Homework 5: Self-organizing Maps

Christian Bock
christian.bock@bsse.ethz.ch

Dr. Michael Moor
michael.moor@bsse.ethz.ch

Leslie O'Bray
leslie.obray@bsse.ethz.ch

Prof. Dr. Karsten Borgwardt
karsten.borgwardt@bsse.ethz.ch

Submission deadline: 19.05.2021 at 14.00

Objectives

The goals of this homework are:

- to implement an online version of a self-organizing map (SOM) algorithm.
- to apply your SOM to an artificial dataset.
- to apply your SOM to a real-world data set.

Exercise 1

In this question you will implement an online version of a self-organizing map algorithm. Although there are more sophisticated variants, we recommend the algorithm outlined in the pseudocode below [Ripley, 2007, Venables and Ripley, 2013]. An important point to remember is that the algorithm deals with distances in two spaces, namely, the *grid space* $\mathcal{G}$, and the *feature space* $\mathcal{F}$. We use the terms *buttons* and *prototypes* interchangeably.

Exercise 1.a Implement in Python the online SOM algorithm described in Algorithm 1 using an exponentially decaying learning rate $\alpha(t) = \alpha_{\max} \exp(-\lambda t)$. The file `somutils.py` contains auxiliary functions which you may find useful. In particular, for this exercise you are asked to provide the code for the function `SOM(·)` and for its auxiliary functions. Feel free to create your own Python functions with their own interfaces and adjust the rest of the code appropriately.

Exercise 1.b Use the function `makeSCurve(·)` (provided in `somutils.py`) to create an S-shaped dataset. Use the function `SOM(·)` from Exercise 1.a to create a SOM on this dataset. Use the following parameters when creating the SOM: $p = 15$, $q = 1$, $N = 150$, $\alpha_{\max} = 1$, $\epsilon_{\max} = 3$, and $\lambda = 0.01$. Plot the data with the SOM buttons superimposed using the function `plotDataAndSOM(·)`. The plot should be saved to disk with the name `exercise_1b.pdf` (alternatively, you can save it as `.png`).

Algorithm 1 Online SOM

Input: Data $\{x_1, x_2, \ldots, x_n\}$, grid dimensions p and q, number of iterations N, upper limit for radius, $\epsilon_{\max}$, upper limit for learning rate, $\alpha_{\max}$.
Output: a $p \times q$ grid, and $K = p \cdot q$ buttons $\{m_1, m_2, \ldots m_K\}$

Create the grid and compute the pairwise distances (in $\mathcal{G}$)
Randomly select K out of the n data points as the initial positions (in $\mathcal{F}$) of the buttons $m_1, \ldots, m_K$
Create a vector of size N of values for the radius ϵ, decreasing from $\epsilon_{\max}$ to 1
for t in $1 : N$ **do**
    Initialize/update ϵ
    Choose a random index $i \in \{1, 2, \ldots, n\}$
    Find button m_* that is nearest to x_i (in $\mathcal{F}$)
    Find all grid points in ϵ-neighborhood of m_* (in $\mathcal{G}$)
    Update position (in $\mathcal{F}$) of all buttons m_j in ϵ-neighborhood of m_*, including m_*, by: $m_j \leftarrow m_j + \alpha(t)(x_i - m_j)$
return $\{m_1, m_2, \ldots, m_K\}$, and the $p \times q$ grid

Execise 1.c Write a function `computeError(·)`, to compute the reconstruction error at iteration t. Include its invocation in the function `SOM(·)` (see template of the function as described in Exercise 1.a). Create a plot showing the reconstruction error on the S-shaped dataset from Exercise 1.b. The layout of the plot should have the error on the Y-axis and the iteration number t on the X-axis. Use the same parameters as in Exercise 1.b. The plot should be saved to disk with the name `exercise_1c.pdf` (or as `.png`)

Exercise 2

The data file `crabs.txt` is included in this homework. It contains morphological measurements of *Leptograpsus variegatus* crabs [Campbell and Mahon, 1974]. There are measurements for two types of crabs: blue (`B`) and orange (`O`), as well as for males and females. The dataset contains 200 samples.

sp	sex	index	FL	RW	CL	CW	BD
B	M	1	8.10	6.70	16.10	19.00	7.00
B	M	2	8.80	7.70	18.10	20.80	7.40
B	F	1	7.20	6.50	14.70	17.10	6.10
O	M	1	9.10	6.90	16.70	18.60	7.40
O	F	1	10.70	9.70	21.40	24.00	9.80
O	F	2	11.40	9.20	21.70	24.10	9.70

Figure 1: Format of the `crabs.txt` file. Columns are tab-separated, one row per sample.

The column `sp` refers to the color of the crab, either `B` (blue) or `O` (orange). The `sex` is either `M` (male) or `F` (female). The index column is the ID for the sample within each sp/sex group. The last five columns are measurements in millimeters: `FL` refers to frontal lobe size, `RW` refers to rear width, `CL` and `CW` refer to carapace length and width respectively, and `BD` refers to body depth. These last five columns are the ones you should

use to run your SOM as indicated in Exercise 2.a. The first two columns are needed to plot the results in Exercise 2.b.

Exercise 2.a Using the function `SOM(·)` from Exercise 1.a, compute buttons $\{m_1, m_2, \ldots, m_K\}$. For each data point x_i in the dataset, with $i \in \{1, 2, \ldots, n\}$, assign a label $y_i \in \{1, 2, \ldots, K\}$ to the data point, such that button m_{y_i} is closest to x_i. Save the output as a text file named `output_som_crabs.txt` with the format detailed in Figure 2. Use the following parameters to generate the file: $p = 6$, $q = 8$, $N = 1000$, $\alpha_{\max} = 1$, $\epsilon_{\max} = 3$, and $\lambda = 0.002$. There is no need to compute the reconstruction error.

sp	sex	index	label
B	M	1	9
B	M	2	9
B	F	1	9
O	M	1	9
O	F	1	9
O	F	2	9

Figure 2: Layout of the output file `output_som_crabs.txt`. The column `label` corresponds to the value y_j mentioned before. Columns are tab-separated, one row per sample.

Exercise 2.b Building on the results from Exercise 2.a, write the code for the function `plotSOMCrabs(·)` in `somutils.py`. This function should create a plot showing how the SOM is applied to the data. The plot should look like the one presented in the class lecture (section "1.5 Self-organizing Maps", slide 111, right panel). For convenience, the layout of the plot is shown in Figure 3. In the figure, each circle corresponds to a button and the points inside of it are the data points labeled with that button in Exercise 2.a. Data points are colored according to their `sp` and `sex` values. Note that a small random jitter was added to the data points to avoid superimposing them inside the button. The plot should be saved to disk with the name `exercise_2b.pdf` (or as `.png`)

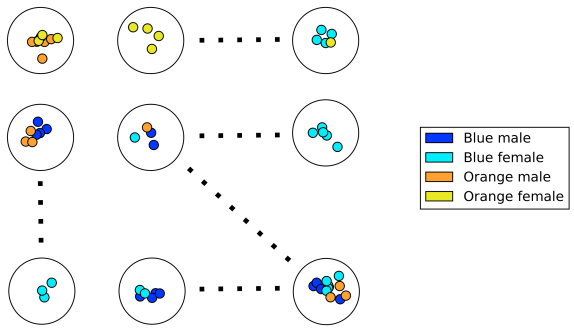

Figure 3: A SOM applied to the crabs dataset. The data points are colored based on their `sp` and `sex` values. The grid should contain $p \times q$ buttons with $p = 6$ and $q = 8$ (See Exercise 2.a for details about the other parameters). The dotted lines in this sample figure represent the missing buttons in the grid.

Command-line arguments

Write a program named `compute_som.py` to perform the tasks previously described. This script will invoke functions from `somutils.py`, i.e. the auxiliary functions should not be in `compute_som.py`. Your program will receive 8 mandatory command-line arguments and 1 optional one:

`--exercise num`: where `num` is either 1 or 2 indicating what exercise the program will solve.

`--outdir path`: is the path to the output directory where the output files for the exercise will be saved.

Parameters for the creation of the SOM:

`--p 99`: number of rows in the grid.

`--q 99`: number of columns in the grid.

`--N 99`: number of iterations.

`--alpha_max 99`: upper limit for learning rate.

`--epsilon_max 99`: upper limit for radius.

`--lamb 0.99`: decay constant (λ) for learning rate decay.

Optional argument, only when `--exercise 2` is used:

`--file path`: the full path to the input file in Exercise 2.

In order to create all the plots and output files required by this assignment, the program will have to be called twice: the first time with `--exercise 1` (to solve the first exercise) and the second time with `--exercise 2`.

Below you have an example of how to invoke the program from the command line for both exercises:

```
$ python compute_som.py \
--exercise 1 \
--outdir /home/hwk5/output/exercise1 \
--p 15 --q 1 --N 150 --alpha_max 1 --epsilon_max 3 \
--lamb 0.01

$ python compute_som.py \
--exercise 2 \
--outdir /home/hwk5/output/exercise2 \
--p 6 --q 8 --N 1000 --alpha_max 1 --epsilon_max 3 \
--lamb 0.002 \
--file /home/hwk5/data/crabs.txt
```

You do not have to check the correctness of the parameters, simply assume that the user will provide valid values.

Grading and submission guidelines

This homework is worth a total of 100 points. Table 1 shows the points assigned to each exercise.

55 pts.	**Exercise 1**	
	40 pts.	Exercise 1.a
	5 pts.	Exercise 1.b
	10 pts.	Exercise 1.c
45 pts.	**Exercise 2**	
	15 pts.	Exercise 2.a
	30 pts.	Exercise 2.b

Table 1: Grading key for Homework 5

Acknowledgements

The first version of this exercise sheet was originally created by Dean Bodenham and Karsten Borgwardt.

References

N. Campbell and R. Mahon. A multivariate study of variation in two species of rock crab of the genus leptograpsus. *Australian Journal of Zoology*, 22(3):417–425, 1974.

B. D. Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.

W. N. Venables and B. D. Ripley. *Modern applied statistics with S-PLUS*. Springer Science & Business Media, 2013.