Exercise Sheet 05

Deadline: 04.06.2018, 1pm
Regulations:

- You need $\geq$ **50**% **of all points** in the weekly exercises and successfully work on a $\geq$ **2 weeks fulltime project** and submit a **written report** in order to pass this class.
- Hand in your solutions in **groups of two or three people**.
- Make sure your code works with python 3.6 or later.
- For exercises that include coding, hand in a **Juypter Notebook** for each exercise containing all your answers, results, plots and code. Separate the different sub-exercises clearly by using headings and describing text. Label each file with the number of the exercise. Before submission, make sure your Jupyter Notebook compiles without any errors when you do Run All. Additionally hand in a PDF version of the Notebook.
- For exercises that can be solved without coding, hand in a **PDF file** for each exercise containing all your answers and results. Separate the different sub-exercises clearly by using headings and describing text. Label each file with the number of the exercise.
- Submit all your results in a **single** .zip archive. The title of the submitted ZIP-file as well as the subject line of your email **must** start with **EX05** followed by the full names of **all group members**. **People not mentioned in the subject will not get credits.**
- Submit all your results to mlcvss18@gmail.com.
- Submissions that do not adhere to the deadline and regulations, or that are chaotic, will not be graded.

Exercise 1. 20 P.

The task is to find the cheapest path on a grid from the top-left to the bottom-right corner which satisfies the following constraint: In each step you can either move to the right, down, or diagonally down-right. You can never move left or up. Consider this problem but assume that the definition of an optimal path is unknown. The task of this exercise is to learn this definition from a given training dataset $\{x, y\}$ of images x and paths y. We will solve this problem using the structured perceptron [1].

a) Load the dataset from the file paths_and_images.h5 and assign 10 images and paths to the test set and the rest to the training set. Plot an example image with corresponding path from the training set.

b) The structured perceptron is a classifier which learns the parameters w from the training set $D = \{x, y\}$. An optimal path $\hat{y}$ can then be obtained from

$$\hat{y} = \arg\max_{y'} w \cdot \phi(x, y') \tag{1}$$

We define the joined features $\phi(x, y)$ as

$$\phi(x, y) = \sum_{l=1}^{L} \phi_l(x, \{p, q\}_l) \tag{2}$$

where $\{p, q\}_l$ are the l-th neighboring points on the path y and

$$\phi(x, \{p, q\}) = \begin{pmatrix} -a \\ -a^2 \\ -b \\ -b^2 \\ -b^3 \end{pmatrix} \tag{3}$$

where

$$a = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2} \tag{4}$$

$$b = |v_p - v_q| \tag{5}$$

and (p_x, p_y) and (q_x, q_y) are the coordinates of the pixels in the image, and $v_{(\cdot)}$ is the gray value of the respective pixel.

From equation (2) it follows that

$$w \cdot \phi(x, y) = w \cdot \sum_{l=1}^{L} \phi_l(x, \{p, q\}_l) = \sum_{l=1}^{L} w \cdot \phi_l(x, \{p, q\}_l) \tag{6}$$

The intuitions behind these features are the following:

- a measures the Euclidean distance of neighboring pixels on the path, which indicates that the path should be short;
- b measures the gray value difference of neighboring pixels, which indicates that the path should not be too bumpy;
- Powers of these features are used to facilitate Taylor approximations to a true but unknown compatibility function.

Explain why we define all features such that they are smaller or equal to zero.

c) Implement and train the structured perceptron algorithm as follows:

1. Initialize $w = \vec{0}$
2. For $t = 1, \ldots, T$ iterations do
   For $d = (x, y)$ in the training set D do
   $\hat{y} \leftarrow \arg\max_{y'} w \cdot \phi(x, y')$
   if $\hat{y} \neq y$
   $w \leftarrow w + \phi(x, y) - \phi(x, \hat{y})$
   end
   end
3. return w

Hint: to compute the $\arg\max_{y'}$ over all possible paths, you can use different strategies. One possibility is to use brute force and try out all paths (computationally costly) or you could use the Viterbi algorithm. Another option is to use beam-search (see e.g. [2]). In beam-search, at each step l' the value $\sum_{l=1}^{l'} w \cdot \phi_l(x, \{p, q\}_l)$ is evaluated for all possible next steps, but only the b best paths are kept at each step. Example for $b = 3$: at the beginning we start from $p = (0, 0)$ and can do 3 possible steps to $q_1 = (0, 1)$, $q_2 = (1, 0)$ or $q_3 = (1, 1)$. Compute and save $w \cdot \phi(x, \{p, q_i\})$ for $i = 1, 2, 3$ and save all three q_i. In the next step, from each of the points (0,1), (1,0) and (1,1) we can continue in 3 different ways, giving us already 9 possible paths. For each of them we compute $w \cdot \phi_1(x, \{p, q\}_1) + w \cdot \phi_2(x, \{p, q\}_2)$ but we save only the 3 paths that obtain the highest value and continue in the next step only from their end points. We continue like this until we reach the lower right corner of the grid.

d) Predict the shortest paths $\hat{y} = \arg\max_{y'} w \cdot \phi(x, y')$ for the test images and plot the images with the predicted and the ground truth paths.

References

[1] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, pages 1–8, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[2] Liang Huang, Suphan Fayong, and Yang Guo. Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 142–151, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.