Abe5b33kui%3Asemtasks%3A05_ml1%3Astart]

Fig. 4: *Industry application for license plate recognition. Videos are available at http://cmp.felk.cvut.cz/cmp/courses/X33KUI/Videos/RP_recognition [http://cmp.felk.cvut.cz/cmp/courses/X33KUI/Videos/RP_recognition].*

References

Christopher M. Bishop. *Pattern Recognition and Machine Learning.* Springer Science+Bussiness Media, New York, NY, 2006.

T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory,* 13(1):21–27, January 1967.

Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern classification.* Wiley Interscience Publication. John Wiley, New York, 2nd edition, 2001.

Vojtěch Franc and Václav Hlaváč. *Statistical pattern recognition toolbox for Matlab.* Research Report CTU–CMP–2004–08, Center for Machine Perception, K13133 FEE. Czech Technical University, Prague, Czech Republic, June 2004. http://cmp.felk.cvut.cz/cmp/software/stprtool/index.html [http://cmp.felk.cvut.cz/cmp/software/stprtool/index.html].

Michail I. Schlesinger and Václav Hlaváč. *Ten Lectures on Statistical and Structural Pattern Recognition.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002.

courses/be5b33kui/semtasks/05_ml1/start.txt · Last modified: 2024/05/24 13:17 by xposik