Imputing Structured Missing Values in Spatial Data with Clustered Adversarial Matrix Factorization

Qi Wang[1], Pang-Ning Tan[1], Jiayu Zhou[1]
[1]Computer Science and Engineering, Michigan State University, East Lansing, MI 48824.

I. Supplementary

A. Synthetic data generation for setting1.

First, we create the latent factor $V \in \mathbb{R}^{r \times n}$ based on c Gaussian distributions (each Gaussian distribution generates data for one spatial cluster), where c is the number of clusters. We set $c = 3$ for all the synthetic data. All the Gaussian distributions have the same variance but different means. The variance we use is $0.01 \times I_{r \times r}$. The means for three Gaussian distributions are $[\vec{\mathbf{0}}, \vec{\mathbf{0.5}}, \vec{\mathbf{1}}]$, where $\vec{\boldsymbol{x}}$ means a vector of x. The columns of V are sampled from those 3 Gaussian distributions. From each spatial cluster, we sample the same number of samples. Denote this number as n_c. Second, we create $U \in \mathbb{R}^{d \times r}$ by randomly sampling all the entries from $0.1 \times \mathcal{N}(0,1)$. Then, X is synthesized by $UV + Noise$, where $Noise = 0.0005 \times \mathcal{N}(0,1)$. In this experiment, we set $n_c = 500$, $d = 50$ and $r = 25$. To create structured missing values, we first partition the data into two equal parts. Then, we pick one part and let the entries whose values are within certain range to be missing (this range is determined by the missing rate). For example, when missing rate is 0.6, this range is $[mean + 0.2, mean + 0.8] \cup [mean - 0.8, mean - 0.2]$, where $mean$ is the mean of all the entries of X. d_{ij} is a predefined similarity between the i-th sample and the j-th sample to represent the spatial relationship between them. We use the first two principal components of V as the longitudes and latitudes (locations) of the samples. Denote them as loc_i for the i-the sample. In this experiment, if the j-th sample is in the 20 nearest neighbors of the i-th sample, we set $d_{ij} = exp(-\alpha \| loc_i - loc_j \|_2^2)$, where α is a tunable parameter. If the j-th sample is not in the 20 nearest neighbors of the i-th sample, we set $d_{ij} = 0$.

B. Parameters setting for setting 1.

There are 5 parameters need to be tuned. In order to tune them efficiently, we first train a clustered matrix factorization model using SVD as initialization and tune γ_1, γ_2 and γ_3. Those three parameters are tuned over $\{1e-4, 1e-3, 1e-2, 1e-1, 1\}$. After finishing tune the clustered matrix factorization model, we use this model to initialize our model. When tune our model, we fix γ_1, γ_2 and γ_3 and just tune λ. λ is tuned over {1e-2, 1e-1, 1, 5, 10, 15, 20}. The weight of the Gaussian kernel, α, is also a tunable parameter. To reduce the number of parameters to be tuned, we fix α as $10/\max(1, dist_m)$, where $dist_m := \max_{i,j} \| loc_i - loc_j \|_2^2$, and i and j denote the i-th sample and the j-th sample. Despite those parameters, the structure of the discriminator also need to be tuned. In our experiment, the discriminator is a fully-connected DNN with several nonlinear layers following a linear layer and a softmax layer. The softmax layer is the output layer which outputs the probability. For the nonlinear layers, we use ReLU as activation function and tune the layer number over {2, 3, 4, 5}. The neuron number for all the hidden layers are set to be the same and tuned over {128, 256, 512}.