Algorithms for Image Processing

Prof. Andreas Weinmann

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbmn
FACHBEREICH MATHEMATIK
UND NATURWISSENSCHAFTEN

Hints for Problem Sheet 3

Problem 1
k nearest neighbor classifier.

- Significantly reduce training and validation data for developing your code, say 5000 and 1000 items.
- It is sufficient to implement the L^2 distance. Optional: Try to get faster by using the identity
$$\|x-y\|^2 = \langle x,x\rangle - 2\langle x,y\rangle + \langle y,y\rangle$$
(Also search the net for information on that: other people may have the same problems.) Optional: Try out the L^1 distance.
- Looking up the numpy functions “argsort” and “unique” may help.
- For cross validation, using about 5 to 10 folds is OK.
- You can get about 30% to 35% accuracy.

Problem 2
Softmax classifier.

- Looking up the numpy functions “tensordot” for multiplying tensors with several indices may help. (There are also people who think that “einsum” is all you need – maybe together with attention :))
- For training store the loss function through gradient descent and plot it to have a visualization of the decay of the loss function values.
- After some iterations, e.g., every 100th iteration print the loss, so you can stop if things go wrong.
- After some iterations, e.g., every 100th iteration calculate and print the accuracy on the test set (and maybe later in development on the validation set as well.)
- If your accuracy stagnates at about 10% there may be some bug, or your hyperparameters are completely out of range.
- a few thousand iterations can be needed. The iteration number can be a cross validation parameter. Or, you can try out and take enough to be on the save side.
- Most important cross validation parameters are the learning rate and the regularization strength. Before starting a whole evaluation machinery, just playing with these two parameters can give you some intuition.
- For hyperparameter tuning you may use cross-validation, but the weaker version of cross validation, i.e., choosing one training and one validation set is also OK.
- You can get about 35% to 40% accuracy.

Problem 3
Classifiers on pre-selected features.

- A potential choice for the setting of the hog features is

```
hog(image, orientations=9, pixels_per_cell=(8, 8), cells_per_block=(4, 4), visualize=False,
multichannel=True)
```

Try this out and feel free to tune.

- Using 8 or 10 angles for the color histogram works OK.
- You can get about 50% to 55% accuracy for kNN and about 40% to 45% for the softmax classifier. (With the two layer fully connected net waiting for you on the next sheet you can get 60% on these features.)