Appendix B

Shape Prior

We complement the discussion of using variational auto-encoders as shape prior with a simple, linear alternative: probabilistic principal component analysis (PCA). Originally, we used probabilistic PCA to perform experiments on our 2D dataset as presented in Appendix E.

B.1 Probabilistic Principal Component Analysis

Again, we assume a flattened version $y \in \mathbb{R}^R \simeq \mathbb{R}^{H \times W \times D}$ for simplicity. We first remind the reader of general, non-probabilistic PCA following [Bis06, Section 12.1]:

Example B.1 *The goal of PCA is to find a linear mapping from $y \in \mathbb{R}^R$ to a lower-dimensional latent code $z \in \mathbb{R}^Q$ that captures as much variance as possible. Considering a one-dimensional latent space $z \in \mathbb{R}$ ($Q = 1$), we are looking for a vector $u \in \mathbb{R}^R$ that maximizes the variance captured in z:*

$$\mathrm{Var}[z] = \mathrm{Var}[u^T y].$$

Defining the mean μ as

$$\mu = \frac{1}{M} \sum_{m=1}^{M} y_m, \tag{B.1}$$

this can be written as

$$\mathbb{E}[u^T (y - \mu)(y - \mu)^T u] = u^T \Sigma u \tag{B.2}$$

with Σ being the corresponding covariance matrix. As Σ is positive semi-definite, its eigenvalues are all real and positive [MN99, Section 1.13, Theorem 4 and Theorem 8]. Because the length of u does not matter, we require $\|u\|_2 = 1$ and can choose it as the eigenvector corresponding to the largest eigenvalue λ. Then

$$u^T \Sigma u = \lambda u^T u = \lambda,$$

and $\mathrm{Var}[z]$ is maximal.

The above idea can be generalized to Q-dimensional latent spaces. Then the linear mapping $U \in \mathbb{R}^{R\times Q}$ is found by computing the eigenvalue decomposition [MN99, Section 1.14, Theorem 13] of Σ:

$$\Sigma = V\Lambda V^T \tag{B.3}$$

where $\Lambda \in \mathbb{R}^{R\times R}$ is a diagonal matrix containing the eigenvalues – sorted from largest to smallest eigenvalue – and $V \in \mathbb{R}^{R\times R}$ is an orthogonal matrix containing the corresponding eigenvectors. Taking U as the first Q eigenvectors (corresponding to the Q largest eigenvalues), i.e.

$$U = V_Q := \begin{bmatrix} v_1 & \dots & v_Q \end{bmatrix},$$

yields the linear mapping maximizing the variance and thereby also minimizing the reconstruction error [Bis06, Section 12.1].

The outline from the above example can be stated more precise in terms of an encoding transformation and a decoding transformation – which will correspond to the recognition model and generative model in probabilistic PCA:

Definition B.1 *Given data $\mathcal{Y} = \{y_1, \dots, y_M\} \subseteq \mathbb{R}^R$ with mean μ and covariance matrix Σ as defined according to Equations* (B.1) *and* (B.2)*, we define*

$$U = V_Q := \begin{bmatrix} v_1 & \dots & v_Q \end{bmatrix} \in \mathbb{R}^{R\times Q}$$

using the first Q eigenvectors $v_1, \dots, v_Q$ obtained from the eigenvalue decomposition $\Sigma = V\Lambda V^T$. Then, PCA defines an encoding transformation

$$z = U^T(y - \mu)$$

and a decoding transformation

$$\tilde{y} = Uz + \mu.$$

The problem with general PCA is that no generative model is included. Probabilistic PCA wraps a probabilistic interpretation around the linear encoding and decoding transformations. Following [Bis06, Section 12.2] and [TB99], we still assume a linear model

$$y = Uz + \mu + \epsilon$$

with $\epsilon \sim \mathcal{N}(\epsilon; 0, \sigma^2 I_R)$ and $U \in \mathbb{R}^{R\times Q}$. With a unit Gaussian prior $p(z) = \mathcal{N}(z; 0, I_Q)$, this implicitly defines a generative model: sample $z \sim p(z)$ and

$$y \sim p(y|z) = \mathcal{N}(y; \mu, Uz + \mu, \sigma^2 I_R). \tag{B.4}$$

Given the generative model, we also need the recognition model, *i.e.* the posterior $p(z|y)$, representing the encoding transformation. This can be easily derived using the following result [Bis06, Section 2.3]:

Lemma B.1 *Given Gaussian distributions $p(z)$ and $p(y|z)$ with parameters*

$$p(z) = \mathcal{N}(z; 0, I)$$
$$p(y|z) = \mathcal{N}(y; Uz + \mu, \sigma^2 I)$$

then the posterior $p(z|y)$ is given as

$$p(z|y) = \mathcal{N}(z; S^{-1}U^T(y - \mu), \sigma^{-2}S^{-1})$$

with $S = U^T U + \sigma^2 I_Q$.

Proof: See [Bis06, Section 2.3]. ∎

Determining the parameters U, μ and σ^2 would involve maximizing the likelihood

$$p(y) = \int p(y|z)p(z)dz.$$

As all involved distributions are Gaussians, the marginalization is again a Gaussian [Bis06, Section 2.3] with mean

$$\mathbb{E}[y] = \mathbb{E}[Uz + \mu + \epsilon] = \mu$$

and covariance matrix

$$\begin{aligned}\mathrm{Cov}[y] &= \mathrm{Cov}[Uz + \mu + \epsilon, Uz + \mu + \epsilon] \\ &= \mathrm{Cov}[Uz + \epsilon, Uz + \epsilon] \\ &= \mathbb{E}[(Uz + \epsilon)(Uz + \epsilon)^T] \\ &= \mathbb{E}[Uzz^T U^T] + \mathbb{E}[\epsilon\epsilon^T] = UU^T + \sigma^2 I_R =: S\end{aligned}$$

Here, we used that $p(z)$ is a standard Gaussian with zero mean and unit variance and $p(y|z)$ takes the form in Equation (B.4). Maximizing the likelihood is equivalent to minimizing the negative log-likelihood:

$$\begin{aligned}\mathcal{L}(U, \mu, \sigma^2) &= -\sum_{m=1}^{M} \ln \mathcal{N}(y_m|\mu, UU^T + \sigma^2 I_R) \\ &= \text{const} + \frac{M}{2}\ln|S| - \frac{1}{2}\sum_{m=1}^{M}(y_m - \mu)^T S^{-1}(y_m - \mu).\end{aligned}$$

Considering the gradient with respect to μ and solving for $\nabla_\mu \mathcal{L} = 0$ yields:

$$\nabla_\mu \mathcal{L} = \sum_{m=1}^{M}(y_m - \mu)S^{-1} \stackrel{!}{=} 0 \quad \Leftrightarrow \quad \mu = \frac{1}{N}\sum_{m=1}^{M} y_m.$$

Regarding Σ, it is easier to first rewrite the log-likelihood

$$\begin{aligned}\mathcal{L}(U,\mu,\sigma^2) =& \text{const} + \frac{M}{2}\ln|S| + \frac{1}{2}\text{Tr}\left(\sum_{m=1}^{M}(y_m-\mu)(y_m-\mu)^T S^{-1}\right)\\ =& \text{const} + \frac{M}{2}\left(\ln|S| + \text{Tr}(\Sigma S^{-1})\right)\end{aligned}$$

where Σ is the data covariance matrix:

$$\Sigma := \frac{1}{M}\sum_{m=1}^{M}(y_m-\mu)(y_m-\mu)^T.$$

Taking the derivative with respect to Σ is more involved and requires some well-known matrix derivative identities [MN99, Section 8]:

Lemma B.2 *Let S, Σ be square, symmetric matrices; then it holds*

$$\begin{aligned}\frac{\partial \ln|S|}{\partial S} &= S^{-T} = S^{-1}\\ \frac{\partial \text{Tr}(\Sigma S^{-1})}{\partial S} &= -(S^{-1}\Sigma S^{-1})^T = -S^{-1}\Sigma S^{-1}\end{aligned}$$

Proof: A proof of the first identity can be found in [MN99, Section 8.3, Theorem 1]; the second identity follows from [MN99, Section 8.2; Section 8.4, Theorem 3]. ∎

With the above lemma we can now use the chain rule to derive

$$\nabla_S \mathcal{L} = M\left[S^{-1}U - S^{-1}\Sigma S^{-1}U\right] \stackrel{!}{=} 0$$

which leads to

$$U = \Sigma S^{-1}U. \tag{B.5}$$

At this point, there are three different cases. The first, $U = 0$ is trivial and not informative. The second, $S = \Sigma$, implies that the observed covariance is exact which is undesirable in the presence of additive noise [TB99]. Therefore, the third case is the most interesting one: $U \neq 0$ and $S \neq \Sigma$. Considering the singular value decomposition

$$U = V\Lambda V'^T$$

(note that Λ holds the singular values here, in contrast to Equation (B.3)), using $S = UU^T + \sigma^2 I_R$ and substituting into Equation (B.5):

$$\begin{aligned}& V\Lambda V'^T = \Sigma(V\Lambda V'^T(V\Lambda V'^T)^T + \sigma^2 I)^{-1}V\Lambda V'^T\\ \Leftrightarrow\quad & V\Lambda = \Sigma(V\Lambda^2 V^T + \sigma^2 I)^{-1}V\Lambda\\ \Leftrightarrow\quad & V(\Lambda^2 + \sigma^2 I) = \Sigma.\end{aligned}$$

From the last identity, it follows that with $\Lambda = \text{diag}(\lambda_i)$ and for a specific $\lambda_i \neq 0$ it needs to hold

$$\Sigma v_i = (\sigma^2 + \lambda_i^2) v_i$$

meaning that v_i is an eigenvector of Σ, *i.e.* the data covariance matrix, with corresponding eigenvalue $(\sigma^2 + \lambda_i^2)^{\frac{1}{2}}$. A solution for U might therefore be

$$U = V_Q (\Lambda_Q - \sigma^2 I_Q)^{\frac{1}{2}}.$$

Note that this solution is not unique; for any orthogonal matrix U', UU' is a solution, as well. Now only σ^2 is left to be determined. However, as we merely approximate σ^2 in practice and it is less relevant to understand the general idea of probabilistic PCA, we refer to [TB99]. Overall, the derivation leads to:

Definition B.2 *Given data $Y = \{y_1, \ldots, y_M\} \subseteq \mathbb{R}^R$ with mean μ and covariance matrix Σ, we define $U = V_Q(\Lambda_Q - \sigma^2 I_Q)^{\frac{1}{2}}$ with $\Sigma = V \Lambda V^T$ being the eigenvalue decomposition of Σ, and*

$$\sigma^2 = \frac{1}{R - Q} \sum_{i=Q+1}^{R} \lambda_i$$

where $\lambda_1, \ldots, \lambda_R$ are the eigenvalues of Σ in decreasing order. Then, probabilistic PCA defines as recognition model

$$p(z|y) = \mathcal{N}(z; S^{-1} U^T (y - \mu), \sigma^{-2} S^{-1})$$

with $S = UU^T + \sigma^2 I_R$ and as generative model

$$p(z) = \mathcal{N}(z; 0, I_Q)$$
$$p(y|z) = \mathcal{N}(y; Uz + \mu, S).$$

B.1.1 Practical Considerations

Due to a simple result from linear algebra non-probabilistic PCA as introduced in Example B.1 can be implemented in tow different ways; following [MN99, Section 1.16]:

Lemma B.3 *Let*

$$Y = \begin{bmatrix} y_1 & \ldots & y_M \end{bmatrix} \in \mathbb{R}^{R \times M}$$

be the data matrix, μ the corresponding mean and Σ the covariance matrix. Then, $\Sigma = \overline{Y}\overline{Y}^T$ where $\overline{Y}$ is the centered data matrix

$$\overline{Y} = \begin{bmatrix} y_1 - \mu & \ldots & y_M - \mu \end{bmatrix}$$

Then, the singular value decomposition of $\overline{Y} = V \Lambda V'^T$ leads to the eigenvalue decomposition

$$\Sigma = \overline{Y}\overline{Y}^T = V \Lambda^2 V^T,$$

meaning that the singular values of $\overline{Y}$ *are the square roots of the eigenvalues of* Σ.

Proof: The result follows directly from

$$\Sigma = \overline{Y}\overline{Y}^T = V\Lambda V'^T(V\Lambda V'^T) = V\Lambda^2 V^T.$$

More details can be found in [MN99, Section 1.16]. ∎

The lemma implies that we can use either the eigenvalue decomposition of the covariance matrix $\Sigma = V\Lambda V^T$, or the singular value decomposition of the centered data matrix $\overline{Y} = V\Lambda^2 V'^T$. Usually, the latter approach is faster and more memory efficient as it avoids explicitly computing the covariance matrix. For non-probabilistic PCA, only the first Q eigenvalues and eigenvectors are required; efficient algorithms for this case are available, *e.g.* see [GL13]. For probabilistic PCA, all eigenvalues are required in order to compute σ^2. In practice, for large data matrices (*e.g.* R and M in the order of ten thousands), we compute the $Q' > Q$ largest eigenvalues. Pick the Q largest eigenvalues and eigenvectors to form U and the remaining to approximate σ^2.