Lehrstuhl Computergrafik
Prof. Dr.-Ing. Hendrik P.A. Lensch
Prof. Dr.-Ing. Andreas Geiger

Machine Learning in Graphics & Vision

Exercise 5

Release date: Thu, 24. June 2020 - **Deadline for Homework: Wed, 08. July 2020 - 21:00**

So far, we have mainly considered *supervised learning* where we provide both inputs and desired outputs to the model during training. In this exercise, we consider *unsupervised learning* where the goal is to make a machine learning model learn about patterns in the data on its own. We consider both a classical approach, *Principal Component Analysis* and a more modern approach, *Variational Autoencoders*[1] (VAEs), in this exercise. Both models can be considered as probabilistic models that are trained by maximizing (a lower bound to) the log-likelihood

$$\frac{1}{N}\sum_{i=1}^{N}\log p_\theta(x_i) \tag{1}$$

of a parametric probabilistic model $p_\theta(x)$. For PCA, $p_\theta(x)$ is a multivariate Gaussian distribution. For VAEs, $p_\theta(x)$ is given by the marginal distribution of a latent variable model $p_\theta(x,z) = p_\theta(x \mid z)p(z)$.

Exercises

For this exercise you need to submit a **.zip** file containing your report as a **.pdf** file and your code (namely the files `report.pdf, utils.py, exercise_5_1.py, exercise_5_2.py`). Make sure that your report is self contained and includes all your results and visualizations. **You will only get points for results that are included in the report. Please use the provided code templates for these exercises. Do not use jupyter-notebooks.**
For the visualizations, you can use the functions provided in the `utils.py` script.

5.1 Principal Component Analysis (2 + 2 + 2 + 2 + 2 Points)

In this exercise, we apply PCA to the Fashion-MNIST dataset.

a) Compute the mean vector of the Fashion-MNIST dataset and apply PCA to the *centered* dataset (subtract the mean from the samples before applying PCA). Fill in the function

```
compute_pca(x)
```

where x is $N \times K$ input array for N data points of dimensionality K. Test your implementation and visualize the first 5 principal components.

b) Plot the percentage of explained variance over the number of principal components. How many components do you need to explain 50%, 90%, 95%, 99% of the variance?

c) We can use PCA to compress test data. For each sample in the *test set* compute the coefficients belonging to each principal component. Reconstruct each test sample using only the first 5 principal components from the previous exercise (don't forget to center your data using the mean vector). Report the averaged mean-squared error (MSE) over all test samples. Visualize the reconstructions of the first five test samples. What is the compression ratio?

[1] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." Proceedings of the 2nd International Conference on Learning Representations (ICLR), 2014.

d) We can use PCA to sample from the probabilistic model obtained by fitting a Gaussian distribution to the training set. To do so, we sample $\epsilon \in \mathbb{R}^K$ from $\mathcal{N}(0, I)$ and then compute $x = \mu + \sum_{k=1}^{K} \sqrt{s_k} v_k \epsilon_k$ where μ denotes the mean vector, s_k denote the k^{th} singular values of the covariance matrix and v_k the k^{th} principal component. Draw 5 samples from this probabilistic model and visualize them. Do they look realistic?

e) Fashion-MNIST contains 10 different classes and is therefore highly varied. Can you obtain better results by applying PCA only to the *Sneaker*-subset (with class id 7) of this dataset? Return your code on this subset and report your results.

5.2 Variational Autoencoder (4 + 4 + 2 Points)

We now want to improve our model by applying a Variational Autoencoder to the dataset. Recall that VAEs are trained by maximizing the so-called *Evidence Lower Bound* (ELBO) to the log-likelihood which is given by

$$\frac{1}{N}\sum_{i=1}^{N} \log p_\theta(x_i) \geq \frac{1}{N}\sum_{i=1}^{N} \mathrm{E}_{q_\phi(z|x_i)}\left[\log p_\theta(x_i \mid z)\right] - \mathrm{KL}(q_\phi(z \mid x_i) \mid p(z)) \tag{2}$$

with an additional inference model (the encoder) $q_\phi(z \mid x)$. Here, the (negative of the) first term can be interpreted as a reconstruction loss and the second one as a regularizer. Usually, $p(z)$ is a standard Gaussian distribution and $q_\phi(z \mid x)$ is a Gaussian distribution with diagonal covariance matrix. Recall that the KL-divergence between a Gaussian distribution $q_\phi(z \mid x)$ with mean μ and diagonal covariance matrix with diagonal entries σ^2 is given by

$$\mathrm{KL}(q(z \mid x) | p(z)) = \frac{1}{2}\sum_k \left(-\log(\sigma_k^2) - 1 + \sigma_k^2 + \mu_k^2\right) \tag{3}$$

a) Complete the function

```
compute_kl
```

which computes the KL-divergence between a multivariate Gaussian distribution with diagonal covariance matrix and a unit Gaussian distribution. Test your implementation by calling

```
test_kl
```

with different values for μ and σ.

b) Implement a fully connected VAE where both the encoder and decoder have two hidden layers with 512 units each. Use RELU-activations and a latent dimension of 5. Use cross-entropy (with logits) as reconstruction error[2] and train the model for 10 epochs. How do the different losses behave during training? What do you observe?

c) Use the trained VAE to compress the test data using the inference model $q_\phi(z \mid x)$. Compute the average mean-squared error for the corresponding reconstructions (using $p_\theta(x \mid z)$) and visualize the first 5 reconstructions. How do the reconstructions compare to the ones obtained using PCA? Sample from the learned distribution and visualize 5 samples. Do they look realistic?

[2] This corresponds to using independent Bernouille variables for the conditional distribution $p_\theta(x \mid z)$