Exercises for the course
Machine Learning for Data Science
Winter Semester 2024/25

G. Montavon
Institute of Computer Science
Department of Mathematics and Computer Science
Freie Universität Berlin

Exercise Sheet 2 (theory part)

Exercise 1: Concentration of Squared Distances $(10+10+10$ P$)$

In this exercise, we would like to study analytically the concentration of (squared) distances in high dimensions, which was discussed in the lecture. Let $\boldsymbol{x} \in \mathbb{R}^d$ denote an input example. Input examples are drawn iid. from the distribution $\boldsymbol{x} \sim \mathcal{N}(0, I)$, where I is an identity matrix of size $d \times d$. We study the function

$$y(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|^2.$$

measuring the squared Euclidean distance between pairs of points drawn from the distribution above.

(a) Express the mean of y as a function of the number of input dimensions d.

(b) Express the standard deviation of y as a function of d.

(c) Show that the ratio $\text{std}[y]/\text{E}[y]$ is given by $\sqrt{\frac{2}{d}}$, and that therefore, square distances concentrate more as d grows.

Exercise 2: Gradient of T-SNE $(15+15$ P$)$

T-SNE is an embedding algorithm that operates by minimizing the cross-entropy between two discrete probability distributions p and q representing pairwise similarities of data points in the input space and in the embedding space respectively. Specifically, p_{ij} is a probability value quantifying how similar the points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are. Likewise, q_{ij} is a probability value quantifing how similar the same points are in embedded space (denoted by y_i and y_j). We assume the dataset consists of N instances, and that the embedding space is for simplicity one-dimensional.

The relation between the coordinates in embedded space and the objective to minimize (the cross-entropy between p and q) is given by the following sequence of computations:

$$\Delta_{kl} = y_k - y_l \qquad s_{kl} = \Delta_{kl}^2 \qquad z_{kl} = (1 + s_{kl})^{-1} \qquad q_{ij} = \frac{z_{ij}}{\sum_{kl} z_{kl}} \qquad C = -\textstyle\sum_{ij} p_{ij} \log q_{ij}$$

where $\sum_{ij}\{\cdot\}$ is a shortcut notation for $\sum_{i=1}^N \sum_{j=1}^N \{\cdot\}$, and likewise for $\sum_{kl}\{\cdot\}$. Note that the cost function can be expressed as a function of the coordinates, i.e. $C(y_1, \dots, y_N)$. In practice, this function will be minimized using gradient descent. Before resorting to automatic differentiation techniques for such purpose, however, it is useful to derive the gradient analytically in order to verify its properties.

(a) Using the decomposition of the computation above, *draw* a computational graph connecting a given point in y_m embedded space to the quantity to minimize C, and *state* the multivariate chain rule for computing the gradient $\partial C / \partial y_m$.

(b) Starting from the statement of the multivariate chain rule, *derive* the following expression of the gradient:

$$\frac{\partial C}{\partial y_m} = 2 \sum_{kl} (p_{kl} - q_{kl}) \cdot \frac{\Delta_{kl}}{1 + \Delta_{kl}^2} \cdot (\delta_{km} - \delta_{lm}) \qquad \text{where} \qquad \delta_{kl} = \begin{cases} 1 & k = l \\ 0 & k \neq l \end{cases}$$

Note that one can verify from this equation that all terms are bounded. Also, the p's and q's as well as the δ's are normalized (they all sum to 1). Gradients only tend to vanish in presence of large Δs.