EM-ALGORITHM FOR A SIMPLE SHAPE MODEL

STATISTICAL MACHINE LEARNING (WS2023)
3. COMPUTER LAB (10P)

You are given a set of binary images, each of which was generated from a common, binary shape in different poses. The poses are unknown. The task is to estimate the shape model.

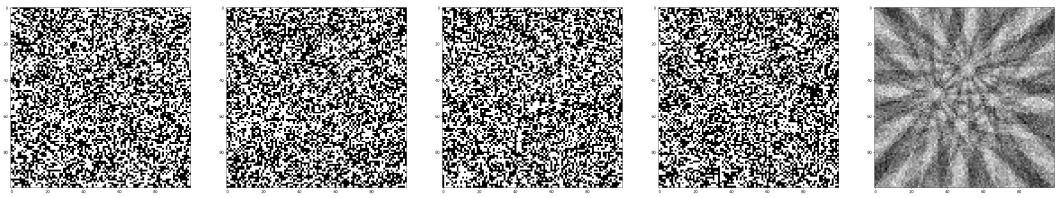

FIGURE 1. From left: four examples from the training set and the average of all training images.

1. NOTATIONS & MODEL

Let us denote the common pixel domain of the images by $D \subset \mathbb{Z}^2$. Binary images are denoted by $x \in \mathcal{B}^D$, where $\mathcal{B}$ denotes the set $\{0, 1\}$. The value of the image x in pixel $i \in D$ is denoted by x_i. The probability distribution for images is parametrised by a binary shape $s \in \mathcal{B}^D$ (with values s_i in pixels $i \in D$) and a pair of (natural) parameters $\eta = (\eta_0, \eta_1)$ for Bernoulli distributions. Recall that the Bernoulli distribution $p_b(z; \eta)$ for $z \in \mathcal{B}$ is given by

$$\log p_b(z) = \eta z - \log(1 + e^{\eta}). \tag{1}$$

The probability distribution for binary images x parametrised by the shape s and the tuple $\eta = (\eta_0, \eta_1)$ is

$$p(x; s, \eta) = \prod_{i \in D} p_b(x_i; \eta_i(s)), \tag{2}$$

i.e. the pixels are statistically independent. The two-dimensional field of parameters $\eta(s) \in \mathbb{R}^D$ is given by

$$\eta_i(s) = \begin{cases} \eta_1 & \text{if } s_i = 1, \\ \eta_0 & \text{if } s_i = 0. \end{cases} \tag{3}$$

Then we can write

$$\log p(x; s, \eta) = \langle x, \eta(s) \rangle - n_0(s) \log(1 + e^{\eta_0}) - n_1(s) \log(1 + e^{\eta_1}), \tag{4}$$

where $n_1(s)$ is the number of foreground pixels of the shape s, i.e. $n_1(s) = \sum_{i \in D} s_i$ and $n_0 = |D| - n_1$.

The training set $\mathcal{T}^m = \{x^j \in \mathcal{B}^D \mid j = 1, \ldots, m\}$ consists of images generated from the shape s in randomly chosen poses. I.e. when generating an image, we first randomly choose one of the poses $r = 1, \ldots, K$ with probabilities π_r, then transform the shape by $s' = T_r(s)$, and finally generate the image from the transformed shape s' using (2). The corresponding probability of the image conditioned on the pose is thus

$$\log p(x \mid r; s, \eta) = \langle x, \eta(T_r s) \rangle - n_0(s) \log(1 + e^{\eta_0}) - n_1(s) \log(1 + e^{\eta_1}), \tag{5}$$

where we assume that the transformations preserve the size of the shape, i.e. $n_1(T_r s) = n_1(s)$.

2. Learning Task

You are given an i.i.d. training set of images $\mathcal{T}^m = \{x^j \in \mathcal{B}^D \mid j = 1, \ldots, m\}$ which has been generated from a model as described above. There are four poses, corresponding to rotations by 0°, 90°, 180° and 270°. The training set contains no pose informations. The model parameters, i.e. the the shape $s \in \mathcal{B}^D$, the 2-tuple $\eta = (\eta_0, \eta_1)$ and the probabilities π_r for choosing one of the four poses are all unknown. Your task is to develop an EM algorithm for learning them from the training set $\mathcal{T}^m$.

3. Assignments

Data: The two datasets `images0.npy` and `images.npy` can be downloaded at the course web-page. Both contain a numpy array with 1000 binary images. The dataset `images0.npy` is provided just for convenience (to test your code for Assignment 1). The main dataset is in `images.npy`.

Assignment 1 (3p)**.** Let us first consider the following simpler task: the ML estimate of the shape s and the 2-tuple (η_0, η_1) in case that all images where generated from the same shape pose. The task reads

$$\frac{1}{m} \sum_{x \in \mathcal{T}^m} \log p(x; s, \eta) \to \max_{s,\eta}. \tag{6}$$

Substitute the model (4) and prove that all you need from the data is the average image $\psi = \frac{1}{m} \sum_{x \in \mathcal{T}^m} x$. Notice that the optimisation task (6) can not be solved by gradient ascent because the shape parameter is discrete. Propose an approximation algorithm that alternates maximisation w.r.t. s and maximisation w.r.t. η. Show that the two subtasks can be solved in closed form. The resulting algorithm should start with an initial estimate of $\eta = (\eta_0, \eta_1)$ and run till convergence. Explain the algorithm you are proposing.

Write a function `shape_mle(avg_image, etas_init)` that implements this algorithm and returns a shape s and a 2-tuple η. Its inputs are: `avg_image` - the average image and `etas_init` - a 2-tuple representing an initial estimate of the Bernoulli distribution parameters. We provide the data set `images0.npy` for testing the function. It contains 1000 images generated from some shape s (which is different from the shape used for the dataset `images.npy`).

We assure you that the foreground pixels of the shape have $p_b(z = 1) > 0.5$, whereas the background pixels have $p_b(z = 1) < 0.5$. This knowledge should be sufficient for choosing a reasonable 2-tuple `etas_init`. Apply the algorithm to the dataset `images0.npy` and report the shape s and the 2-tuple η you obtained as approximation of the ML estimate. (The true shape used for generating this image set is an ellipse)

Assignment 2 (7p)**.** In this assignment you will apply the EM-algorithm to the training data in `images.npy`. Let us denote the auxiliary variables of the EM algorithm by $\alpha_x(r) \geqslant 0$. They should fulfil $\sum_r \alpha_x(r) = 1$ for each image $x \in \mathcal{T}^m$.

E-step: Given the current model parameter estimates for s, η and π, you need to compute

$$\alpha_x(r) = p(r \mid x; s, \eta) \tag{7}$$

for each image $x \in \mathcal{T}^m$. Prove that this posterior probabilities can be computed by

$$\alpha_x(r) = \operatorname{softmax}_r\big(\langle x, \eta(T_r s)\rangle + \log \pi_r\big). \tag{8}$$

Implement a function `posterior_pose_probs(images, ...)` that computes the array of alpha's for all images.

M-step: Fixing the alphas from the E-step, you need to solve the task

$$\frac{1}{m}\sum_{x\in\mathcal{T}^m}\sum_r \alpha_x(r)\big[\log(p(x \mid r; s, \eta) + \log \pi_r\big] \to \max_{s,\eta,\pi}. \tag{9}$$

Show that the maximiser for the pose probabilities π_r can be found in closed form. Give the maximisation task w.r.t. s and η by substituting (5). Prove that all you need from the data is the array

$$\psi = \frac{1}{m}\sum_{x\in\mathcal{T}^m}\sum_r \alpha_x(r) T_r^T x \tag{10}$$

Show that the optimisation task (9) can be solved by the function `shape_mle()` which you have implemented in Assignment 1.

It remains to find a suitable initialisation for the algorithm. You can choose the initial 2-tuple η as you did in Assignment 1. The initial pose probabilities can be chosen to be uniform. The initial shape s can be a random binary image. Last but not least, you need to choose an appropriate stopping criterion for the EM algorithm. Explain your choice.

Implement the EM-algorithm. You may consider to visualise the "average" image ψ from (10) before each M-step when running the EM algorithm. Report the final estimate of the shape s, the prior pose probabilities π_r, and the 2-tuple of etas.

Final Hints:

(1) All necessary functions can be found in numpy and scipy.

(2) We strongly discourage using loops over images and loops over image pixels. Use `numpy` arrays and array operations instead.
(3) Just for reference: my complete code for the EM algorithm (IPython notebook) has less than 80 lines. Its runtime is less than 3 sec.