A Proof of Theorem 1

Firstly, we give the definition of f-divergence.

Definition 1 (f-divergence) *The f-divergence between two probability density functions (pdf) p and q is defined as,*

$$\mathbb{D}_f(p\|q) = \mathbb{E}_q\left[f\left(\frac{p}{q}\right)\right],$$

where $f : [0, \infty) \to \mathbb{R}$ is a convex function and $f(1) = 0$.

As shown in [1], since partition functions for $\phi(x, m; \theta)$ and $\phi(x; \theta)$ are the same, we have the following factorization,

$$\phi(x, \boldsymbol{m}; \theta) = \phi(x; \theta)p(\boldsymbol{m} \mid x; \theta).$$

The difference between the two objective becomes,

$$\begin{aligned}
&\mathcal{L}_{\mathrm{VCNCE}}(\theta, \varphi) - \mathcal{L}_{\mathrm{CNCE}}(\theta)\\
=&2\mathbb{E}_{xy}\mathbb{E}_{q(\boldsymbol{m};\varphi)}\left\{\log\left[1+\frac{\phi(y;\theta)p_c(x\mid y)q(\boldsymbol{m};\varphi)}{\phi(x,\boldsymbol{m};\theta)p_c(y\mid x)}\right]-\log\left[1+\frac{\phi(y;\theta)p_c(x\mid y)}{\phi(x;\theta)p_c(y\mid x)}\right]\right\}\\
=&2\mathbb{E}_{xy}\mathbb{E}_{q(\boldsymbol{m};\varphi)}\log\frac{\phi(x,\boldsymbol{m};\theta)\phi(x;\theta)p_c(y\mid x)+\phi(y;\theta)p_c(x\mid y)\phi(x;\theta)q(\boldsymbol{m};\varphi)}{\phi(x,\boldsymbol{m};\theta)\phi(x;\theta)p_c(y\mid x)+\phi(y;\theta)p_c(x\mid y)\phi(x,\boldsymbol{m};\theta)}\\
=&2\mathbb{E}_{xy}\mathbb{E}_{q(\boldsymbol{m};\varphi)}\log\frac{p(\boldsymbol{m}\mid x;\theta)\phi(x;\theta)p_c(y\mid x)+\phi(y;\theta)p_c(x\mid y)q(\boldsymbol{m};\varphi)}{p(\boldsymbol{m}\mid x;\theta)\phi(x;\theta)p_c(y\mid x)+\phi(y;\theta)p_c(x\mid y)p(\boldsymbol{m}\mid x;\theta)}\\
=&2\mathbb{E}_{xy}[\mathbb{D}_{f_{xy}}(p(\boldsymbol{m}\mid x;\theta)\|q(\boldsymbol{m}))],
\end{aligned}$$

where

$$f_{xy}(u) = \log\left(\frac{\kappa_{xy}+u^{-1}}{\kappa_{xy}+1}\right),$$

with $\kappa_{xy} = \frac{\phi(x;\theta)p_c(y|x)}{\phi(y;\theta)p_c(x|y)}$. It is straightforward to verify that $f(1) = 0$. The derivatives of f is

$$f'(u) = -\frac{1}{u^2\kappa+u}, \quad f''(u) = \frac{2u\kappa+1}{(u^2\kappa+u)^2}.$$

Since κ and u are positive, f is a convex function. Therefore, f satisfy the requirements of f-divergence.

B Proof of Corollaries 1 and 2

Corollary 1 is a straightforward consequence of Theorem 1. Since the f-divergence becomes zero if and only if the two distributions are identical, we have,

$$\mathcal{L}_{\mathrm{VCNCE}}(\theta, \varphi) = \mathcal{L}_{\mathrm{CNCE}}(\theta) \iff q(\boldsymbol{m}; \varphi) = p(\boldsymbol{m} \mid x; \theta).$$

Moreover, since the f-divergence is positive and Theorem 1, we have

$$p(\boldsymbol{m} \mid x; \theta) = \arg\min_{q(\boldsymbol{m};\varphi)} \mathcal{L}_{\mathrm{VCNCE}}(\theta, q(\boldsymbol{m}; \varphi)).$$

Then, plugging the optimal distribution gives the tight bound, we have,

$$\min_\theta \mathcal{L}_{\mathrm{CNCE}}(\theta) = \min_\theta \min_{q(\boldsymbol{m};\varphi)} \mathcal{L}_{\mathrm{VCNCE}}(\theta, \varphi).$$

C Experimental details

C.1 Simulation study

Tensors with non-Gaussian distributions For both GPTF and our model, we set batch size to 1000 and run 500 epochs with Adam optimizer. The initial learning rate is 1e−3 and subsequently reduced by 0.3 at $60\%, 75\%$ and 90% of the maximum epochs. Moreover, the rank is set to 3 for both models. For GPTF, radial basis function (RBF) kernel with band width 1.0 is used, where 100 inducing points is adopted for approximation. For the conditional distribution $p(x_{\mathbf{i}} \mid \boldsymbol{m}_{\mathbf{i}}) = \mathcal{N}(x_{\mathbf{i}} \mid f(\boldsymbol{m}_{\mathbf{i}}), \sigma^2)$ in GPTF, σ is fixed and chosen as the sample standard variance. For our model, we use 5 hidden layers of width 64 for both g_1, g_3 and g_4 defined in Section 3. g_2 is a summation layer. We use ELU activation for non-linearity. For the VCNCE loss, the conditional noise distribution is set as $p_c(y \mid x) = \mathcal{N}(y \mid x, 0.3^2)$ and $\nu = 10$ noise samples are used for each data point.

Continuous-time tensors The data sizes and optimization parameters are the same with the previous simulation. The rank of all models are set to 3. For NONFAT, 100 inducing points are used to approximate the kernel function. We run the NONFAT model for 5000 epochs because we find that the algorithm converges very slowly. Other hyper-parameters are chosen by their default settings. For BCTT, we do not modify their code and settings. For our model, we use 3 hidden layers of length 64 with ELU activation. The conditional noise distribution in the VCNCE loss is set to $p_c(y \mid x) = \mathcal{N}(y \mid x, 1)$ and $\nu = 20$ noise samples are used for each datum.

C.2 Tensor completion

For all datasets, when training our model, we scale the data to $[0, 1]$ based on the *training* data. For testing, we multiply the scale statistic computed by the training data and evaluate the performance on the original domain. We do not employ such data normalization for baselines models, because that will influence their default settings.

C.2.1 Sparse tensor completion

For both *Alog* and *ACC*, the batch size is set to 1000. We run 1000 epochs for *Alog* and *100* epochs for *ACC* due to their different sample numbers. For *Alog* dataset, we add i.i.d. Gaussian noises from $\mathcal{N}(0, 0.05^2)$ during training, while for *ACC*, the standard variance is set to 0.02. The Adam optimizer is used with learning rate chosen from $\{1\mathrm{e}{-2}, 1\mathrm{e}{-3}, 1\mathrm{e}{-4}\}$. We also use gradient clip with maximum infinity norm of 2.0 for training stability. Moreover, we use learning rate scheduler by reducing the initial learning rate by 0.3 at 40%, 60%, and 80% of the total iterations. For both datasets, we use 2 hidden layers of length 50 with ELU activation for g_1, g_3 and g_4 for our model. For the VCNCE loss, we set $\nu = 20$ noise samples with noise variance tuned from $\{0.3^2, 0.5^2, 0.8^2, 1.0^2\}$. In practice, we find that the noise variance is influential to the final performance, even we are using conditional noises. However, with VCNCE, there is only one hyper-parameter for the noise distribution. While for CNCE, one may need to tune both mean and variance of the noise.

C.2.2 Continuous-time tensor completion

For *Air* and *Click* datasets, we set batch size to 128. We run 400 epochs for *Air* and 200 epochs for *Click* due to their different data sizes. For *Alog* dataset, we add i.i.d. Gaussian noises from $\mathcal{N}(0, 0.05^2)$ during training, while for *ACC*, the variance is set as 0.15^2. To encode the temporal information into the energy function, we use the sinusoidal positional encoding, as described in Section 3. Other settings are the same with Appendix C.2.1.

It should be noted that we use the standard definition of root mean square error (RMSE) and mean absolute error (MAE), namely,

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{N}}, \quad \mathrm{MAE} = \frac{\sum_{i=1}^{N}|x_i - \hat{x}_i|}{N},$$

where x_i is the ground truth and $\hat{x}_i$ is the estimate. Therefore, the results are different from those presented in [2], where the authors used *relative* versions of RMSE and MAE,

$$\mathrm{RMSE} = \sqrt{\sum_{i=1}^{N}\frac{(x_i - \hat{x}_i)^2}{x_i^2}}, \quad \mathrm{MAE} = \sum_{i=1}^{N}\frac{|x_i - \hat{x}_i|}{|x_i|}.$$

We modify the evaluation part of their code[1] and report the results.

[1] https://github.com/wzhut/NONFAT

C.3 Ablation study on the objective function

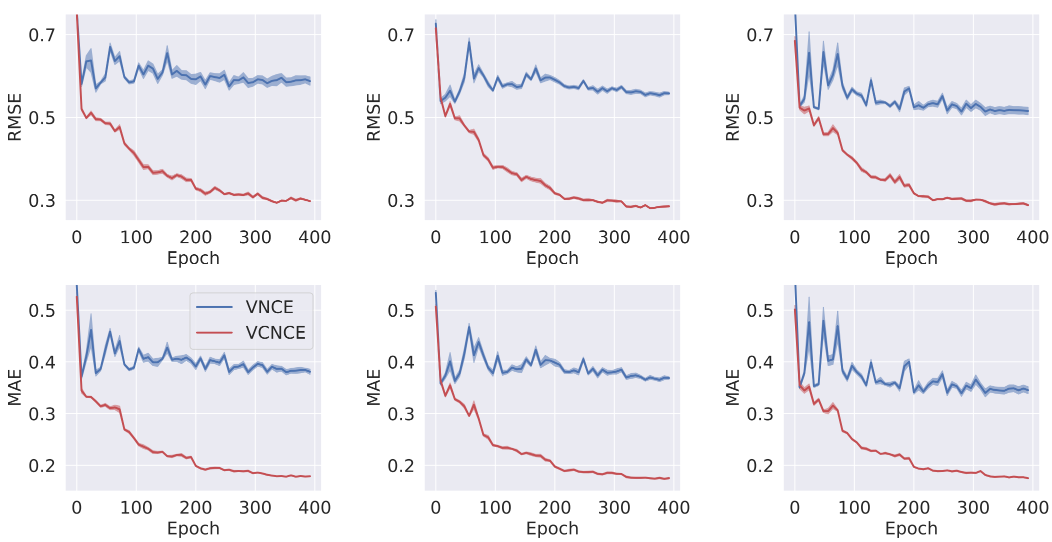
We conduct an additional ablation study to show the advantage of VCNCE over the variational noise-contrastive estimation [VNCE, 1] objective. The main difference between the VNCE and VCNCE is that VNCE uses noises from a fixed Gaussian distribution, *e.g.*, $y \sim p_n(y) = \mathcal{N}(y \mid \mu, \sigma^2)$, while VCNCE uses conditional noises, *e.g.*, $y \sim p_c(y \mid x) = \mathcal{N}(y \mid x, \sigma^2)$. Hence, these two strategies yield different objective functions. The objective function of VNCE is defined as

$$\mathcal{L}_{\text{VNCE}} = \mathbb{E}_x \mathbb{E}_{q(\boldsymbol{m}|x;\varphi)} \log \left(\frac{\phi(x, \boldsymbol{m}; \theta)}{\phi(x, \boldsymbol{m}; \theta) + \nu q(\boldsymbol{m} \mid x; \varphi) p_n(x)} \right) + \nu \mathbb{E}_y \log \left(\frac{\nu p_n(y)}{\nu p_n(y) + \mathbb{E}_{q(\boldsymbol{m}|y)} \left[\frac{\phi(y, \boldsymbol{m}; \theta)}{q(\boldsymbol{m}|y)} \right]} \right),$$

where $p_n(\cdot)$ is the fixed noise distribution. For VNCE, choosing inappropriate noise distributions may result in bad performances.

We test the proposed model on the *Air* dataset, training on the VCNCE loss and VNCE loss, respectively. We set the batch size to 128 and run 400 epochs. Adam optimizer with initial learning rate $1\text{e}{-2}$ is adopted. The initial learning rate is subsequently reduce by 0.3 at $20\%, 50\%$ and 80% of the total epochs. For VNCE, we set $\mu = 0$, which is a common practice in relevant literature. To show how the noise variance affects the learning process, we test different noise variances, *e.g.*, $\sigma \in \{0.3, 0.5, 0.7\}$ for both VNCE and VCNCE. Other settings are the same with Appendix C.2.2.

Fig. 1 depicts the RMSE and MAE on the test data when optimizing VNCE and VCNCE objective functions. We test five runs, plot mean values in lines and standard deviations in shadowed areas. It is shown that VCNCE gets better and more stable results on both RMSE and MAE.



(a) Noise $\sigma = 0.3$ (b) Noise $\sigma = 0.5$ (c) Noise $\sigma = 0.7$

Figure 1: Learning process of optimizing the VNCE and VCNCE loss. The first row is RMSE and the second row is MAE.

References

[1] Benjamin Rhodes and Michael U Gutmann. Variational noise-contrastive estimation. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2741–2750. PMLR, 2019.

[2] Zheng Wang and Shandian Zhe. Nonparametric factor trajectory learning for dynamic tensor decomposition. In *International Conference on Machine Learning*, pages 23459–23469. PMLR, 2022.