

## Supplementary Material:

To ensure a comprehensive understanding of our paper and to support reproducibility and reliability, we present additional results and provide complete proofs for the theorems articulated in the main paper. This supplementary material is meticulously organized as follows:

## Table of Contents

<b>A</b>	<b>Extended Related work and Proofs</b>	<b>17</b>
A.1	Extend the Discussion on Related Work . . . . .	18
A.2	Generalization, Compositionality and irreducibility assumptions . . . . .	18
A.3	Element-wise Identifiability given index support $i$ for Piecewise Linear . . . . .	19
A.4	The Generative Process and The ELBO for Multivariate Mixture Gaussian . . . . .	21
A.4.1	Variational Lower Bound for TimeCSL . . . . .	22
A.4.2	The Equivalence Between Matrix Normal and Multivariate Normal Distributions . . . . .	24
A.5	Structural Sparsity and Sufficient Partial Selective Pairing Assumptions . . . . .	25
<b>B</b>	<b>Experiments and Implementation Settings</b>	<b>26</b>
B.1	Implementation source. (TimeCSL-Lib) . . . . .	26
B.2	Datasets. . . . .	27
B.3	Contrastive Partial Selective Pairing - Data Augmentations . . . . .	27
B.4	Implementation of Metrics and study case . . . . .	27
B.4.1	Alignment prior to measuring Weak MCC . . . . .	28
B.4.2	Measuring Identifiability strong-MCC and weak-MCC . . . . .	28
B.4.3	Measuring disentanglement of the learned representation . . . . .	28
B.5	ResTimeCSL Architecture . . . . .	29
B.6	Pipeline Correlated samples. . . . .	30
B.7	Impact of ReLU/LeakyReLU and Attention layer with GELU activation on Decoder Behavior . . . . .	30
B.8	Validation of results on synthetic Data Generation . . . . .	31
B.9	Additional Experiment Results. . . . .	31
B.9.1	Experiment on REDD and REFIT datasets . . . . .	31
B.9.2	Experiment on Synthetic Datasets . . . . .	33
B.9.3	Comparisons Between TimeCSL and Baselines on KITTI Dataset . . . . .	33

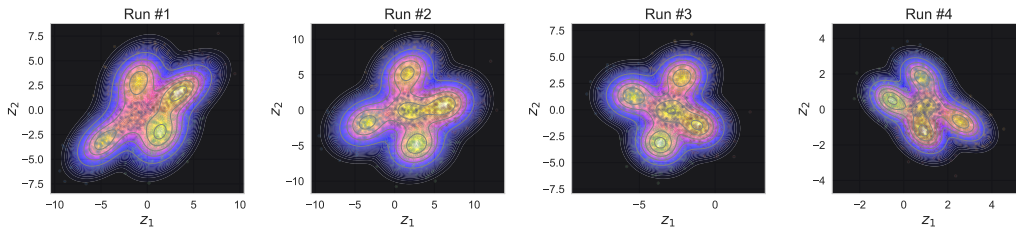


Figure 7: Recovered latent spaces for 4 runs of TimeCSL on REDD dataset with 5 latents ( $n = 5, d = 16$ ) {FR, DW, WM, HTR, LT}.

## A EXTENDED RELATED WORK AND PROOFS

In this section, we detail the contributions of the paper, including all the details. Although there is no change in their contents, the formulation of some definitions and theorems are slightly altered here to be more precise and cover edge cases omitted in the main text. Hence, the numbering of the restated elements is reminiscent of that used in the main text.

## A.1 EXTEND THE DISCUSSION ON RELATED WORK

Self-supervised learning (SSL) methods have moved away from using negative pairs, as in contrastive learning (CL), and instead focus on alignment with various forms of regularization to prevent collapsed representations. For example, BYOL (Grill et al., 2020) and SimSiam (Chen & He, 2021) use architectural regularization with moving-average updates for a separate *target* network (BYOL only) or a stop-gradient operation (for both). Meanwhile, BarlowTwins (Zbontar et al., 2021) promotes redundancy reduction and alignment by optimizing the cross-correlation between  $\mathbf{z}$  and  $\mathbf{z}'$  to match the identity matrix, ensuring zero off-diagonals and ones on the diagonal. We can interpret positive augmentation as a modified representation  $\mathbf{z}'$  that is connected to the original  $\mathbf{z}$  through a conditional distribution  $p(\mathbf{z}' | \mathbf{z})$ . This implies that the augmented observation  $\mathbf{x}'$  shares similar information with the anchor observation  $\mathbf{x}$ , and is generated by applying the same mixing function  $g_\theta$  as defined in data-generating process Eq. (2.2).

Table 3: Related work in nonlinear ICA for time series. A blue check denotes that a method has an attribute, whereas a red cross denotes the opposite. <sup>†</sup> indicates an approach we implemented.

Approach	Temporal Data	Dependent Factors	Nonparametric Expression	Stationary Process
TCL (Hyvarinen & Morioka, 2016)	✓	✗	✗	✗
PCL (Hyvarinen & Morioka, 2017)	✓	✗	✓	✓
GCL (Hyvarinen et al., 2019)	✓	✗	✓	✗
iVAE (Khemakhem et al., 2020b)	✗	✗	✗	✗
GIN (Sorenson et al., 2020)	✗	✗	✗	✗
HM-NLICA (Hälvä & Hyvärinen, 2020)	✓	✗	✓	✗
SlowVAE (Klindt et al., 2021)	✓	✗	✗	✓
(Yao et al., 2021) LEAP (Theorem 1)	✓	✓	✓	✗
(Yao et al., 2021) LEAP (Theorem 2)	✓	✓	✗	✓
TimeCSL (our) <sup>†</sup> TimeCSL (Theorem 1)	✓	✓	✓	✓ + ✗

## A.2 GENERALIZATION, COMPOSITIONALITY AND IRREDUCIBILITY ASSUMPTIONS

**Compositional contrast** In recent work on compositionality (Assouel et al., 2022; Zhao et al., 2022; Kurth-Nelson et al., 2022) and its importance in learning models that can generalize well to novel situations, the concept of *compositional contrast* has emerged as a powerful tool for evaluating how well a model separates information into independent, non-interacting components. This concept is particularly relevant in the context of time series analysis or image generation, where the model’s ability to decompose an input into distinct parts, or “slots,” can significantly impact the quality of predictions and interpretability. Compositionality ensures that each slot, or latent variable, corresponds to a specific factor or component of the data. In highly compositional models, these components do not interact with each other—each one affects a distinct aspect of the output. In contrast, non-compositional models tend to mix these components, making it harder to disentangle the factors and interpret the model’s output. Evaluating how well a model adheres to compositionality principles can be challenging, as it requires quantifying how independent the slots are in their contribution to the final output. To address this, Brady et al. (2023) introduced the notion of *compositional contrast*, which measures the extent to which the model’s latent variables (slots) interact when producing the final output. This measure is particularly useful in determining whether a decoder is truly compositional—that is, whether each slot contributes independently of the others, or if there are unwanted interactions between them. Before we introduce the formal definition of compositional contrast, it is important to understand the underlying principle. The intuition behind the compositional contrast is that if a model is fully compositional, each slot should affect only a specific subset of the output (e.g., one region of an image or one time series variable) and have no influence on other components. Conversely, if the model is not compositional, changes in one slot will influence multiple components of the output simultaneously, indicating that the slots are not independent. The compositional contrast function captures this idea by calculating how much the gradients of each slot (with respect to the model’s output) overlap. If the gradients of different slots with respect to the same output component are non-zero, this suggests interaction between the slots, indicating a lack of compositionality. The function sums these interactions across all slots and output components, providing a single value that quantifies the degree of interaction. A lower compositional contrast value suggests higher compositionality, while a higher value indicates more interaction between slots. Formally, the compositional contrast is defined as follows:

**Definition A.1** (Compositional Contrast). Let  $g_\theta : \mathcal{Z} \rightarrow \mathcal{X}$  be differentiable. The *compositional contrast* of  $g_\theta$  at  $\mathbf{z}$  is

$$C_{\text{comp}}(g_\theta, \mathbf{z}) = \sum_{n=1}^N \sum_{k=1}^K \sum_{j=k+1}^K \left\| \frac{\partial g_{\theta n}}{\partial \mathbf{z}_k}(\mathbf{z}) \right\| \left\| \frac{\partial g_{\theta n}}{\partial \mathbf{z}_j}(\mathbf{z}) \right\|. \quad (\text{A.1})$$

This contrast function was proven to be zero if and only if  $g_\theta$  is compositional according to Eq. (4.5). The function can be understood as computing each pairwise product of the (L2) norms for each pixel’s gradients with respect to any two distinct slots  $k \neq j$  and taking the sum. This quantity is non-negative and will only be zero if each pixel is affected by at most one slot, ensuring that  $g_\theta$  satisfies Eq. (4.5). We can use this function to measure the compositional of a decoder in our experiments (see § 4), where it serves as a key indicator of how effectively the model decomposes its inputs into independent components. More empirical and theoretical details on the function can be found in Brady et al. (2023).

### A.3 ELEMENT-WISE IDENTIFIABILITY GIVEN INDEX SUPPORT $\mathcal{I}$ FOR PIECEWISE LINEAR

In this section, we present the proof of Thm. 4.2. To establish a solid foundation for the argument, we first restate Asm 4.1, which plays a pivotal role in the proof.

**Assumption 4.1 (Sufficient Partial Selective Pairing).** For each factor  $k \in [n]$ , there exist observations  $(\mathbf{x}, \mathbf{x}') \in \mathcal{X}$  such that the union of the shared support indices  $\mathbf{i} = \mathbf{I}(\mathbf{x}, \mathbf{x}')$  that do not include  $k$  must cover all other factors. Formally:

$$\bigcup_{\mathbf{i} \in \mathcal{I} | k \notin \mathbf{i}} \mathbf{i} = [n] \setminus \{k\} \quad , \quad \mathcal{I} := \{\mathbf{i} \subseteq [n] \mid p(\mathbf{i}) > 0\} \quad (4.1)$$

where  $\mathcal{I}$  is the set of shared support indices and  $p(\mathbf{i}) := \frac{1}{\#\mathcal{X}} \cdot \#\{\mathcal{S}(\mathbf{x}) = \mathbf{i}, \mathbf{x} \in \mathcal{X}\}$  gives the probability that the factors indexed by  $\mathbf{i}$  are active, with  $k \notin \mathbf{i}$  inactive.

Additionally, we introduce some notation. For  $\mathbf{i} \in \mathcal{I}$ , we assume that the probability measure  $\mathbb{P}_{\mathbf{z}_\mathbf{i}}$  admits a density with respect to the Lebesgue measure on  $\mathbb{R}^{|\mathbf{i}|}$ . We let  $\equiv$  denote equality in the distribution.

**Theorem 4.2** (Element-wise Identifiability given index support  $\mathbf{i}$  for Piecewise Linear  $g_\theta$ ). *Let  $\mathbf{f}_\phi : \mathbb{R}^{d \times n} \rightarrow \mathbb{R}^{T \times n}$  be a continuous invertible piecewise linear function and  $\hat{g}_\theta : \mathbb{R}^{d \times n} \rightarrow \mathbb{R}^{T \times n}$  be a continuous invertible piecewise linear function onto its image. Assume that Asm 4.1, Asm 2.1 holds, and the mixed observations  $(\mathbf{x}, \mathbf{x}') \stackrel{\text{i.i.d.}}{\sim} \mathcal{X}$ , follows the data-generating process Eq. (2.2). The learnable latent  $\hat{\mathbf{z}}$  (resp.  $\hat{\mathbf{z}}'$ ) of  $\mathbf{z}$  (resp.  $\mathbf{z}'$ ). If all following conditions hold:*

$$\mathbb{E}\|\hat{\mathbf{z}}\|_0 \leq \mathbb{E}\|\mathbf{z}\|_0 \quad \text{and} \quad \mathbb{E}\|\hat{\mathbf{z}}'\|_0 \leq \mathbb{E}\|\mathbf{z}'\|_0, \quad \text{and}, \quad (4.2)$$

$$\mathcal{R}_{\text{align}}(\hat{\mathbf{z}}, \hat{\mathbf{z}}', \mathbf{i}) := \sum_{i \in \mathbf{i}} \left| \frac{\hat{\mathbf{z}}_i'^\top \hat{\mathbf{z}}_i}{\|\hat{\mathbf{z}}_i'\|_2 \|\hat{\mathbf{z}}_i\|_2} - 1 \right| = 0. \quad (4.3)$$

then  $\mathbf{z}$  is identified by  $\mathbf{h} := \hat{g}_\theta^{-1}(\mathbf{x})$ , i.e.,  $\hat{g}_\theta^{-1} \circ g_\theta$  is a permutation composed with element-wise invertible linear transformations (Def. 2.2).

*Proof.* The proving strategy has three steps: Intuitively, based result (Kivva et al., 2022) combined with contrastivity between tow latent based their shared support indices  $\mathbf{i}$ . This means that for the data that satisfy Asm 4.1,  $g_\theta(\mathbf{z})$  and  $\hat{g}_\theta(\hat{\mathbf{z}})$  are equally distributed, then there exists an invertible affine transformation such that  $\mathbf{h}(\mathbf{z}) = \mathbf{z}'$ . Second, we use the strategy of linear identifiability (Lachapelle & Lacoste-Julien, 2022) to obtain element wise identifiability:

**Step 1) Contrastive Sparsity and Linear Identifiability given pairs  $\mathbf{i}$**  We begin by recalling the result from Kivva et al. (2022) on the existing of an invertible function affine transformation  $\mathbf{h}_k$ , we adapt this for the case where if the reconstruction objective is minized and alignment. The theorem on identifiability of MVNs states:

**Theorem A.2.** Let  $g_\theta, g'_\theta : \mathbb{R}^{d \times n} \rightarrow \mathbb{R}^{C \times T}$  be piecewise affine functions satisfying 2.1. Let  $\mathbf{z} \sim \sum_{i=1}^J \omega_i \mathcal{N}(\mu_i, \Sigma_i)$  and  $\mathbf{z}' \sim \sum_{j=1}^{J'} \omega'_j \mathcal{N}(\mu'_j, \Sigma'_j)$  be a pair of GMMs (in reduced form). Suppose that  $g_\theta(\mathbf{z})$  and  $g'_\theta(\mathbf{z}')$  are equally distributed. Then there exists an invertible affine transformation  $\mathbf{h} : \mathbb{R}^{d \times n} \rightarrow \mathbb{R}^{d \times n}$  such that  $\mathbf{h}(\mathbf{z}) \equiv \mathbf{z}'$ , i.e.,  $J = J'$  and for some permutation  $\pi$  we have  $\omega_i = \omega'_{\pi(i)}$  and  $\mathbf{h}_\# \mathcal{N}(\mu_i, \Sigma_i) = \mathcal{N}(\mu'_{\pi(i)}, \Sigma'_{\pi(i)})$ .

We recall that the transformation and the number of components can be unknown and arbitrary, and that no assumption of separation or independence is necessary for the distribution.

By Theorem C.2 (Kivva et al., 2022), since contrastive learning involves the minimisation of a contrastive loss which ensures that similar data points (positive pairs) are moved closer together and dissimilar data points (negative pairs) are moved further apart. Let the inferred latent representation  $(\mathbf{z}, \mathbf{z}')$  be handled by the exact same function  $\mathbf{f}_\phi$ , and we consider the zero reconstruction under  $\mathcal{R}_{align} = 0$  for all slot indices in  $\mathbf{i}$ . Alongside this, contrastive loss minimization induces the distributions of  $g_\theta(\mathbf{z})$  and  $g_\theta(\mathbf{z}')$  to become indistinguishable on  $i \in \mathbf{i}$  to be well-aligned, apart from for  $k \notin \mathbf{i}$ , but as we consider the Asm 4.1 on the sufficient partial pairing that will cover this factor  $k$  in another pairing sample of the pair  $(\mathbf{x}, \mathbf{x}')$ . Thus, according to Theorem C.2 (Kivva et al., 2022), there must exist an invertible affine transformation  $\mathbf{h}$  such that  $\mathbf{h}(\mathbf{z}) \equiv \mathbf{z}'$ . It is more likely to observe that :

$$\sum_{j=1}^J \omega_k g_\theta \# \mathcal{N}(\mu_k, \sigma_k) \sim g_\theta \# \mathbf{f}_\phi \left( \sum_{j=1}^J \omega_k \mathcal{N}(\mu_k, \sigma_k) \right). \quad (\text{A.2})$$

In other words, minimizing to hold (i) and zeros error construction, implies a mixture model whose components are piecewise affine transformations identifiable.

## Step 2) Sparsity Pattern of an Invertible Matrix with an element-wise linear transformation

Since  $\mathbf{x} = g_\theta(\mathbf{z})$ , we can rewrite perfect reconstruction as:

$$\mathbb{E} \|g_\theta(\mathbf{z}) - \hat{g}_\theta(\mathbf{f}_\phi(g_\theta(\mathbf{z})))\|_2^2 = 0 \quad (10)$$

This means  $g_\theta$  and  $\hat{g}_\theta \circ \mathbf{f}_\phi \circ g_\theta$  are equal  $\mathbb{P}_\mathbf{z}$ -almost everywhere. Both of these functions are continuous,  $g_\theta$  by Asm 2.1, and  $\hat{g}_\theta \circ \mathbf{f}_\phi \circ g_\theta$  because  $\hat{g}_\theta$  is continuous, and  $g_\theta, \mathbf{f}_\phi$  are linear. Since they are continuous and equal  $\mathbb{P}_\mathbf{z}$ -almost everywhere  $\mathcal{Z}$ , this means that they must be equal over the support of  $\mathcal{Z}$ , i.e.,

$$g_\theta(\mathbf{z}) = \hat{g}_\theta \circ \mathbf{f}_\phi \circ g_\theta(\mathbf{z}), \quad \forall \mathbf{z} \in \mathcal{Z}. \quad (11)$$

This can be easily shown by contradiction considering any slot latent  $\mathbf{z}' \in \mathcal{Z}$  on which  $g_\theta$  and  $\hat{g}_\theta \circ \mathbf{f}_\phi \circ g_\theta$  are different, i.e.,  $\hat{g}_\theta \circ \mathbf{f}_\phi \circ g_\theta(\mathbf{z}') \neq g_\theta(\mathbf{z}')$ . This would imply that  $(g_\theta - \hat{g}_\theta \circ \mathbf{f}_\phi \circ g_\theta)$ , which is also a continuous function, is non-zero at  $\mathbf{z}'$  and in its neighborhood, which contradict the assumption that  $g_\theta$  and  $\hat{g}_\theta \circ \mathbf{f}_\phi \circ g_\theta$  are the same  $\mathbb{P}_\mathbf{z}$ -almost everywhere. We can now apply the inverse of  $\hat{g}_\theta$  on both sides to obtain

$$\hat{g}_\theta^{-1} \circ g_\theta(\mathbf{z}) = \mathbf{f}_\phi \circ g_\theta(\mathbf{z}) = \mathbf{h}(\mathbf{z}), \quad \forall \mathbf{z} \in \mathcal{Z}. \quad (12)$$

Since both  $g_\theta$  and  $\mathbf{f}_\phi$  are invertible linear functions, given the first part of the proof (Step 1-App. A.3)  $\mathbf{h}$  is also an invertible linear function. We now show that  $\mathbf{h}$  is a permutation composed with an element-wise linear transformation. To do this, we leverage the sparsity constraint:

$$\mathbb{E} \|\hat{\mathbf{z}}\|_0 \leq \mathbb{E} \|\mathbf{z}\|_0 \quad (\text{A.3})$$

$$\mathbb{E} \|\mathbf{f}_\phi(g_\theta(\mathbf{z}))\|_0 \leq \mathbb{E} \|\mathbf{z}\|_0 \quad (\text{A.4})$$

$$\mathbb{E} \|\mathbf{h}(\mathbf{z})\|_0 \leq \mathbb{E} \|\mathbf{z}\|_0 \quad (\text{A.5})$$

$$(\text{A.6})$$

Since  $\mathbf{h}_k$  is invertible linear transformation, we have that  $\mathbf{h}_k(\mathbf{z}) = \mathbf{w}_k \cdot \mathbf{z}$  and its determinant is non-zero, i.e.,

$$\det(\mathbf{h}) := \sum_{\pi \in \mathcal{P}} \text{sign}(\pi) \prod_{k=1}^n \mathbf{h}_{k, \pi(k)} \neq 0, \quad (\text{A.7})$$

where  $\mathcal{P}$  denotes the set of all  $n$ -permutations. This expression implies that at least one term in the sum is non-zero, meaning there exists a permutation  $\pi \in \mathcal{P}$  such that for every  $k \in [n]$ ,  $\frac{\partial \mathbf{h}_k}{\partial \mathbf{z}_{\pi(k)}} \neq 0$ .

Following the steps outlined in Theorem B.4 by (Lachapelle et al., 2022), and under the assumption of Asm 4.1, we extend the disentanglement analysis to our setting. This leads to the conclusion that  $\mathbf{h}$  can be expressed as a permutation composed with an element-wise invertible linear transformation, based on the shared support indices  $\mathbf{i}$  of the latent slot within the subspace  $\mathcal{Z}_{\mathbf{i}}$ . Specifically, there exists a permutation  $\pi$  on  $[n]$  such that, for each latent slot  $k$ , the corresponding permutation is given by  $\pi(k)$ . Since  $\mathcal{I}$  is a finite set, which allows us to order its elements as  $\{\mathbf{i}_1, \dots, \mathbf{i}_{|\mathcal{I}|}\}$ . Therefore, we can express  $\mathcal{Z}$  as the union  $\mathcal{Z} = \bigcup_{\mathbf{i} \in \mathcal{I}} \mathcal{Z}^{(\mathbf{i})}$ . While we have already shown that  $\mathbf{h}$  is affine on each  $\mathcal{Z}_{\mathbf{i}}$ , we now demonstrate that  $\mathbf{h}$  is linear on  $\mathcal{Z}$ , i.e.,  $\mathbf{h}(\mathbf{z})$  is a linear function on the entire set  $\mathcal{Z} = \bigcup_{\mathbf{i} \in \mathcal{I}} \mathcal{Z}_{\mathbf{i}}$ . This completes the proof.  $\square$

#### A.4 THE GENERATIVE PROCESS AND THE ELBO FOR MULTIVARIATES MIXTURE GAUSSIAN

We in this subsection how TimeCSL is trained based on a VAE process does similar to (Kivva et al., 2022; Jang et al., 2017), which more kind of unsupervised generative approach for clustering that performance well, we herein first describe the generative process of TimeCSL. Specifically, suppose there are  $n$  slots latents each has a dimension  $d$ , an observed sample  $\mathbf{x} \sim \mathcal{X}$  is generated by the following process:

---

##### Algorithm 1 Generative Process

---

- 1: **Input:** Prior probabilities  $\mathbf{w}$ , neural network parameters  $\theta$
  - 2: **for**  $j = 1, 2, \dots, N$  **do**
  - 3:   Sample slot  $k \sim \text{Cat}(\mathbf{w})$
  - 4:   Sample latent vector  $\mathbf{z}^{(j)} \sim \mathcal{N}(\boldsymbol{\mu}_k^{(j)}, \boldsymbol{\sigma}_k^{(j)} \cdot \boldsymbol{\sigma}_k^{(j)} \mathbf{I})$
  - 5:   Compute  $[\boldsymbol{\mu}_\phi(\mathbf{x}^{(j)}); \log \boldsymbol{\sigma}_\phi(\mathbf{x}^{(j)})^2] = \mathbf{g}_\theta(\mathbf{z}^{(j)})$
  - 6:   Sample observation  $\mathbf{x}_j \sim \mathcal{N}(\boldsymbol{\mu}_\theta(\mathbf{x}^{(j)}), \boldsymbol{\sigma}_\theta(\mathbf{x}^{(j)})^2 \mathbf{I})$  or  $\text{Ber}(\boldsymbol{\mu}_\theta(\mathbf{x}^{(j)}))$
  - 7: **end for**
  - 8: **return**  $\{\mathbf{x}^{(j)}, \mathbf{z}^{(j)}, k\}_{j=1}^N$
- 

**Lemma A.3.** Given two multivariate Gaussian distributions  $q(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\sigma}}^2 \mathbf{I})$  and  $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2 \mathbf{I})$ , we have:

$$\int q(\mathbf{z}) \log p(\mathbf{z}) d\mathbf{z} = \sum_{j=1}^J -\frac{1}{2} \log(2\pi\sigma_j^2) - \frac{\hat{\sigma}_j^2}{2\sigma_j^2} - \frac{(\hat{\mu}_j - \mu_j)^2}{2\sigma_j^2}, \quad (\text{A.8})$$

where  $\mu_j$ ,  $\sigma_j$ ,  $\hat{\mu}_j$  and  $\hat{\sigma}_j$  simply denote the  $j^{\text{th}}$  element of  $\boldsymbol{\mu}$ ,  $\boldsymbol{\sigma}$ ,  $\hat{\boldsymbol{\mu}}$  and  $\hat{\boldsymbol{\sigma}}$ , respectively, and  $J = d \times n$  is the dimensionality of  $\mathbf{z}$ .

*Proof.*

$$\begin{aligned}
& \int q(\mathbf{z}) \log p(\mathbf{z}) d\mathbf{z} = \int \mathcal{N}(\mathbf{z}; \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\sigma}}^2 \mathbf{I}) \log \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2 \mathbf{I}) d\mathbf{z} \\
&= \int \prod_{j=1}^J \frac{1}{\sqrt{2\pi\hat{\sigma}_j^2}} \exp\left(-\frac{(z_j - \hat{\mu}_j)^2}{2\hat{\sigma}_j^2}\right) \log \left[ \prod_{j=1}^J \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left(-\frac{(z_j - \mu_j)^2}{2\sigma_j^2}\right) \right] d\mathbf{z} \\
&= \sum_{j=1}^J \int \frac{1}{\sqrt{2\pi\hat{\sigma}_j^2}} \exp\left(-\frac{(z_j - \hat{\mu}_j)^2}{2\hat{\sigma}_j^2}\right) \log \left[ \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left(-\frac{(z_j - \mu_j)^2}{2\sigma_j^2}\right) \right] dz_j \\
&= \sum_{j=1}^J \int \frac{1}{\sqrt{2\pi\hat{\sigma}_j^2}} \exp\left(-\frac{(z_j - \hat{\mu}_j)^2}{2\hat{\sigma}_j^2}\right) \left[ -\frac{1}{2} \log(2\pi\sigma_j^2) \right] dz_j - \int \frac{1}{\sqrt{2\pi\hat{\sigma}_j^2}} \exp\left(-\frac{(z_j - \hat{\mu}_j)^2}{2\hat{\sigma}_j^2}\right) \frac{(z_j - \mu_j)^2}{2\sigma_j^2} dz_j \\
&= \sum_{j=1}^J -\frac{1}{2} \log(2\pi\sigma_j^2) - \int \frac{1}{\sqrt{2\pi\hat{\sigma}_j^2}} \exp\left(-\frac{(z_j - \hat{\mu}_j)^2}{2\hat{\sigma}_j^2}\right) \frac{(z_j - \hat{\mu}_j)^2 + 2(z_j - \hat{\mu}_j)(\hat{\mu}_j - \mu_j) + (\hat{\mu}_j - \mu_j)^2}{2\hat{\sigma}_j^2} \frac{\hat{\sigma}_j^2}{\sigma_j^2} dz_j \\
&= \mathbf{b} - \frac{\hat{\sigma}_j^2}{\sigma_j^2} \int \frac{1}{\sqrt{2\pi\hat{\sigma}_j^2}} \exp\left(-\frac{(z_j - \hat{\mu}_j)^2}{2\hat{\sigma}_j^2}\right) \frac{(z_j - \hat{\mu}_j)^2}{2\hat{\sigma}_j^2} dz_j - \int \frac{1}{\sqrt{2\pi\hat{\sigma}_j^2}} \exp\left(-\frac{(z_j - \hat{\mu}_j)^2}{2\hat{\sigma}_j^2}\right) \frac{(\hat{\mu}_j - \mu_j)^2}{2\sigma_j^2} dz_j \\
&= \mathbf{b} - \frac{\hat{\sigma}_j^2}{\sigma_j^2} \int \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x_j^2}{2}\right) \frac{x_j^2}{2} dx_j - \frac{(\hat{\mu}_j - \mu_j)^2}{2\sigma_j^2} \\
&= \mathbf{b} - \frac{\hat{\sigma}_j^2}{\sigma_j^2} \int \frac{1}{\sqrt{2\pi}} \left(-\frac{x_j}{2}\right) d\left(\exp\left(-\frac{x_j^2}{2}\right)\right) - \frac{(\hat{\mu}_j - \mu_j)^2}{2\sigma_j^2} \\
&= \mathbf{b} - \frac{\hat{\sigma}_j^2}{\sigma_j^2} \left[ \frac{1}{\sqrt{2\pi}} \left(-\frac{x_j}{2}\right) \exp\left(-\frac{x_j^2}{2}\right) \Big|_{-\infty}^{\infty} - \int \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x_j^2}{2}\right) d\left(-\frac{x_j}{2}\right) \right] - \frac{(\hat{\mu}_j - \mu_j)^2}{2\sigma_j^2} \\
&= \sum_{j=1}^J -\frac{1}{2} \log(2\pi\sigma_j^2) - \frac{\hat{\sigma}_j^2}{2\sigma_j^2} - \frac{(\hat{\mu}_j - \mu_j)^2}{2\sigma_j^2}
\end{aligned}$$

where  $\mathbf{b}$  denotes  $\sum_{j=1}^J -\frac{1}{2} \log(2\pi\sigma_j^2)$  for simplicity.

□

#### A.4.1 VARIATIONAL LOWER BOUND FOR TIMECSL

A TimeCSL instance is tuned to maximize the likelihood of the given data points. Given the generative process in Section A.4, by using Jensen's inequality, the log-likelihood of TimeCSL can be written as:

$$\begin{aligned}
\log p(\mathbf{x}) &= \log \int_{\mathbf{z}} \sum_k p(\mathbf{x}, \mathbf{z}, k) d\mathbf{z} \\
&\geq E_{q(\mathbf{z}, k|\mathbf{x})} [\log \frac{p(\mathbf{x}, \mathbf{z}, k)}{q(\mathbf{z}, k|\mathbf{x})}] = \mathcal{L}_{\text{ELBO}}(\mathbf{x})
\end{aligned} \tag{A.9}$$

where  $\mathcal{L}_{\text{ELBO}}$  is the evidence lower bound (ELBO),  $q(\mathbf{z}, k|\mathbf{x})$  is the variational posterior to approximate the true posterior  $p(\mathbf{z}, k|\mathbf{x})$ . In TimeCSL, we assume  $q(\mathbf{z}, k|\mathbf{x})$  to be a mean-field distribution and can be factorized as:

$$q(\mathbf{z}, k|\mathbf{x}) = q(\mathbf{z}|\mathbf{x})q(k|\mathbf{x}). \tag{A.10}$$



Then, according to Equation A.10, the  $\mathcal{L}_{\text{ELBO}}(\mathbf{x})$  in Equation A.9 can be rewritten as:

$$\begin{aligned}\mathcal{L}_{\text{ELBO}}(\mathbf{x}) &= E_{q(\mathbf{z}, k|\mathbf{x})} \left[ \log \frac{p(\mathbf{x}, \mathbf{z}, k)}{q(\mathbf{z}, k|\mathbf{x})} \right] \\ &= E_{q(\mathbf{z}, k|\mathbf{x})} [\log p(\mathbf{x}, \mathbf{z}, k) - \log q(\mathbf{z}, k|\mathbf{x})] \\ &= E_{q(\mathbf{z}, k|\mathbf{x})} [\log p(\mathbf{x}|\mathbf{z}) + \log p(\mathbf{z}|k) \\ &\quad + \log p(k) - \log q(\mathbf{z}|\mathbf{x}) - \log q(k|\mathbf{x})]\end{aligned}\tag{A.11}$$

In TimeCSL, similar to VAE, we use a neural network  $g$  to model  $q(\mathbf{z}|\mathbf{x})$ :

$$[\hat{\boldsymbol{\mu}}; \log \hat{\boldsymbol{\sigma}}^2] = \mathbf{f}_{\phi}(\mathbf{x}; \phi) \tag{A.12}$$

$$q(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\sigma}}^2 \mathbf{I}) \tag{A.13}$$

where  $\phi$  is the parameter of network  $g$ .

By substituting the terms in Equation A.11 and using the SGVB estimator and the *reparameterization* trick, the  $\mathcal{L}_{\text{ELBO}}(\mathbf{x})$  can be rewritten as:<sup>5</sup>

$$\begin{aligned}\mathcal{L}_{\text{ELBO}}(\mathbf{x}) &= \frac{1}{N} \sum_{l=1}^N \sum_{i=1}^{C \times T} \left[ x_i \log \boldsymbol{\mu}_{x_i}^{(l)} + (1 - x_i) \log \mathbf{f}_{\phi}(1 - \boldsymbol{\mu}_{x_i}^{(l)}) \right] \\ &\quad - \frac{1}{2} \sum_{k=1}^n \gamma_k \sum_{j=1}^J \left( \log \sigma_k^2|_j + \frac{\hat{\sigma}^2|_j}{\sigma_k^2|_j} + \frac{(\hat{\boldsymbol{\mu}}|_j - \boldsymbol{\mu}_k|_j)^2}{\sigma_k^2|_j} \right) \\ &\quad + \sum_{k=1}^n \gamma_k \log \frac{w_k}{\gamma_k} + \frac{1}{2} \sum_{j=1}^J (1 + \log \hat{\sigma}^2|_j)\end{aligned}\tag{A.14}$$

where  $N$  is the number of Monte Carlo samples in the SGVB estimator,  $C \times T$  is the dimensionality of  $\mathbf{x}$ ,  $n$  is number of slots or factors, and  $\boldsymbol{\mu}_x^{(l)}$ ,  $x_i$  is the  $i^{\text{th}}$  element of  $\mathbf{x}$ ,  $J$  is the dimensionality of  $\boldsymbol{\mu}_k$ ,  $\sigma_k^2$ ,  $\hat{\boldsymbol{\mu}}$  and  $\hat{\boldsymbol{\sigma}}^2$ , and  $*|_j$  denotes the  $j^{\text{th}}$  element of  $*$ ,  $n$  is the number of slots,  $w_k$  is the prior probability of slot  $k$ , and  $\gamma_k$  denotes  $q(k|\mathbf{x})$  for simplicity. In Equation A.14, we compute  $\boldsymbol{\mu}_x^{(l)}$  as

$$\boldsymbol{\mu}_x^{(l)} = \mathbf{f}_{\phi}(\mathbf{z}^{(l)}; \theta), \tag{A.15}$$

where  $\mathbf{z}^{(l)}$  is the  $l^{\text{th}}$  sample from  $q(\mathbf{z}|\mathbf{x})$  by Equation A.13 to produce the Monte Carlo samples. According to the *reparameterization* trick,  $\mathbf{z}^{(l)}$  is obtained by

$$\mathbf{z}^{(l)} = \hat{\boldsymbol{\mu}} + \hat{\boldsymbol{\sigma}} \circ \boldsymbol{\epsilon}^{(l)}, \tag{A.16}$$

where  $\boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(0, \mathbf{I})$ ,  $\circ$  is element-wise multiplication, and  $\hat{\boldsymbol{\mu}}$ ,  $\hat{\boldsymbol{\sigma}}$  are derived by Equation A.12. We now describe how to formulate  $\gamma_c \triangleq q(k|\mathbf{x})$  in Equation A.14 to maximize the ELBO. Specifically,  $\mathcal{L}_{\text{ELBO}}(\mathbf{x})$  can be rewritten as:

$$\begin{aligned}\mathcal{L}_{\text{ELBO}}(\mathbf{x}) &= E_{q(\mathbf{z}, c|\mathbf{x})} \left[ \log \frac{p(\mathbf{x}, \mathbf{z}, c)}{q(\mathbf{z}, c|\mathbf{x})} \right] \\ &= \int_{\mathbf{z}} \sum_c q(k|\mathbf{x}) q(\mathbf{z}|\mathbf{x}) \left[ \log \frac{p(\mathbf{x}|\mathbf{z}) p(\mathbf{z})}{q(\mathbf{z}|\mathbf{x})} + \log \frac{p(k|\mathbf{z})}{q(k|\mathbf{x})} \right] d\mathbf{z} \\ &= \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{x}|\mathbf{z}) p(\mathbf{z})}{q(\mathbf{z}|\mathbf{x})} d\mathbf{z} - \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) D_{KL}(q(k|\mathbf{x}) || p(k|\mathbf{z})) d\mathbf{z}\end{aligned}\tag{A.17}$$

Once the training is done by maximizing the ELBO w.r.t the parameters of  $\{\boldsymbol{\pi}, \boldsymbol{\mu}_k, \boldsymbol{\sigma}_k, \boldsymbol{\theta}, \boldsymbol{\phi}\}$ ,  $k \in \{1, \dots, K\}$ , a latent representation  $\mathbf{z}$  can be extracted for each observed sample  $\mathbf{x}$ . This is done by Equation A.12 and Equation A.13.

<sup>5</sup>This is the case when the observation  $\mathbf{x}$  is binary. For the real-valued situation, the ELBO can be obtained in a similar way.

#### A.4.2 THE EQUIVALENCE BETWEEN MATRIX NORMAL AND MULTIVARIATE NORMAL DISTRIBUTIONS

In our formulation, we use a vectorization of the matrix  $\mathbf{z} \in \mathbb{R}^{d \times n}$ , which follows a multivariate Gaussian model. We now show that this can also be interpreted as a Matrix Normal distribution. The equivalence between the Matrix Normal and the Multivariate Normal density functions can be established using properties of the trace and the Kronecker product.

*Proof.* Let  $\mathbf{z}$  be modeled as a mixture of  $J$  Matrix Normal distributions. Each component of this mixture is characterized by a mean matrix  $\boldsymbol{\mu}_j \in \mathbb{R}^{d \times n}$  and a covariance matrix  $\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}_n \otimes \boldsymbol{\Sigma}_n \in \mathbb{R}^{d \times d} \otimes \mathbb{R}^{n \times n}$ , where  $\boldsymbol{\Sigma}_n$  and  $\boldsymbol{\Sigma}_n$  are the row and column covariance matrices, respectively. The probability density function of  $\mathbf{z}$  is thus given by

$$f_{\mathbf{z}}(\mathbf{z}) = \sum_{j=1}^J \omega_j \mathcal{N}(\mathbf{z} \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j),$$

where  $\omega_j$  are the mixing weights such that  $\omega_j > 0$  and  $\sum_{j=1}^J \omega_j = 1$ .

The Matrix Normal distribution is defined as

$$\mathcal{N}(\mathbf{z} \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \frac{1}{(2\pi)^{\frac{dn}{2}} |\boldsymbol{\Sigma}_j|^{\frac{n+d}{2}}} \exp \left( -\frac{1}{2} \text{tr} [\boldsymbol{\Sigma}_d^{-1} (\mathbf{z} - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_n^{-1} (\mathbf{z} - \boldsymbol{\mu}_j)] \right),$$

where  $\mathbf{z}$  is a  $d \times n$  matrix, and the covariance matrix  $\boldsymbol{\Sigma}_j$  is the Kronecker product  $\boldsymbol{\Sigma}_n \otimes \boldsymbol{\Sigma}_n$ , with  $\boldsymbol{\Sigma}_n$  and  $\boldsymbol{\Sigma}_n$  being the covariance matrices of the rows and columns of  $\mathbf{z}$ , respectively.

To connect the Matrix Mixture Normal distribution with the Mixture of Multivariate Normal distributions, we vectorize the matrix  $\mathbf{z}$ . The vectorization of a matrix  $\mathbf{z} \in \mathbb{R}^{d \times n}$  is given by

$$\text{vec}(\mathbf{z}) = [z_{11} \ z_{21} \ \cdots \ z_{d1} \ z_{12} \ \cdots \ z_{dn}]^T \in \mathbb{R}^{1 \times (d \cdot n)}$$

where  $\mathbf{z}_i$  denotes the  $i$ -th column of  $\mathbf{z}$ , and the resulting vector  $\text{vec}(\mathbf{z})$  is a  $d \cdot n$ -dimensional vector.

Now, substituting the vectorized form of  $\mathbf{z}$  into the Matrix Normal distribution, we have

$$\mathcal{N}(\text{vec}(\mathbf{z}) \mid \text{vec}(\boldsymbol{\mu}_j), \boldsymbol{\Sigma}_j) = \frac{1}{(2\pi)^{\frac{dn}{2}} |\boldsymbol{\Sigma}_j|^{\frac{d+n}{2}}} \exp \left( -\frac{1}{2} \bar{\mathbf{z}}^T \boldsymbol{\Sigma}_j^{-1} \bar{\mathbf{z}} \right), \quad (\text{A.18})$$

where  $\bar{\mathbf{z}} = \text{vec}(\mathbf{z}) - \text{vec}(\boldsymbol{\mu}_j)$ . Next, observe that the mixture model for  $\mathbf{z}$  in the original form becomes

$$f_{\mathbf{z}}(\mathbf{z}) = \sum_{j=1}^J \omega_j \mathcal{N}(\text{vec}(\mathbf{z}) \mid \text{vec}(\boldsymbol{\mu}_j), \boldsymbol{\Sigma}_n \otimes \boldsymbol{\Sigma}_n), \quad (\text{A.19})$$

which is a mixture of multivariate normal distributions in the vectorized space  $\mathbb{R}^{d \cdot n}$ . This shows that the Matrix Mixture Normal distribution is equivalent to a Mixture of Multivariate Normal distributions upon vectorization. To complete the proof, we use the determinant property of the Kronecker product:

$$|\boldsymbol{\Sigma}_n \otimes \boldsymbol{\Sigma}_n| = |\boldsymbol{\Sigma}_n|^n |\boldsymbol{\Sigma}_n|^d. \quad (\text{A.20})$$

Thus, the determinant of the covariance matrix  $\boldsymbol{\Sigma}_n \otimes \boldsymbol{\Sigma}_n$  can be written as the product of the determinants of  $\boldsymbol{\Sigma}_n$  and  $\boldsymbol{\Sigma}_n$ , raised to the appropriate powers. This confirms that the matrix mixture normal distribution is indeed equivalent to the mixture of multivariate normal distributions.  $\square$



## A.5 STRUCTURAL SPARSITY AND SUFFICIENT PARTIAL SELECTIVE PAIRING ASSUMPTIONS

**Comparison of Structural Sparsity and Sufficient Partial Selective Pairing Assumptions** We compare two important assumptions in the context of source separation: the Structural Sparsity assumption from (Ng et al., 2023) and the Sufficient Partial Selective Pairing assumption. The Structural Sparsity assumption for sources  $\mathbf{y} = \{\mathbf{y}_1, \dots, \mathbf{y}_n\}$  in the mixing matrix  $\mathbf{A}$  stipulates that for any pair of sources  $k$  and  $\ell$ , their supports (denoted  $\text{supp}(\mathbf{y}_k)$  and  $\text{supp}(\mathbf{y}_\ell)$ ) must differ in at least two observed variables, i.e.,

$$|\text{supp}(\mathbf{y}_k) \cup \text{supp}(\mathbf{y}_\ell)| - |\text{supp}(\mathbf{y}_k) \cap \text{supp}(\mathbf{y}_\ell)| > 1$$

Here,  $\text{supp}(\mathbf{y}_k)$  represents the indices of the observed variables affected by the source  $\mathbf{y}_k$ . This assumption ensures that the sources  $\mathbf{y}_k$  and  $\mathbf{y}_\ell$  are distinguishable in terms of the observed variables they influence.

**Example of Structural Sparsity Assumption** Consider a scenario where we have three sources  $\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3$  and four observed variables  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$ . The observed data  $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4]$  is a mixture of the sources. The supports for the sources are defined as follows:

$$\text{supp}(\mathbf{y}_1) = \{1\}, \quad \text{supp}(\mathbf{y}_2) = \{2\}, \quad \text{supp}(\mathbf{y}_3) = \{3\}$$

For the Structural Sparsity assumption to hold between sources  $\mathbf{y}_1$  and  $\mathbf{y}_2$ , the supports must differ in at least two observed variables. For example, we have:

$$|\text{supp}(\mathbf{y}_1) \cup \text{supp}(\mathbf{y}_2)| - |\text{supp}(\mathbf{y}_1) \cap \text{supp}(\mathbf{y}_2)| = 2 - 0 = 2$$

This satisfies the assumption, as the supports of sources  $\mathbf{y}_1$  and  $\mathbf{y}_2$  differ in at least two variables. If, however, both sources share the same support:

$$\text{supp}(\mathbf{y}_1) = \{1\}, \quad \text{supp}(\mathbf{y}_2) = \{1\}$$

Then the assumption would not hold because the supports are identical, and they do not differ by at least two observed variables.

**Sufficient Partial Selective Pairing Assumption (Assumption 1)** The Sufficient Partial Selective Pairing assumption requires that for each factor  $k \in [n]$ , there exist observations  $(\mathbf{x}, \mathbf{x}') \in \mathcal{X}$  such that the union of the shared support indices  $\mathbf{i} = \mathbf{I}(\mathbf{x}, \mathbf{x}')$  that do not include  $k$  must cover all other factors. Formally, we have:

$$\bigcup_{\mathbf{i} \in \mathcal{I} | k \notin \mathbf{i}} \mathbf{i} = [n] \setminus \{k\}, \quad \mathcal{I} := \{\mathbf{i} \subseteq [n] \mid p(\mathbf{i}) > 0\} \quad (\text{A.21})$$

Here,  $\mathcal{I}$  is the set of shared support indices, and  $p(\mathbf{i})$  is the probability that the factors indexed by  $\mathbf{i}$  are active, with  $k \notin \mathbf{i}$  inactive. The assumption ensures that when one factor is inactive, the shared support indices from the remaining factors provide enough information to reconstruct all active factors.

**Example of Sufficient Partial Selective Pairing Assumption** In the same scenario with three sources  $\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3$  and observed variables  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$ , we can define the shared support indices for each observation. Let's assume that the following shared support indices hold:

- Observation 1:  $\mathbf{i} = \{1, 2\}$  - Observation 2:  $\mathbf{i} = \{2, 3\}$  - Observation 3:  $\mathbf{i} = \{3, 4\}$

Now, for the Sufficient Partial Selective Pairing assumption to hold for factor  $k = 1$ , we must ensure that the union of the shared supports where factor 1 is inactive covers all other factors. For example, if we exclude  $k = 1$ , the union of the shared supports for the remaining factors should cover  $\mathbf{y}_2$  and  $\mathbf{y}_3$ :

$$\bigcup_{i|1 \notin i} i = \{2, 3, 4\} = [2, 3, 4]$$

This satisfies the assumption because when  $y_1$  is inactive, the shared support indices from  $y_2$  and  $y_3$  cover all remaining factors.

### Why the Sufficient Partial Selective Pairing Assumption is More Flexible

- It does not require the supports of every pair of sources to differ by exactly two observed variables.
- It only requires that when one factor is inactive, the shared support indices must still cover all other active factors, which allows for more overlap between the supports of different sources.
- This assumption is better suited for real-world scenarios where the supports of factors may not be completely distinct but still provide enough information to disentangle the factors.

In contrast, the Structural Sparsity assumption proposed in (Ng et al., 2023) can be too strict in cases where factors share common supports, and it would fail to identify factors in such cases.

**Example.1 (Assumption-1 fails)** This ensures distinct influences across observed variables. If the supports are nearly identical, Assumption-1 fails. For example, consider the mixing matrix  $A$ :

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} = \begin{bmatrix} 1 & 0.5 & 0 & 0.2 \\ 0.3 & 1 & 0.4 & 0 \\ 0 & 0.2 & 1 & 0.5 \\ 0.1 & 0 & 0.6 & 1 \end{bmatrix} \begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \\ y_4(t) \end{bmatrix} + \epsilon$$

with supports  $\text{supp}(a_1) = \{1, 2, 4\}$ ,  $\text{supp}(a_2) = \{1, 2, 3\}$ ,  $\text{supp}(a_3) = \{2, 3, 4\}$ , and  $\text{supp}(a_4) = \{1, 3, 4\}$ . For  $y_1$  and  $y_2$ , the difference in support is 2 (validating Assumption-1), as is the case for  $y_3$  and  $y_4$ . However, the significant overlap in the observed variables they influence ( $y_1$  and  $y_2$  both affect  $x_1(t)$ ,  $x_2(t)$ , and  $y_3$  and  $y_4$  affect  $x_3(t)$ ,  $x_4(t)$ ) limits the ability to uniquely identify each source, pointing to a practical challenge in real-world data.

## B EXPERIMENTS AND IMPLEMENTATION SETTINGS

### B.1 IMPLEMENTATION SOURCE. (TIMECSL-LIB)

We have implemented the ResTimeCSL architecture from scratch, and our code is available at <https://anonymous.4open.science/r/TimeCSL-4320>. Some components of our code are inspired by the following works:

- The GMM-based VAE sampling is inspired by VaDE (Jiang et al., 2016), and we adapted the implementation from <https://github.com/mperezcarrasco/Pytorch-VaDE>.
- For the Diffusion model D3VAE (Li et al., 2023), we utilized the authors’ implementation from <https://github.com/PaddlePaddle/PaddleSpatial/tree/main/research/D3VAE>.
- Regarding the methods listed in Tab. 3, the TCL model was adapted from <https://github.com/hmorioka/TCL/tree/master/tcl>, while the other models are derived from <https://github.com/rpatrik96/nl-causal>.
- For iVAE (Khemakhem et al., 2020b), we used the implementation available at <https://github.com/MatthewWilletts/algostability>.

Our experiments were conducted with 5 different random seeds, and we report the average results along with standard deviations. The experiments were run using 8 NVIDIA A100 GPUs.

## B.2 DATASETS.

In this section, we provide details about the datasets used for our experiments. We consider both real-world and synthetic datasets, each with specific characteristics relevant to the study. The table below summarizes the key properties of these datasets, including the number of samples, input dimensions, the number of sources/factors, and the names of the factors. The real-world datasets include REDD, REFIT, and UKDALE, which are commonly used in energy consumption modeling. Additionally, we employ synthetic datasets (Synthetic-1, Synthetic-2, and Synthetic-3) to simulate various scenarios with varying factors and input sizes. These datasets allow for comprehensive testing of our proposed method across different contexts.

Table 4: Synthetic and real-world datasets

Dataset	# Samples	Input Dim	# Sources/Factors	Factors name
REDD	5400	256	3	{FR, DW, WM, HTR, LT}
REFIT	1299	256	5	{FR, DW, WM, HTR, LT}
UKDALE	1300	256	5	{FR, DW, WM, HTR, LT}
Synthetic-1	12000	24	3	{FR, LT, HTR}
Synthetic-2	11000	96	5	{FR, LT, HTR}
Synthetic-3	11000	64	3	{FR, LT, HTR}
Synthetic-4	23000	256	5	{FR, DW, WM, HTR, LT}

## B.3 CONTRASTIVE PARTIAL SELECTIVE PAIRING - DATA AUGMENTATIONS

Four augmentations were sequentially applied to all contrastive methods’ pipeline branches. The parameters from the random search are: **1) Crop and delay:** applied with a 0.5 probability and a minimum size of 50% of the initial sequence. **2) Cutout or Masking:** time cutout of 5 steps with a 0.8 probability. **3) Channel Masks powers:** each time series is randomly masked out with a 0.4 probability. **4) Gaussian noise:** random Gaussian noise is added to window input  $\mathbf{x}$  with a standard deviation from 0.1 to 0.3. Further details in [App. B.3](#). Also in our experiments, we utilize a composition of three data augmentations, applied in the following order - scaling, shifting, and jittering, activating with a probability of 0.3 to 0.5.

**Scaling** The time-series is scaled by a single random scalar value, obtained by sampling  $\epsilon \sim \mathcal{N}(0, 0.5)$ , and each time step is  $\mathbf{x}'_t = \epsilon x_t$ .

**Shifting** The time-series is shifted by a single random scalar value, obtained by sampling  $\epsilon \sim \mathcal{N}(0, 0.5)$  and each time step is  $\mathbf{x}'_t = x_t + \epsilon$ .

**Jittering** I.I.D. Gaussian noise is added to each time step, from a distribution  $\epsilon_t \sim \mathcal{N}(0, 0.5)$ , where each time step is now  $\mathbf{x}'_t = x_t + \epsilon_t$ .

## B.4 IMPLEMENTATION OF METRICS AND STUDY CASE

Previous work has relied on the Mean Correlation Coefficient (MCC) as a metric to quantify identifiability. For consistency with previous work, we report this metric, but also propose a new metric to quantify identifiability up to an affine transformation. There are two challenges in designing such a metric: Firstly, for two Gaussian mixtures, standard distance metrics such as TV-distance or KL-divergence do not have a closed form. Secondly, we need to find an affine map  $A$  that best aligns a pair of Gaussian mixtures. Therefore, developing a metric to quantify identifiability up to an affine transformation has natural challenges. We propose  $d_{\text{aff}, L2}$ , defined below, as an additional metric in this setting.

#### B.4.1 ALIGNMENT PRIOR TO MEASURING WEAK MCC

We seek an affine map  $\Gamma$  to align two GMMs using two methods. One approach, used in previous works on MCC, is Canonical Correlation Analysis (CCA). Alternatively, we explore a different method. For two GMMs, we iterate over all permutations of the components, and for each permutation, we compute the optimal map  $\Gamma$  that aligns the components. While ideally  $\Gamma$  would align both the means and the covariance matrices, solving this as an optimization problem is challenging. Thus, we focus on aligning the means of the first GMM to those of the second GMM. The map  $\Gamma$  is found by solving the least-squares problem:

$$\min_{\Gamma} \sum_i \|\mu_1^{(i)} - \Gamma \mu_2^{(i)}\|^2 \quad (\text{B.1})$$

This can be efficiently solved using Singular Value Decomposition (SVD). Empirically, aligning the means provides good results.

#### B.4.2 MEASURING IDENTIFIABILITY STRONG-MCC AND WEAK-MCC

The other metric we consider is the Mean Correlation Coefficient (MCC) metric which had been used in prior works (Khemakhem et al., 2020a). There are two versions of MCC that have been used:

1. The *weak* MCC is defined to be the MCC after alignment via the affine map  $\Gamma$  transformation see App. B.4.1.
2. The *strong* MCC is defined to be the MCC before alignment.

Furthermore, in this work, we consider two different metrics. For a pair of distributions  $p_1, p_2$ , we define  $d_{\text{aff}, L_2}$  loss as

$$d_{\text{aff}, L_2}(p_1, p_2) = \min_{\substack{A: \mathbb{R}^m \rightarrow \mathbb{R}^m \\ \text{affine}}} \Delta_{L_2}(\Gamma_{\sharp} p_1, p_2), \quad \text{where} \quad \Delta_{L_2}(p_1, p_2) = \frac{\|p_1 - p_2\|_{L_2}}{\|p_1\|_{L_2}^{1/2} \|p_2\|_{L_2}^{1/2}} \quad (\text{B.2})$$

In our experiments, we report both the strong MCC and weak MCC. Moreover, all reported MCC s are out-of-sample, i.e. the optimal affine map  $\Gamma$  is computed over half the dataset and then reused for the other half of the dataset.

#### B.4.3 MEASURING DISENTANGLEMENT OF THE LEARNED REPRESENTATION

In implementing the disentanglement metrics, we adhere to the methodology outlined in (Locatello et al., 2019), expanding it to accommodate time series data. For the computation of DCI metrics, we employ a gradient boosted tree from the scikit-learn package.

**$\beta$ -VAE Metric** Disentanglement is then measured as the accuracy of a linear classifier that predicts the index of the fixed factor based on the coordinate-wise sum of absolute differences between the representation vectors in the two mini-batches. (Higgins et al., 2016) suggest fixing a random attributes of variation in the underlying generative model and sampling two mini-batches of observations  $\mathbf{x}$ . We sample two batches of 256 points with a random factor fixed to a randomly sampled value across the two batches, and the others varying randomly. We compute the mean representations for these points and take the absolute difference between pairs from the two batches. We then average these 64 values to form the features of a training (or testing) point.

**FactorVAE Metric (Kim & Mnih, 2019)** (Kim & Mnih, 2019) address several issues with this metric by using a majority vote classifier that predicts the index of the fixed ground-truth attribute based on the index of the representation vector with the least variance. First, we estimate the variance of each latent dimension by embedding  $10k$  random samples from the data set, excluding collapsed dimensions with variance smaller than .05. Second, we generate the votes for the majority vote classifier by sampling a batch of 64 points, all with a factor fixed to the same random value. Third, we compute the variance of each dimension of their latent representation and divide it by the variance of that dimension computed on the data without interventions. The training point for the majority vote classifier consists of the index of the dimension with the smallest normalized variance. We train on  $10k$  points and evaluate on  $5k$  points.

**Mutual Information Gap Metric (Chen et al., 2018b)**  $\beta$ -VAE metric and the FactorVAE metric are neither general nor unbiased as they depend on some hyperparameters (Chen et al., 2018b). They compute the mutual information between each ground-truth factor and each dimension in the computed representation  $r(\mathbf{x})$ . For each ground-truth factor  $z_k$ , they then consider the two dimensions in  $r(\mathbf{x})$  that have the highest and second highest mutual information with  $z_k$ . The Robust Mutual Information Gap (MIG) is then defined as the average, normalized difference between the highest and second highest mutual information of each factor with the dimensions of the representation. The original metric was proposed evaluating the sampled representation. Instead, we consider the mean representation, in order to be consistent with the other metrics. We estimate the mutual information by binning each dimension of the representations. Then, the score is computed as follows:

$$RMIG = \frac{1}{K} \sum_{k=1}^K [I(v_{jk}, z_k) - \max I(v_j, z_k)]$$

Where  $z_k$  is a factor of variation,  $v_i$  is a dimension of the latent representation. The MIG score of all factors are averaged to report one score.

**Disentanglement, Completeness and Informativeness (DCI)** In (Carbonneau et al., 2022), a framework is proposed to evaluate disentangled representations using metrics for modularity, compactness, and explicitness, referred to as disentanglement, completeness, and informativeness. Regressors predict factors from codes, with modularity and compactness estimated by importance weights  $R_{ij}$ . These weights are computed using a lasso regressor or random forests. The compactness for factor  $\mathbf{v}_i$  is defined as:

$$C_i = 1 + \sum_{j=1}^d p_{ij} \log_d p_{ij}, \quad p_{ij} = \frac{R_{ij}}{\sum_{k=1}^d R_{ik}}.$$

Compactness for the entire representation is the average over all factors. The modularity for code dimension  $\mathbf{z}_j$  is:

$$D_j = 1 + \sum_{i=1}^M p_{ij} \log_M p_{ij}, \quad p_{ij} = \frac{R_{ij}}{\sum_{k=1}^M R_{kj}}.$$

The modularity score is the weighted average over all code dimensions, with weights  $\rho_j$  reflecting their importance in predicting factors. Explicitness is defined by the MSE of the regressor, normalized between 0 and 1:

$$\text{Explicitness} = 1 - 6 \cdot \text{MSE}, \quad \text{MSE} = E[(\mathbf{x} - \mathbf{y})^2] = \frac{1}{6}.$$

**Time Disentanglement Score TDS** Time series data often exhibit variations that may not always align with conventional metrics, especially when considering the presence or absence of underlying attributes. To address this challenge, (Oubal et al., 2024) introduce the Time Disentanglement Score (TDS), a metric designed to assess the disentanglement of attributes in time series data. The foundation of TDS lies in an Information Gain perspective, which measures the reduction in entropy when an attribute is present compared to when it's absent.

$$TDS = \frac{1}{\dim(\mathbf{z})} \sum_{n \neq m} \sum_k \frac{\|z_m - z_{n,k}^+\|^2}{\text{Var}[z_m]}, \quad (\text{B.3})$$

In the context of TDS, we augment factor  $m$  in a time series window  $\mathbf{x}$  with a specific objective: to maintain stable entropy when the factor is present and reduce entropy when it's absent. This augmentation aims to capture the essence of attribute-related information within the data.

## B.5 RES TIMECSL ARCHITECTURE

The architecture employs multiple ResTimeCSL residual units Fig. 8 to model both the encoder and decoder for temporal sequential data. The input size is  $T = 256$  (time steps) with  $C = 1$  (features).

The encoder compresses the input into a latent representation of size  $n = 5 \times d = 16$ , while the decoder reconstructs the sequence into an output of size  $T = 256 \times n = 5$ . An additive layer is applied after decoding to sum the  $n$  components at each time step  $t$ , ensuring the output matches the input dimensions. Let  $\mathbf{x} \in \mathbb{R}^{T \times C}$  represent the input sequence. A linear patching operation is applied to preprocess the input:  $\mathbf{x}_{\text{patch}} = \text{LinearPatching}(\mathbf{x})$ . The encoder comprises multiple stacked "ResTimeCSL" residual units to map the input into a latent representation  $\mathbf{z} \in \mathbb{R}^{n \times d}$ , where  $n = 5$  and  $d = 16$ . Each "ResTimeCSL" block performs:

$$\mathbf{h}_{\text{out}} = \text{TCN}(\text{Affine}(\mathbf{h}_{\text{in}}) + \text{SkipConnections}),$$

with  $\mathbf{h}_{\text{in}}$  and  $\mathbf{h}_{\text{out}}$  denoting the input and output of a block, respectively. Similarly, the decoder uses multiple "ResTimeCSL" blocks to reconstruct the sequence, producing an output  $\mathbf{y} \in \mathbb{R}^{T \times n}$ , where  $n = 5$ . Finally, an additive layer combines the  $n$  components at each time step  $t$ :

$$\mathbf{y}_{\text{final}}(t) = \sum_{i=1}^n \mathbf{y}_i(t),$$

ensuring that the final output size matches the input:  $\mathbf{y}_{\text{final}} \in \mathbb{R}^{C \times T}$ , with  $C = 1$ . This hierarchical structure, powered by multiple "ResTimeCSL" units, ensures effective representation learning and reconstruction while maintaining temporal and feature dimensions.

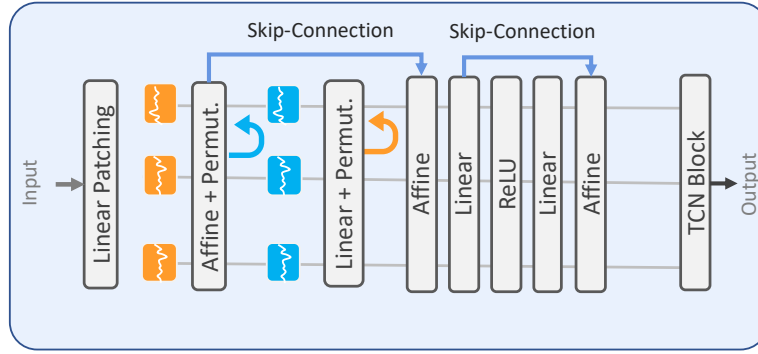


Figure 8: The residual unit ResTimeCSL, is employed in both the encoder and decoder.

The training process uses the Adamax (Kingma & Ba, 2014) optimizer with an initial learning rate of  $10^{-3}$  and  $\beta_1 = 0.9, \beta_2 = 0.999$ . A cosine annealing learning rate decay is applied to improve convergence

## B.6 PIPELINE CORRELATED SAMPLES.

Robustness of the model to correlations between data is assessed by examining different pairs. We focus mainly on linear correlations between two different devices and on the case where one device correlates with two others. To do this, we parameterize the correlations by sampling a dataset from the common distribution. We build on the correlation time series framework by introducing a pairwise correlation between the attributes  $y_m$  and  $y_n$  as follows:  $p(y_m, y_n) \propto \exp(-||y_m - \alpha y_n||^2 / 2\sigma^2)$ , where  $\alpha$  is a scaling factor. A high value of  $\sigma$  indicates a lower correlation between the normalised attributes  $y_m$  and  $y_n$  (No.Corr,  $\sigma = \infty$ ). We also extend this framework to cover correlations between several attributes in the time window  $T$ . Therefore, we consider correlation pair scenarios such as : *No correlation*; *Pair:1* washer-dryer; *Pair:2* dryer-oven and, finally, a *Random pair*: approach with randomly selected appliances.

## B.7 IMPACT OF ReLU/LeakyReLU AND ATTENTION LAYER WITH GELU ACTIVATION ON DECODER BEHAVIOR

In this study, we evaluate the impact of different activation functions on the decoder’s behavior to satisfies Asm 2.1. Specifically, we compare the use of ReLU (a piecewise affine activation) and GELU (a smooth, nonlinear activation) within an MLP decoder. The results suggest that the choice of activation function has a significant impact on the latent representation produced by the model.



**ReLU Activation:** The decoder becomes piecewise affine, meaning that it can be broken down into affine transformations over different regions of the input space. This causes the decoder to create latent representations that reflect distinct linear transformations in various regions of the input. As a result, the learned latent space is structured around these distinct affine regions, potentially making the model more sensitive to certain regions of the data space and leading to more discrete or sharply defined latent representations.

**LeakyReLU Activation:** In contrast, the GELU activation is smooth and nonlinear across the entire input space. This means that the decoder no longer operates piecewise affine, and the latent space learned by the model is more continuous and smooth. Since GELU smoothly transforms the input, it enables the decoder to create more nuanced, continuous latent representations. The absence of piecewise linear behavior allows for better modeling of complex, smooth relationships in the data, which may improve generalization to unseen data or tasks that require such smooth transformations.

## B.8 VALIDATION OF RESULTS ON SYNTHETIC DATA GENERATION

We simulate time-series data for energy disaggregation by leveraging the appliance signatures  $y_k \in \mathbb{R}^T$  from the REDD and REFIT datasets, where  $T$  is the number of time steps. The observed mixed signal  $x \in \mathbb{R}^T$  is generated as the sum of the individual appliance contributions, i.e.,  $x_t = \sum_{k=1}^n y_{k,t} + \epsilon_t$  where  $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$  is Gaussian noise. Each appliance signature  $y_k$  represents the time-series power consumption of appliance  $k$ , and these signatures are directly taken from the dataset. The final mixed signal  $x$  is the result of combining the contributions from multiple appliances, with each  $y_k$  corresponding to the power usage of a particular appliance in the dataset. This model serves as a foundation for evaluating energy disaggregation methods.

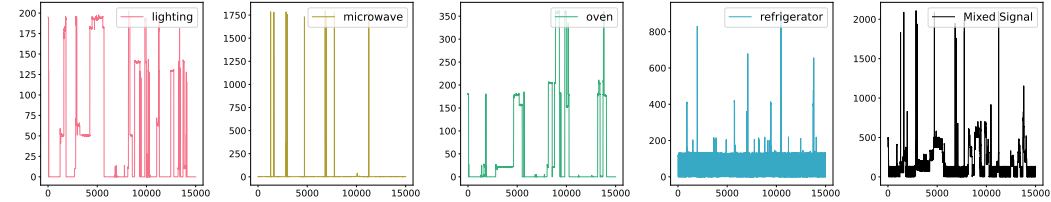


Figure 9: An example of a mixed signal from four sources in the REDD dataset.

## B.9 ADDITIONAL EXPERIMENT RESULTS.

### B.9.1 EXPERIMENT ON REDD AND REFIT DATASETS

*Remark B.1.* In [Tab. 6](#), we observe a similarity in metrics across the REDD and REFIT datasets (with 5 seed experiments), despite their differences, can be explained by the fact that certain factors, particularly "FR", are highly represented in both datasets. This suggests that these common factors capture underlying patterns relevant to both datasets, leading to similar model performance. However, factors like "LT" and "HTR" are less prominent, which means their influence on the results is smaller. To address this and more accurately evaluate our approach in real datasets, we consider a broader set of factors such as {FR, DW, WM, HTR, LT} for REDD and UKDALE datasets, which would better capture the unique characteristics of each dataset and provide a more nuanced evaluation.

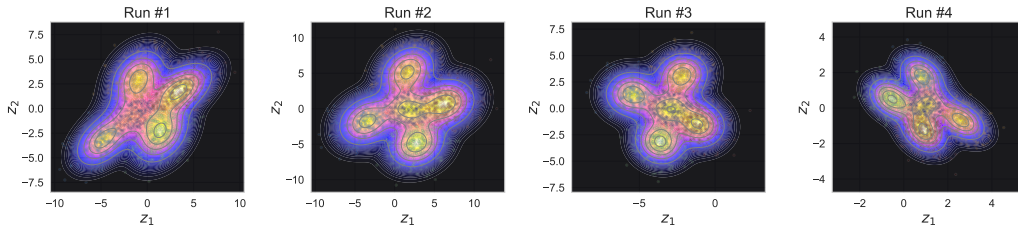


Figure 10: Recovered latent spaces for 4 runs of TimeCSL on REDD dataset with 5 latents ( $n = 5, d = 16$ ) {FR, DW, WM, HTR, LT}.



Table 5: Average performance, considering factors  $\{FR, DW, WM, HTR, LT\}$  with 5 seed on real datasets REDD and REFIT. Metrics reported are DCI, RMIG and RMSE. Lower values are better for all metrics. ( $\downarrow$  lower is better,  $\uparrow$  higher is worse Top-1 , Top-2 ).

Sc.	Methods	$\sigma = \infty$			$\sigma = 0.3$			$\sigma = 0.8$		
		DCI $\downarrow$	RMIG $\downarrow$	RMSE $\downarrow$	DCI $\downarrow$	RMIG $\downarrow$	RMSE $\downarrow$	DCI $\downarrow$	RMIG $\downarrow$	RMSE $\downarrow$
Synthetic-1	BertNILM	-	-	52.81 $\pm$ 25.41	-	-	75.78 $\pm$ 7.76	-	-	66.50 $\pm$ 6.69
	S2S	-	-	47.99 $\pm$ 24.45	-	-	63.64 $\pm$ 20.56	-	-	67.93 $\pm$ 15.57
	Autoformer	-	-	61.52 $\pm$ 7.66	-	-	52.23 $\pm$ 11.25	-	-	48.45 $\pm$ 9.31
	Informer	-	-	48.59 $\pm$ 10.89	-	-	59.29 $\pm$ 11.36	-	-	63.45 $\pm$ 10.52
	TimesNet	-	-	63.57 $\pm$ 10.61	-	-	67.02 $\pm$ 9.10	-	-	69.93 $\pm$ 9.89
	C-DSVAE	72.83 $\pm$ 11.71	1.08 $\pm$ 0.45	40.50 $\pm$ 6.45	71.76 $\pm$ 9.74	1.08 $\pm$ 0.44	51.67 $\pm$ 7.88	72.64 $\pm$ 10.89	1.23 $\pm$ 0.51	55.26 $\pm$ 7.80
	SlowVAE	82.31 $\pm$ 11.96	1.08 $\pm$ 0.47	43.46 $\pm$ 7.93	81.65 $\pm$ 10.75	1.08 $\pm$ 0.46	54.81 $\pm$ 5.93	84.09 $\pm$ 6.93	1.27 $\pm$ 0.49	53.65 $\pm$ 7.48
	CoST	79.86 $\pm$ 10.86	1.16 $\pm$ 0.23	50.14 $\pm$ 6.77	79.16 $\pm$ 10.49	1.15 $\pm$ 0.22	55.91 $\pm$ 5.72	80.16 $\pm$ 9.68	1.25 $\pm$ 0.20	58.76 $\pm$ 5.51
	SlowVAE+HDF	88.69 $\pm$ 1.11	1.11 $\pm$ 0.24	65.87 $\pm$ 8.13	85.99 $\pm$ 1.34	0.97 $\pm$ 0.21	69.94 $\pm$ 7.29	89.47 $\pm$ 0.58	1.14 $\pm$ 0.24	72.21 $\pm$ 7.47
	C-DSVAE + HDF	76.94 $\pm$ 6.38	0.89 $\pm$ 0.37	33.61 $\pm$ 5.80	75.66 $\pm$ 6.53	0.84 $\pm$ 0.33	37.92 $\pm$ 5.88	74.45 $\pm$ 5.78	0.89 $\pm$ 0.40	42.58 $\pm$ 6.49
Synthetic-2	SparseVAE	71.35 $\pm$ 8.48	0.67 $\pm$ 0.25	26.46 $\pm$ 5.68	72.67 $\pm$ 8.54	0.68 $\pm$ 0.27	31.07 $\pm$ 5.34	73.98 $\pm$ 8.23	0.74 $\pm$ 0.29	32.56 $\pm$ 5.16
	TimeCSL	75.44 $\pm$ 6.93	0.59 $\pm$ 0.17	25.53 $\pm$ 6.69	74.50 $\pm$ 6.29	0.61 $\pm$ 0.19	29.23 $\pm$ 6.57	76.66 $\pm$ 5.70	0.74 $\pm$ 0.16	33.76 $\pm$ 6.73
	BertNILM	-	-	60.83 $\pm$ 5.80	-	-	72.63 $\pm$ 2.25	-	-	71.02 $\pm$ 2.55
	S2S	-	-	53.73 $\pm$ 5.84	-	-	65.57 $\pm$ 5.35	-	-	69.21 $\pm$ 4.06
	Autoformer	-	-	54.60 $\pm$ 1.70	-	-	50.48 $\pm$ 2.82	-	-	50.39 $\pm$ 2.26
	Informer	-	-	45.92 $\pm$ 3.03	-	-	53.77 $\pm$ 2.86	-	-	61.08 $\pm$ 2.51
	TimesNet	-	-	54.68 $\pm$ 3.68	-	-	55.28 $\pm$ 3.02	-	-	59.24 $\pm$ 3.41
	C-DSVAE	74.83 $\pm$ 5.72	1.12 $\pm$ 0.23	47.04 $\pm$ 3.14	73.42 $\pm$ 2.40	1.10 $\pm$ 0.21	53.02 $\pm$ 3.49	75.29 $\pm$ 3.34	1.21 $\pm$ 0.14	54.81 $\pm$ 3.46
	SlowVAE	80.92 $\pm$ 2.73	1.10 $\pm$ 0.20	44.58 $\pm$ 3.11	79.95 $\pm$ 2.64	1.09 $\pm$ 0.18	51.92 $\pm$ 2.58	81.45 $\pm$ 1.37	1.21 $\pm$ 0.14	50.69 $\pm$ 2.99
	CoST	71.18 $\pm$ 3.83	1.04 $\pm$ 0.06	47.10 $\pm$ 1.66	71.01 $\pm$ 3.86	1.05 $\pm$ 0.05	53.58 $\pm$ 1.39	70.56 $\pm$ 3.50	1.14 $\pm$ 0.04	55.29 $\pm$ 1.22
	SlowVAE+HDF	81.13 $\pm$ 0.17	0.85 $\pm$ 0.08	60.50 $\pm$ 3.01	80.21 $\pm$ 0.19	0.79 $\pm$ 0.07	62.72 $\pm$ 2.77	81.68 $\pm$ 0.10	0.89 $\pm$ 0.05	64.03 $\pm$ 2.99
	C-DSVAE + HDF	74.77 $\pm$ 1.56	0.78 $\pm$ 0.05	35.62 $\pm$ 2.52	74.39 $\pm$ 1.51	0.75 $\pm$ 0.05	38.40 $\pm$ 1.83	74.88 $\pm$ 0.98	0.79 $\pm$ 0.07	39.95 $\pm$ 1.62
	SparseVAE	69.84 $\pm$ 4.10	0.62 $\pm$ 0.06	27.28 $\pm$ 2.59	69.95 $\pm$ 4.15	0.60 $\pm$ 0.05	29.61 $\pm$ 1.67	72.52 $\pm$ 3.77	0.65 $\pm$ 0.07	30.35 $\pm$ 1.45
	TimeCSL	71.72 $\pm$ 3.23	0.46 $\pm$ 0.04	25.02 $\pm$ 2.77	71.21 $\pm$ 2.58	0.51 $\pm$ 0.03	25.91 $\pm$ 2.62	72.68 $\pm$ 2.33	0.61 $\pm$ 0.02	28.82 $\pm$ 2.83

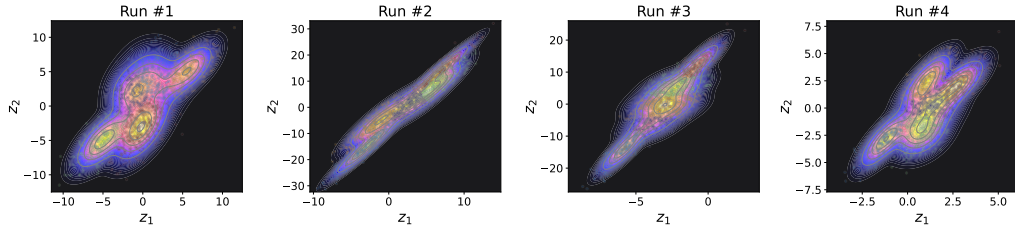


Figure 11: Recovered latent spaces for 4 runs of TDRL on REDD dataset with 5 latents ( $n = 5, d = 16$ )  $\{FR, DW, WM, HTR, LT\}$ .

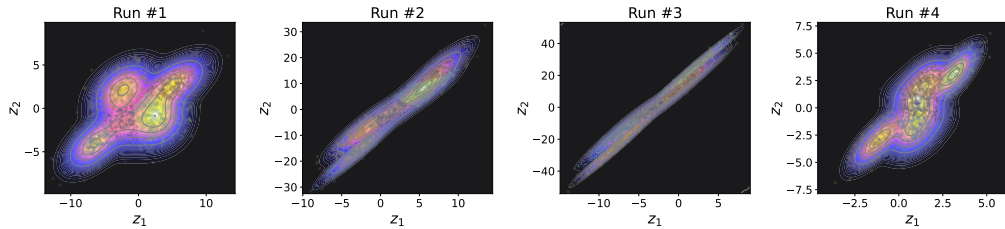


Figure 12: Recovered latent spaces for 4 runs of SlowVAE on REDD dataset with 5 latents ( $n = 5, d = 16$ )  $\{FR, DW, WM, HTR, LT\}$ .

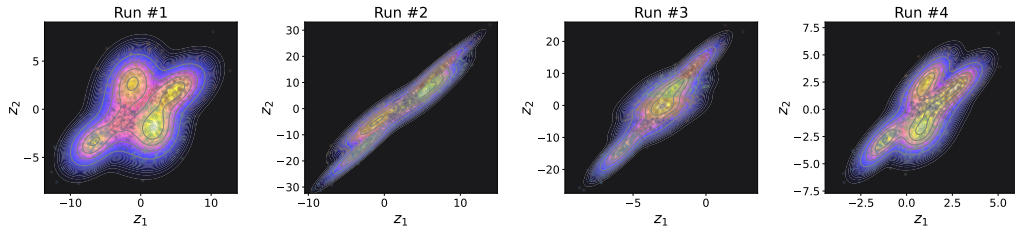


Figure 13: Recovered latent spaces for 4 runs of iVAE on REDD dataset with 5 latents ( $n = 5, d = 16$ )  $\{FR, DW, WM, HTR, LT\}$ .

### B.9.2 EXPERIMENT ON SYNTHETIC DATASETS

Table 6: Average performance, considering factors {FR, LT, HTR} with 5 seed on synthetics datasets (1 & 2). Metrics reported are: DCI, RMIG and RMSE. Lower values are better for all metrics. ( $\downarrow$  lower is better,  $\uparrow$  higher is worse Top-1, Top-2).

Sc.	Methods	$\sigma = \infty$			$\sigma = 0.3$			$\sigma = 0.8$		
		DCI $\downarrow$	RMIG $\downarrow$	RMSE $\downarrow$	DCI $\downarrow$	RMIG $\downarrow$	RMSE $\downarrow$	DCI $\downarrow$	RMIG $\downarrow$	RMSE $\downarrow$
Synthetic-1	BertNILM	-	-	36.86 $\pm$ 1.68	-	-	45.84 $\pm$ 1.00	-	-	46.29 $\pm$ 0.76
	S2S	-	-	35.46 $\pm$ 2.04	-	-	45.36 $\pm$ 2.47	-	-	45.76 $\pm$ 2.26
	Autoformer	-	-	32.45 $\pm$ 0.56	-	-	33.02 $\pm$ 1.49	-	-	34.68 $\pm$ 1.13
	Informer	-	-	32.92 $\pm$ 1.67	-	-	35.03 $\pm$ 1.71	-	-	38.47 $\pm$ 1.54
	TimesNet	-	-	32.12 $\pm$ 1.99	-	-	33.38 $\pm$ 1.83	-	-	35.84 $\pm$ 1.61
	CoST	44.68 $\pm$ 1.57	0.61 $\pm$ 0.02	31.14 $\pm$ 0.93	48.01 $\pm$ 1.57	0.64 $\pm$ 0.09	34.81 $\pm$ 0.71	46.98 $\pm$ 1.13	0.65 $\pm$ 0.01	38.14 $\pm$ 0.57
	SlowVAE	50.96 $\pm$ 0.71	0.61 $\pm$ 0.09	28.26 $\pm$ 1.54	53.04 $\pm$ 1.26	0.61 $\pm$ 0.09	32.15 $\pm$ 0.78	52.14 $\pm$ 0.58	0.70 $\pm$ 0.08	35.74 $\pm$ 1.03
	SlowVAE+HDF	52.17 $\pm$ 0.07	0.42 $\pm$ 0.02	37.35 $\pm$ 1.49	53.00 $\pm$ 0.12	0.46 $\pm$ 0.05	38.86 $\pm$ 1.26	52.53 $\pm$ 0.03	0.47 $\pm$ 0.01	40.22 $\pm$ 1.06
	TDRL	42.34 $\pm$ 1.02	0.28 $\pm$ 0.04	18.64 $\pm$ 1.41	49.75 $\pm$ 0.87	0.31 $\pm$ 0.01	17.18 $\pm$ 1.36	50.43 $\pm$ 0.69	0.38 $\pm$ 0.08	20.91 $\pm$ 1.07
	D3VAE	41.30 $\pm$ 1.97	0.26 $\pm$ 0.05	27.64 $\pm$ 1.40	41.55 $\pm$ 0.91	0.33 $\pm$ 0.26	30.11 $\pm$ 1.10	43.47 $\pm$ 1.31	0.44 $\pm$ 0.03	32.77 $\pm$ 0.51
	C-DSVAE	47.35 $\pm$ 2.14	0.59 $\pm$ 0.05	31.78 $\pm$ 1.61	47.79 $\pm$ 0.99	0.62 $\pm$ 0.26	34.55 $\pm$ 1.18	50.02 $\pm$ 1.42	0.71 $\pm$ 0.03	37.57 $\pm$ 0.53
	C-DSVAE + HDF	44.31 $\pm$ 1.93	0.56 $\pm$ 0.05	29.68 $\pm$ 1.51	45.01 $\pm$ 0.92	0.59 $\pm$ 0.25	32.42 $\pm$ 1.04	46.68 $\pm$ 1.33	0.66 $\pm$ 0.03	35.12 $\pm$ 0.50
	SparseVAE	40.15 $\pm$ 0.86	0.25 $\pm$ 0.09	13.72 $\pm$ 1.30	43.98 $\pm$ 0.81	0.28 $\pm$ 0.21	14.81 $\pm$ 1.20	44.53 $\pm$ 0.58	0.31 $\pm$ 0.07	18.89 $\pm$ 1.30
	TimeCSL	39.02 $\pm$ 0.87	0.23 $\pm$ 0.07	12.03 $\pm$ 1.26	42.51 $\pm$ 0.74	0.27 $\pm$ 0.15	12.72 $\pm$ 1.16	42.91 $\pm$ 0.59	0.31 $\pm$ 0.05	14.76 $\pm$ 0.92
	Avg.	45.62 $\pm$ 1.27	0.52 $\pm$ 0.07	31.02 $\pm$ 1.26	48.02 $\pm$ 0.85	0.58 $\pm$ 0.12	34.08 $\pm$ 1.04	48.92 $\pm$ 1.18	0.64 $\pm$ 0.06	35.67 $\pm$ 0.91
Synthetic-2	BertNILM	-	-	40.06 $\pm$ 2.41	-	-	44.14 $\pm$ 1.22	-	-	45.04 $\pm$ 0.99
	S2S	-	-	38.48 $\pm$ 2.87	-	-	45.07 $\pm$ 2.71	-	-	46.22 $\pm$ 2.26
	Autoformer	-	-	33.56 $\pm$ 0.79	-	-	34.13 $\pm$ 2.07	-	-	37.51 $\pm$ 1.81
	Informer	-	-	36.02 $\pm$ 2.37	-	-	37.61 $\pm$ 1.98	-	-	38.81 $\pm$ 2.36
	TimesNet	-	-	36.69 $\pm$ 2.08	-	-	39.08 $\pm$ 2.71	-	-	42.55 $\pm$ 2.35
	CoST	50.87 $\pm$ 1.13	0.58 $\pm$ 0.06	28.93 $\pm$ 1.81	53.10 $\pm$ 1.23	0.61 $\pm$ 0.14	30.72 $\pm$ 1.31	52.63 $\pm$ 1.19	0.67 $\pm$ 0.14	33.15 $\pm$ 1.12
	SlowVAE	48.11 $\pm$ 1.06	0.45 $\pm$ 0.05	31.73 $\pm$ 2.19	50.15 $\pm$ 1.35	0.47 $\pm$ 0.06	34.12 $\pm$ 1.57	50.97 $\pm$ 0.78	0.55 $\pm$ 0.02	35.27 $\pm$ 1.06
	SlowVAE + HDF	51.09 $\pm$ 1.64	0.34 $\pm$ 0.04	32.85 $\pm$ 2.40	51.97 $\pm$ 1.07	0.39 $\pm$ 0.05	35.72 $\pm$ 2.17	51.85 $\pm$ 1.58	0.43 $\pm$ 0.06	37.38 $\pm$ 2.51
	TDRL	45.12 $\pm$ 2.15	0.39 $\pm$ 0.05	22.87 $\pm$ 1.36	50.61 $\pm$ 1.53	0.44 $\pm$ 0.03	23.98 $\pm$ 1.41	51.18 $\pm$ 0.90	0.49 $\pm$ 0.08	27.13 $\pm$ 2.30
	D3VAE	43.77 $\pm$ 1.31	0.36 $\pm$ 0.06	28.43 $\pm$ 1.61	46.17 $\pm$ 0.86	0.39 $\pm$ 0.04	30.14 $\pm$ 1.35	48.02 $\pm$ 1.23	0.44 $\pm$ 0.06	32.46 $\pm$ 1.10
	C-DSVAE	49.68 $\pm$ 2.12	0.55 $\pm$ 0.07	31.03 $\pm$ 2.15	49.92 $\pm$ 1.05	0.58 $\pm$ 0.08	33.60 $\pm$ 1.77	51.51 $\pm$ 1.76	0.61 $\pm$ 0.03	35.38 $\pm$ 1.42
	C-DSVAE + HDF	47.38 $\pm$ 1.19	0.53 $\pm$ 0.05	30.76 $\pm$ 2.13	48.85 $\pm$ 1.62	0.56 $\pm$ 0.03	32.89 $\pm$ 2.04	49.98 $\pm$ 1.34	0.60 $\pm$ 0.05	34.25 $\pm$ 1.22
	SparseVAE	46.56 $\pm$ 2.49	0.44 $\pm$ 0.08	19.88 $\pm$ 2.06	50.49 $\pm$ 1.07	0.47 $\pm$ 0.06	21.42 $\pm$ 2.53	50.83 $\pm$ 1.73	0.53 $\pm$ 0.05	23.59 $\pm$ 2.17
	TimeCSL	43.45 $\pm$ 1.12	0.33 $\pm$ 0.02	16.32 $\pm$ 2.16	47.33 $\pm$ 1.29	0.35 $\pm$ 0.04	17.22 $\pm$ 2.01	48.09 $\pm$ 0.81	0.39 $\pm$ 0.06	18.95 $\pm$ 2.08
	Avg.	47.02 $\pm$ 1.56	0.45 $\pm$ 0.06	28.04 $\pm$ 1.84	50.43 $\pm$ 1.19	0.48 $\pm$ 0.09	30.32 $\pm$ 1.56	50.95 $\pm$ 1.26	0.54 $\pm$ 0.07	32.83 $\pm$ 1.57

Table 7: Average performance, considering factors {FR, DW, WM, HTR, LT} with 5 seed on synthetics datasets. Metrics reported are DCI, RMIG and RMSE. Lower values are better for all metrics. ( $\downarrow$  lower is better,  $\uparrow$  higher is worse Top-1, Top-2).

Sc.	Methods	$\sigma = \infty$			$\sigma = 0.3$			$\sigma = 0.8$		
		DCI $\downarrow$	RMIG $\downarrow$	RMSE $\downarrow$	DCI $\downarrow$	RMIG $\downarrow$	RMSE $\downarrow$	DCI $\downarrow$	RMIG $\downarrow$	RMSE $\downarrow$
Synthetic-3	BertNILM	-	-	56.4 $\pm$ 2.58	-	-	70.2 $\pm$ 1.45	-	-	70.92 $\pm$ 1.15
	S2S	-	-	54.3 $\pm$ 3.12	-	-	69.5 $\pm$ 3.56	-	-	69.95 $\pm$ 3.26
	Autoformer	-	-	49.7 $\pm$ 0.81	-	-	50.5 $\pm$ 2.15	-	-	52.95 $\pm$ 1.63
	Informer	-	-	50.3 $\pm$ 2.41	-	-	53.5 $\pm$ 1.98	-	-	58.95 $\pm$ 1.89
	FEDformer	-	-	50.3 $\pm$ 2.12	-	-	52.5 $\pm$ 2.45	-	-	59.01 $\pm$ 1.76
	TimesNet	-	-	49.24 $\pm$ 2.87	-	-	51.10 $\pm$ 2.64	-	-	54.91 $\pm$ 2.31
	C-DSVAE	72.42 $\pm$ 3.10	0.96 $\pm$ .15	48.6 $\pm$ 2.32	73.12 $\pm$ 1.43	0.95 $\pm$ .15	52.9 $\pm$ 2.31	74.29 $\pm$ 2.04	1.08 $\pm$ .09	52.99 $\pm$ 1.91
	SlowVAE	78.0 $\pm$ 1.09	0.94 $\pm$ .13	43.2 $\pm$ 2.23	78.0 $\pm$ 1.09	0.94 $\pm$ .13	49.2 $\pm$ 1.13	79.74 $\pm$ 0.84	1.07 $\pm$ .11	49.65 $\pm$ 1.43
	CoST	68.4 $\pm$ 2.41	0.97 $\pm$ .03	47.7 $\pm$ 1.35	68.4 $\pm$ 2.41	0.97 $\pm$ .03	53.2 $\pm$ 1.02	69.95 $\pm$ 1.63	1.00 $\pm$ .02	53.45 $\pm$ 0.82
	SlowVAE+HDF	79.8 $\pm$ .10	0.64 $\pm$ .05	57.2 $\pm$ 2.15	79.8 $\pm$ .10	0.64 $\pm$ .05	61.3 $\pm$ 1.82	80.37 $\pm$ .05	0.72 $\pm$ .03	61.64 $\pm$ 1.52
	C-DSVAE + HDF	73.1 $\pm$ 1.01	0.69 $\pm$ .02	34.4 $\pm$ 1.89	73.1 $\pm$ 1.01	0.69 $\pm$ .02	38.1 $\pm$ 1.34	74.25 $\pm$ 0.59	0.73 $\pm$ .05	38.48 $\pm$ 1.04
	SparseVAE	67.2 $\pm$ 2.01	0.52 $\pm$ .02	24.3 $\pm$ 1.81	67.2 $\pm$ 2.01	0.52 $\pm$ .02	27.4 $\pm$ 1.13	71.79 $\pm$ 1.27	0.58 $\pm$ .04	27.77 $\pm$ 0.83
	TimeCSL	63.5 $\pm$ 1.35	0.38 $\pm$ .02	19.6 $\pm$ 1.95	69.3 $\pm$ 1.2	0.44 $\pm$ .02	20.3 $\pm$ 1.79	70.12 $\pm$ 0.91	0.51 $\pm$ .01	23.63 $\pm$ 1.49
Synthetic-4	BertNILM	-	-	61.42 $\pm$ 3.47	-	-	67.61 $\pm$ 1.95	-	-	69.06 $\pm$ 1.43
	S2S	-	-	59.08 $\pm$ 4.15	-	-	68.60 $\pm$ 3.91	-	-	70.68 $\pm$ 3.25
	Autoformer	-	-	49.87 $\pm$ 0.92	-	-	51.53 $\pm$ 1.48	-	-	51.88 $\pm$ 1.34
	Informer	-	-	54.23 $\pm$ 1.78	-	-	57.70 $\pm$ 1.78	-	-	62.51 $\pm$ 1.55
	FEDformer	-	-	52.84 $\pm$ 1.69	-	-	55.83 $\pm$ 1.82	-	-	61.92 $\pm$ 1.57
	TimesNet	-	-	51.37 $\pm$ 2.41	-	-	55.35 $\pm$ 2.23	-	-	58.47 $\pm$ 2.21
	C-DSVAE	72.97 $\pm$ 3.44	1.04 $\pm$ 0.16	47.17 $\pm$ 2.11	73.60 $\pm$ 1.82	0.98 $\pm$ 0.14	52.16 $\pm$ 1.89	73.96 $\pm$ 2.46	1.11 $\pm$ 0.12	53.73 $\pm$ 1.79
	SlowVAE	77.41 $\pm$ 1.67	0.94 $\pm$ 0.15	46.61 $\pm$ 1.91	77.80 $\pm$ 1.63	0.95 $\pm$ 0.14	49.82 $\pm$ 1.71	79.47 $\pm$ 1.26	1.04 $\pm$ 0.13	50.88 $\pm$ 1.58
	CoST	70.75 $\pm$ 2.01	0.96 $\pm$ 0.09	48.92 $\pm$ 1.62	70.87 $\pm$ 2.04	0.96 $\pm$ 0.09	52.73 $\pm$ 1.34	71.93 $\pm$ 1.84	0.98 $\pm$ 0.09	54.46 $\pm$ 1.19
	SlowVAE+HDF	79.97 $\pm$ 0.14	0.72 $\pm$ 0.05	56.96 $\pm$ 2.34	79.77 $\pm$ 0.14	0.72 $\pm$ 0.05	59.75 $\pm$ 2.21	80.22 $\pm$ 0.07	0.75 $\pm$ 0.03	60.77 $\pm$ 2.22
	C-DSVAE + HDF	73.85 $\pm$ 0.85	0.69 $\pm$ 0.05	34.19 $\pm$ 1.47	73.71 $\pm$ 0.85	0.69 $\pm$ 0.05	37.53 $\pm$ 1.21	74.34 $\pm$ 0.56	0.71 $\pm$ 0.04	39.35 $\pm$ 1.06
	TDRL	70.86 $\pm$ 0.816	0.57 $\pm$ 0.041	32.80 $\pm$ 1.41	70.75 $\pm$ 0.816	0.57 $\pm$ 0.041	36.04 $\pm$ 1.16	71.94 $\pm$ 0.54	0.58 $\pm$ 0.033	37.83 $\pm$ 1.02
	SparseVAE	70.13 $\pm$ 1.44	0.61 $\pm$ 0.04	25.46 $\pm$ 1.10	70.13 $\pm$ 1.44	0.61 $\pm$ 0.04	28.99 $\pm$ 1.22	71.44 $\pm$ 1.30	0.63 $\pm$ 0.05	29.47 $\pm$ 1.10
	TimeCSL	66.14 $\pm$ 1.66	0.40 $\pm$ 0.04	19.81 $\pm$ 1.29	69.00 $\pm$ 1.41	0.44 $\pm$ 0.04	20.46 $\pm$ 1.45	70.41 $\pm$ 1.22	0.48 $\pm$ 0.03	22.08 $\pm$ 1.36

### B.9.3 COMPARISONS BETWEEN TIMECSL AND BASELINES ON KITTI DATASET

We evaluate TimeCSL on time-sequential data using preprocessed frames from the KITTI and MOTChallenge datasets. The original KITTI image resolutions are  $1080 \times 1920$  or  $480 \times 640$  for MOTChallenge, and between 370–374 pixels tall by 1224–1242 pixels wide for KITTI MOTS. The video frame rates vary from 14 to 30 fps, as described in (Milan, 2016). To preprocess the data, we apply nearest-neighbor down-sampling to reduce each frame’s height to 64 pixels while maintaining the aspect ratio for the width. Using a horizontal sliding window, we extract six equally spaced windows of size  $64 \times 64$  (with overlap) from each sequence in both datasets. This preprocessing produces a sequence of shape  $64 \times 64 \times T$ , where  $T$  represents the number of time steps in the sequence. Our approach assumes reasonable invariance to horizontal translation and scale within

the dataset. Scale invariance is supported by the fact that the data was collected from a car-mounted camera, leading to varying distances to pedestrians. To validate translation invariance, we conducted an ablation study on the number of horizontal sliding windows. Using only two horizontally spaced windows, instead of six, resulted in no significant changes in key statistics, such as kurtosis (remaining within  $\pm 10\%$  of the original value for  $\Delta x$  transitions). This experiment results Fig. 14 demonstrates the robustness of TimeCSL to time-sequential data, showcasing its potential for applications beyond its original domain.

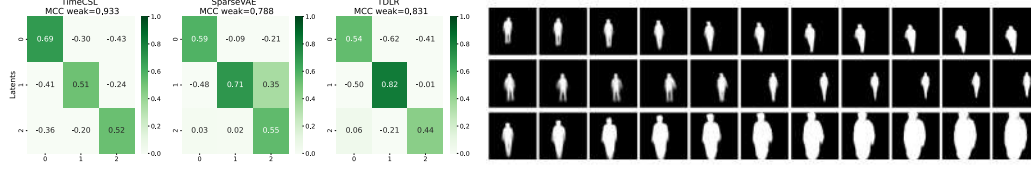


Figure 14: **Validation on KITTI dataset.** **Left.** MCC correlation matrix of the top 3 latents corresponding to y-position (1), x-position (2) and scale (3). **Right.** Images produced by varying the TimeCSL latent unit that corresponds to the corresponding row in the MCC matrix.