NON-REDUNDANT TIME-SERIES CLUSTERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Traditional time-series clustering methods typically return a single grouping. However, especially with the growing size of time-series data, a single dataset can have more than one meaningful clustering. Therefore, we introduce a non-redundant time-series clustering approach that explicitly uncovers multiple distinct clusterings within the same temporal data. To achieve this, the architecture employs a single-encoder, multiple-decoders design and is trained with a consistency clustering loss. Each decoder is then responsible for a single individual clustering. By independently masking and routing latent representations to these distinct decoders, this design acts as a tool for interpretability. It allows researchers to reconstruct the temporal sequence from isolated embeddings using a dedicated decoder, enabling explicit understanding of the perspectives and structural patterns that drive each separate clustering criterion. Furthermore, we introduce new non-redundant time-series clustering datasets and also collect existing benchmarks that already have multiple labelings. This provides a foundation for future non-redundant time-series analysis. Our evaluation shows that our method outperforms other methods on these datasets.

1 INTRODUCTION

Should a collection of time-series be clustered by trend or by how fast the series oscillates? Both clusterings can be correct (Figure 1), yet existing time-series clustering algorithms only return a single partition. They must commit to a single notion of similarity, and the strongest signal in the data can decide which one: a dominant pattern, such as a large amplitude or a regular seasonal cycle, can take over the similarity computation and hide the remaining structure. The analyst receives one clustering and has no indication that others exist.

This matters because clustering is mostly used when little is known about the data. It is a standard tool for exploring unlabeled temporal data, applied, for example, to discover patient subtypes from clinical recordings (Chen et al., 2022), to uncover regimes in stock returns (Dias et al., 2015), and to identify metastable atmospheric patterns (Horenko, 2010), and its output is interpreted as a description of the data. As time-series collections grow, it is increasingly likely that a single dataset contains multiple meaningful clusterings. Real recordings show that PTB-XL, a large electrocardiogram dataset, is annotated separately with statements on diagnosis, rhythm, and waveform form (Wagner et al., 2020), and any one category alone defines a valid clustering criterion. Changing the number of clusters or the random seed usually alters clustering granularity, not the aspects it is based on.

Finding several distinct clusterings of one dataset is the goal of non-redundant clustering (Yu et al., 2024). Existing methods were developed for tabular and image data, where they cluster the data by different learned aspects, such as color or shape. To our knowledge, non-redundant clustering has not been studied specifically for time-series. There, the task is harder for two reasons: first, the aspects are entangled in the raw values. Trend, seasonality, and amplitude all act on the same measurements, and none of them is a separate feature that could be selected. Second, the aspects are usually not known in advance, so methods that require the user to specify them cannot be applied. We therefore study *non-redundant time-series clustering*: given a set of unlabeled univariate time-series, find several clusterings that each reflect a different aspect of the series, without being told what these aspects are.

We propose NRTSC (**N**on-**r**edundant **T**ime-**s**eries **C**lustering), which learns these aspects from the data. NRTSC combines a single encoder with multiple decoders, one per clustering, and is

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

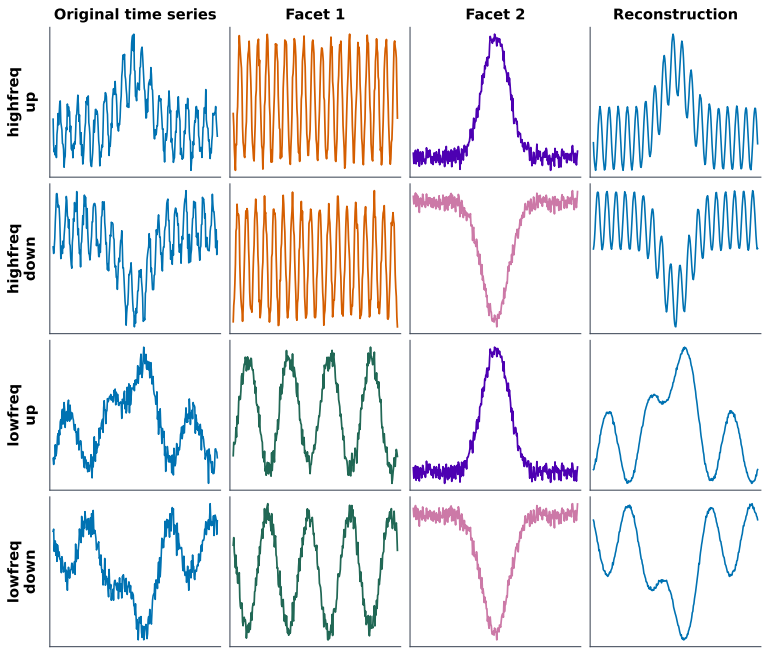


Figure 1: Non-redundant time-series clustering. The left column shows a dataset whose series differ in the direction of their trend and in their dominant frequency. Non-redundant time-series clustering (NRTSC) returns both clusterings (second and third columns). The decoder assigned to each clustering reconstructs the series from that clustering's embedding, which shows the pattern it is based on. The sum of the reconstructed parts is the total reconstruction of the signal (fourth column).

trained with a consistency clustering loss Chen & He (2021). Latent masks select clustering-specific representations for separate decoders, whose outputs are summed to reconstruct the input. Non-redundancy is encouraged by penalizing squared cosine similarity between reconstructed components and cross-facet dependence through a linear-kernel HSIC loss between one facet's predictions and the others' masked representation. The decoders also make the result inspectable: each one reconstructs the series from the embedding of a single clustering, which shows what temporal pattern this clustering is based on. In Figure 1, NRTSC recovers both the trend-based and the frequency-based clusterings.

Evaluating non-redundant clustering requires multiple reference labelings for the same time-series, whereas standard benchmarks typically provide only one (Dau et al., 2019). We assemble a benchmark comprising six datasets derived from real-world observations, two existing synthetic benchmarks, and three newly constructed datasets. Alternative reference labelings are obtained by aligning existing annotations or decomposing documented class definitions; the newly constructed datasets provide additional controlled evaluations. On this collection, NRTSC outperforms the evaluated competing methods. Our contributions are as follows:

- **Task.** We introduce non-redundant time-series clustering: finding multiple distinct clusterings in one unlabeled univariate time-series dataset without user-specified aspects.
- **Method.** We propose NRTSC, a single-encoder multiple-decoder architecture trained with a consistency clustering loss. Masking and routing of the latent representation give each clustering its own decoder, which makes each clustering inspectable.
- **Data.** We assemble an eleven-dataset benchmark suing eight existing real-world and synthetic datasets and three newly constructed ones, with multiple reference labelings for each.
- **Evidence.** In our experiments, NRTSC outperforms competing methods on these datasets. We restrict ourselves to univariate time-series and leave the multivariate case for future work.

2 Related Work

Our work connects deep time-series clustering, which typically targets a single partition, with non-redundant clustering, which discovers multiple complementary partitions and has been studied primarily on tabular and image data. We briefly review both areas.

2.1 Time-series clustering

Classical time-series clustering combines a similarity measure with a clustering algorithm. Dynamic time warping (DTW) (Sakoe & Chiba, 1978; Berndt & Clifford, 1994) aligns series for comparison, DTW barycenter averaging (Petitjean et al., 2011) computes corresponding centroids, and k-Shape (Paparrizos & Gravano, 2015) uses normalized cross-correlation to cluster by shape. Deep methods instead learn embeddings, either before clustering (Baytas et al., 2017) or jointly with a clustering objective (Ma et al., 2019; Chen et al., 2022). TS2Vec (Yue et al., 2022) and TS-TCC (Eldele et al., 2021) learn contrastive representations that can subsequently be clustered. DTCC (Zhong et al., 2023) combines instance- and cluster-level contrast (Li et al., 2021) with reconstruction and a k-Means objective. For comprehensive reviews, we refer to Alqahtani et al. (2021), Paparrizos et al. (2024), and Lafabregue et al. (2022). All of these methods return a single clustering, although, as shown in Figure 1, multiple non-redundant clusterings can provide more insights into the clustered time-series. We focus on univariate series to study this diversity within a single channel, leaving multivariate series, where channels provide an additional source of diversity, to future work (Section 6).

2.2 Non-redundant clustering

Non-redundant clustering, also called multiple or alternative clustering, seeks m high-quality, pairwise dissimilar partitions $C_1, \ldots, C_m$ of the same dataset, such as groupings by color, shape, or material (Miklautz et al., 2020). Similarity between partitions is commonly assessed using normalized mutual information or the adjusted Rand index (Bailey, 2014; Yu et al., 2024). Unlike multi-view clustering, which fuses several views into a single consensus clustering, and cluster ensembles (Strehl & Ghosh, 2002), which combine many clusterings into a single one, non-redundant clustering preserves distinct solutions. Meta-clustering (Caruana et al., 2006) instead groups candidate clusterings rather than directly optimizing their dissimilarity. Existing approaches discover alternatives in the original feature space, learn separate subspaces, or use user-specified aspects.

Original feature space. Early methods operate on the given features. Sequential approaches seek a high-quality partition dissimilar to a reference clustering, using constraints derived from that clustering (Bae & Bailey, 2006), information-theoretic criteria (Gondek & Hofmann, 2004), constrained optimization (Qi & Davidson, 2009), or nonnegative matrix factorization (Yang & Zhang, 2017). Simultaneous approaches learn multiple partitions through mutual-information penalties (Dang & Bailey, 2010) or separate centroid sets (Jain et al., 2008). Extensions address co-clustering (Wang et al., 2021), matrix-valued data (Tokuda et al., 2021), and multiple views or relations supporting distinct clusterings (Yao et al., 2019; Wei et al., 2020; 2021). Their reliance on predefined features and similarity measures can limit their suitability for raw time-series with temporal shifts and warping.

Subspaces and deep representations. Subspace methods seek complementary clusterings in distinct feature subspaces using orthogonal projections (Cui et al., 2007), spectral criteria (Niu et al., 2010; Hu et al., 2017), subspace-optimized k-Means (Mautz et al., 2020), or independence constraints (Wang et al., 2019). Deep approaches learn nonlinear representations: ENRC jointly optimizes autoencoding and clustering while softly assigning latent dimensions to clusterings (Miklautz et al., 2020); other methods use redundancy-controlled multi-head attention (Ren et al., 2022), disentanglement through variational EM (Yao & Hu, 2024), or multi-facet variational autoencoders (Falck et al., 2021). Further work selects parameters automatically via minimum description length (Leiber et al., 2022). Beyond subspace methods, DivClust controls clustering diversity through a dedicated loss (Metaxas et al., 2023). For time-series, disentanglement methods typically target predefined components, such as trend and seasonality (Woo et al., 2022; Wang et al., 2022).

User-guided approaches. Guided methods use external input to identify relevant clustering aspects. This input includes expert knowledge (Chang et al., 2017), aspect-specific augmentations in AugDMC (Yao et al., 2023), and user-tailored augmentations with clustering-level contrast in CMClusts (Zhang et al., 2025). Multi-modal methods align user-provided keywords with image content through

Table 1: Notation.

Notation	Description	Notation	Description
B	Batch size	E_θ	Encoder with parameters θ
T	Number of time steps	$Z \in \mathbb{R}^{B\times D}$	Encoder output
C	Number of input channels	$Z^f \in \mathbb{R}^{B\times d}$	Representation of facet f
F	Number of facets	$M \in \mathbb{R}^{B\times F\times d}$	Normalized facet masks
D	Total embedding dimension	$M_{:,f,:} \in \mathbb{R}^{B\times d}$	Mask for facet f
$d = D/F$	Embedding dimension per facet	$\widetilde{Z}^f \in \mathbb{R}^{B\times d}$	Masked representation of facet f
$X \in \mathbb{R}^{B\times T\times C}$	Input time-series batch	k_f	Number of clusters in facet f

vision-language models (Yao et al., 2024a;b), or use language models to propose subpopulations or clustering types (Luo et al., 2024; Zhang et al., 2026; Chen et al., 2025). These approaches emphasize externally specified or proposed aspects rather than discovering them solely from the data, and could complement non-redundant methods by helping users select relevant partitions. Their transfer to time-series poses challenges: keyword-based guidance requires text–signal alignment, while prior work reports limitations in language models' zero-shot reasoning about raw time-series (Merrill et al., 2024). Augmentation-based guidance requires prior knowledge of which transformations preserve each desired aspect, which may be unavailable when the relevant temporal patterns are unknown.

Our setting. To our knowledge, non-redundant clustering has not been studied for time-series. Existing non-redundant clustering methods have not been developed with a focus on time-series, and were evaluated primarily on tabular and image data. The aspects that separate different time-series clusterings are usually unknown in advance, so we follow a subspace approach and learn them from the data without user input. A single encoder is trained with a consistency clustering loss, and masked latent representations are routed to separate decoders, one per clustering. Because each decoder reconstructs the series from its own masked embedding, one can inspect which temporal patterns each clustering encodes. Section 3 describes this design in detail.

3 METHOD

NRTSC learns embeddings through a single encoder, then splits the embeddings into equal blocks. Each block is then passed to the gater, which selects features relevant for the different clusterings. The selected features are then passed through the corresponding facet, which decodes them into a relevant time-series component and performs the cluster assignment. A two-facet example of the architecture (finds only two non-redundant clusterings) is shown in Figure 2. The architecture can be extended to any number of clusterings by repeating the facet blocks as needed.

3.1 MODEL ARCHITECTURE

A convolutional encoder is used to learn a fixed-dimension latent representation. It consists of four convolutional layers with 32, 64, 128 and 128 output channels with kernel sizes of 7, 5, 5, 3, respectively. Each convolutional layer is followed by group normalization and a rectified linear unit (ReLU) activation. The first three blocks are followed by a max-pooling layer, while the last block is followed by an adaptive average pooling layer. Finally, a linear layer projects the resulting 128-dimensional vector into the embedding space, which is then split into equal-sized facet representations $Z = E_\theta(X) \in \mathbb{R}^{B\times D}, \qquad [Z^1; \ldots; Z^F] = Z.$

The gater learns sample-specific masks that control the contribution of each coordinate to its facet's latent representation. It first passes the facet representations through a multi-layer perceptron (MLP), then applies softmax across the facet dimension to obtain the masks $M \in \mathbb{R}^{B\times F\times d}$. Thus, for each sample b and within-block coordinate j, the masks satisfy $\sum_{f=1}^{F} M_{bfj} = 1$ where $b = 1, \ldots, B$ and $j = 1, \ldots, d$. Each mask weights the corresponding coordinate in that facet's latent block through element-wise multiplication: $\widetilde{Z}^f = Z^f \odot M_{:,f,:} \in \mathbb{R}^{B\times d}, \qquad f = 1, \ldots, F.$

Each facet receives its masked representation $\widetilde{Z}^f$, and processes it through a decoder and a clustering head. The decoder consists of two linear layers of sizes 128 and 256, each followed by a ReLU activation, and a linear output layer that reconstructs a time-series component $\widehat{X}^f \in \mathbb{R}^{B\times T\times C}$. The facets reconstructed components are then summed up to obtain the full reconstruction. In parallel to each decoder, there is a linear clustering head followed by a softmax activation that predicts clustering probabilities $Q^f \in \mathbb{R}^{B\times k_f}$ for each $\widetilde{Z}^f$ where $Q^f = \mathrm{Head}_f(\widetilde{Z}^f), \widehat{X}^f = \mathrm{Dec}_f\left(\widetilde{Z}^f\right), \widehat{X} = \sum_{f=1}^{F} \widehat{X}^f$.

3.2 Loss Function

The loss function combines three groups of losses: embedding preparation, clustering, and non-redundancy. These three losses encourage learning informative facet representations, reconstructing different time-series components, and finding multiple distinct clusterings.

The embedding preparation loss consists of three parts: the reconstruction loss, the mask sharpness loss, and the mask usage loss. *The reconstruction loss* measures the mean squared error between the input batch and the sum of the facet reconstructions $\widehat{X} = \sum_{f=1}^{F} \widehat{X}^f$. This serves as a non-degeneracy control for the gater Miklautz et al. (2020) by requiring the network to reconstruct the original input from the learned facet representations. Additionally, it encourages all facet representations to retain the information needed to reconstruct the time-series. Therefore, individual facets can construct different aspects of the signal. *The mask sharpness loss* minimizes the entropy of the mask distribution across facets. This encourages decisive gating for facet representation, i.e., it teaches the gater to select those dimensions/features in the embeddings that are important for the corresponding facet. This reduces diffuse masking and supports facet specialization. *The mask usage loss* encourages all facets to receive comparable average mask weights, preventing any facet mask from becoming inactive. This is achieved by minimizing the squared difference between each facet's average activation and the uniform target of $1/F$. For compact notation, we unfold the mask tensor $M \in \mathbb{R}^{B\times F\times d}$ into a matrix $\mathbf{A} \in \mathbb{R}^{(Bd)\times F}$, and $\mathbf{1}_n$ denotes the vector of n ones. Therefore, the total embedding preparation loss becomes:

$$\mathcal{L}_{\text{prep}} = \frac{1}{BTC}\left\|X - \widehat{X}\right\|_F^2 - \frac{1}{Bd}\,\mathrm{Tr}\left(\mathbf{A}^\top \log \mathbf{A}\right) + \frac{1}{F}\left\|\frac{1}{Bd}\mathbf{A}^\top \mathbf{1}_{Bd} - \frac{1}{F}\mathbf{1}_F\right\|^2 \tag{1}$$

The clustering loss uses a consistency loss across augmented latent views to learn clustering assignments and a balancing loss to prevent mode collapse. For each facet representation, NRTSC creates two augmented views by adding random noise in the encoder embedding space Z. The two views then use the same gater and the same facet-specific clustering head to produce two sets of predictions. *The consistency loss* uses cross-entropy to penalize differing predictions between the

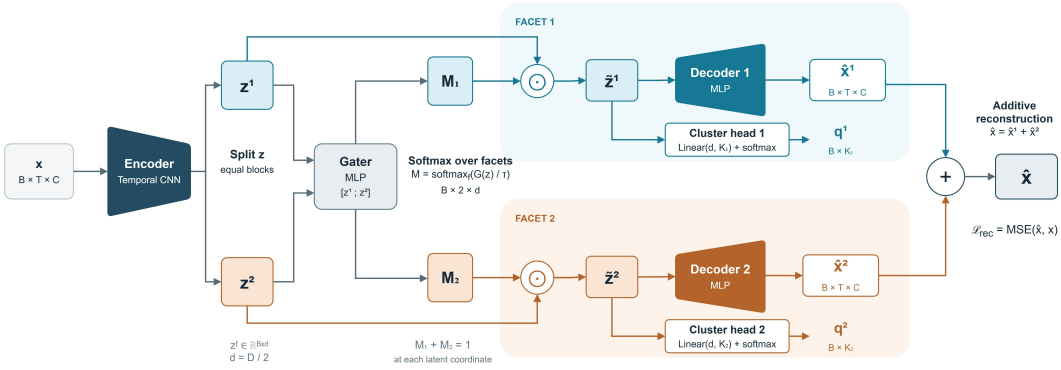


Figure 2: Method architecture for a dataset containing two facets. For more facets, the architecture is scaled correspondingly.

two augmented views for the same facet.

$$\mathcal{L}_{\text{cons}} = -\frac{1}{2BF}\sum_{f=1}^{F}\text{Tr}\left((Q^{f,(1)})^\top \log Q^{f,(2)} + (Q^{f,(2)})^\top \log Q^{f,(1)}\right) \tag{2}$$

In this context, augmentations in the embedding space are preferable to raw time-series augmentations. Standard augmentations (e.g., scaling and permutations) require domain-specific priors and can affect multiple signal properties. However, embedding-space augmentations directly affect semantic representations and, consequently, the generated views align with the facet-specific clustering criterion.

Using $\mathcal{L}_{\text{cons}}$ alone can lead to assigning all the data points to a single cluster. Therefore, a *cluster balance loss* encourages the objective function to minimize the negative entropy for each facet-specific clustering head Peng et al. (2024); Wang et al. (2026). Thus, the consistency term makes the predictions stable for each sample, and the balance term promotes diversity of assignments across the batch. Therefore, the mean probability distribution across the two views becomes $\bar{q}^f = \frac{1}{2B}\left(Q^{f,(1)} + Q^{f,(2)}\right)^\top \mathbf{1}_B$. Additionally, the balance loss term and the total cluster loss become:

$$\mathcal{L}_{\text{bal}} = \frac{1}{F}\sum_{f=1}^{F}(\bar{q}^f)^\top \log \bar{q}^f, \qquad \mathcal{L}_{\text{cluster}} = \mathcal{L}_{\text{cons}} + \mathcal{L}_{\text{bal}} \tag{3}$$

The non-redundancy loss penalizes overlap between facets at both the reconstruction and representation levels. *For the reconstruction level*, it minimizes *the squared cosine similarity* between the reconstructed time-series components $\{\widehat{X}^1, \ldots, \widehat{X}^F\}$. This encourages different facets to reconstruct distinct components in the signal space. Moreover, squaring the cosine similarity penalizes both positive and negative alignments, preventing identical components and opposing ones that can cancel each other out in the summed reconstruction.

Let $\widetilde{X}^f \in \mathbb{R}^{B\times TC}$ denote the row-normalized matrix of the flattened reconstructions for facet f, where its b-th row is defined as $\widetilde{X}^f_{b,*} = \frac{\widehat{x}^f_b}{\|\widehat{x}^f_b\|_2}$. Using this normalized matrix, the component decorrelation loss becomes:

$$\mathcal{L}_{\text{decorr}} = \frac{1}{BF(F-1)}\sum_{\substack{f,g=1\\ f\neq g}}^{F}\left\|(\widetilde{X}^f \odot \widetilde{X}^g)\mathbf{1}_{TC}\right\|^2 \tag{4}$$

For the representation level, the model minimizes *the Hilbert-Schmidt Independence Criterion (HSIC) loss* to prevent one facet representation from having the clustering information of another facet. This loss penalizes the statistical dependency between the clustering predictions of one facet view ($Q^{f,(v)} \in \mathbb{R}^{B\times K_f}$, $v \in \{1,2\}$) and the masked representation of another facet ($\widetilde{Z}^g$). Since there is a single encoder and a single gater, minimizing this dependency directly may reduce the loss by making predictions less informative, rather than learning non-redundant representation. To prevent this short-cut, we adopt an asymmetric gradient strategy inspired by Chen & He (2021). We detach the predictions $Q^{f,(v)}$ from the computational graph, such that it encourages having facet-representation independent from all other facets' predictions. We use the stop-gradient operator $\text{sg}(\cdot)$ to block differentiation through the prediction argument of each HSIC term. Forcing all the gradients to flow exclusively back through the other masked representation, the gater, and the encoder. Consequently, the HSIC loss does not directly update the clustering head parameters, but it forces the gater to adjust its masking strategy and the encoder to learn more suitable embeddings. Therefore, it mainly affects the facets representations, encouraging the model to select embeddings relevant only to a single clustering criterion for each facet.

Now, let $H = I_B - \frac{1}{B}\mathbf{1}_B\mathbf{1}_B^\top$ denote the batch-centering matrix. The implemented HSIC loss uses linear kernels for simplicity and can be expressed as the squared norm of the cross-covariance matrix:

$$\mathcal{L}_{\text{HSIC}} = \frac{1}{2F(F-1)}\sum_{v=1}^{2}\sum_{\substack{f,g=1\\ f\neq g}}^{F}\left\|\frac{1}{B-1}\,\text{sg}\left(Q^{f,(v)}\right)^\top H\widetilde{Z}^g\right\|^2 \tag{5}$$

Minimizing this loss encourages having distinct clustering criteria across facets. Although it does not guarantee full statistical independence, the reconstruction decorrelation term complements this objective by discouraging the reconstruction of similar components. Therefore, the non-redundancy loss becomes $\mathcal{L}_{\text{nonred}} = \mathcal{L}_{\text{decorr}} + \mathcal{L}_{\text{HSIC}}$. Overall, the total loss of our whole architecture is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{prep}} + \mathcal{L}_{\text{cluster}} + \mathcal{L}_{\text{nonred}} \tag{6}$$

3.3 Training Procedure

The training procedure has two phases: warm-up and joint training. During the warm-up phase, the model is optimized using the loss terms for reconstruction, mask sharpness, mask usage (Eq. 1), and decorrelation of the reconstructed components (Eq. 4). This forces the network to learn relevant semantic representations for each facet that also encode the necessary information to reconstruct the signal while creating distinct routing pathways within the shared gater. Therefore, the facets extract different and informative signal components before any clustering attempts. After the warm-up, the clustering heads are initialized with k-Means centers computed for each facet representation. This initialization only runs once and is not repeated every epoch. This makes the network ready for the joint training phase, which uses the full loss (Eq. 6). Finally, we optionally allow multiple restarts of the training process and define an unsupervised score to select a single model. To ensure non-redundancy, this score combines the average Silhouette score across all facet representations with inter-facet diversity, measured as the inverse of the Normalized Mutual Information (NMI) between predicted clusterings. Further details can be found in Appendix A.1.

4 Non-redundant Time-series Datasets

Evaluating non-redundant clusterings requires datasets with multiple ground-truth labelings. Existing benchmarks come mainly from images, where the same object can be labeled, for example, by color, shape, and material (Miklautz et al., 2020) or by color and species (Yao et al., 2024b). Time-series clustering, in contrast, is usually evaluated on the UCR archive (Dau et al., 2019), which provides only a single label set per dataset (Javed et al., 2020). Time-series with several label types do exist, but such data are distributed over several sources and have not been assembled for non-redundant clustering evaluation. We therefore collect these publicly available real-world time-series datasets and consolidate existing alternative labelings for the same samples, without modifying the underlying time-series. Moreover, we also introduce additional new ones.

New Datasets. We introduce three non-redundant time-series clustering datasets: BumpFreq, BME-AC, and Plane-AC. *The BumpFreq dataset* contains 1000 synthetic time-series of length 300 that contain two independently sampled characteristics. The first is determined by a positive or negative Gaussian bump, while the second determines its frequency via a sine wave with either high or low frequency. *The BME-AC dataset* contains 2000 time-series of length 200. It is inspired by the UCR BME dataset Dau et al. (2019) but designed for the non-redundant clustering task. The first

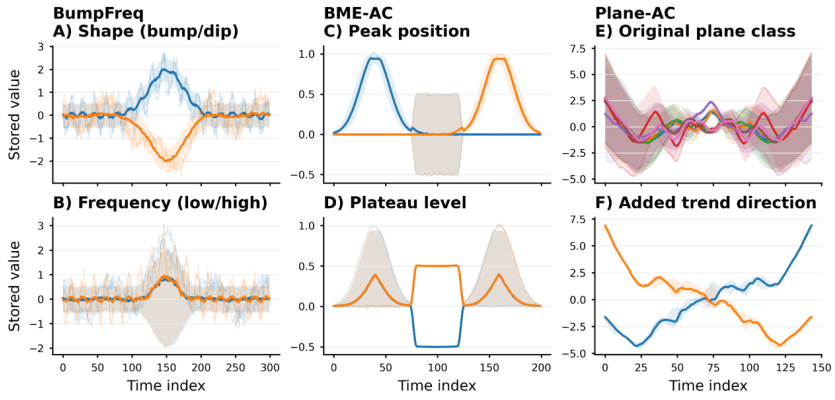


Figure 3: Examples of non-redundant time-series datasets.

label indicates whether a peak occurs at the beginning or end of the signal, while the second label determines the middle-part plateau level. *The Plane-AC dataset* is derived from the UCR Plane dataset Dau et al. (2019) whose time-series represent aircraft outlines with seven signal classes for different aircraft types. Each signal is duplicated to create upward- and downward-trending versions. The added linear trend is centered around zero and scaled according to the amplitude range of the individual time-series to preserve the original signal shape. The resulting dataset contains 416 samples, each of length 144. Figure 3 shows the centroids of the three introduced dataset clusterings.

Benchmark Datasets. We collect eight datasets from the UCR Archive Dau et al. (2019) and keep the original signals without injecting any artificial components. The non-redundant clusterings are obtained by combining labels from different tasks for the same signal observations or by selecting a subset of the dataset. The datasets cover human actions, subject identities, hand movements, chronological ages, and more. Therefore, these datasets complete the evaluation benchmark with real-world examples. Further details about each dataset can be found in Appendix A.2.

5 Experiments

In this section, we evaluate NRTSC by running an ablation study to quantify the importance of each loss term, then quantitatively compare performance against other clustering methods. Finally, we show the predictions and the learned facet representations of the BumpFreq dataset as an example.

Ablation Study. We first investigate the effect of each individual loss term by removing one loss term at a time and evaluating NRTSC performance over three different runs. Figure 4 (a) shows the NMI when all losses were used, and (b) shows the change in NMI after removing one loss term. Removing the reconstruction loss consistently degrades performance, which shows the importance of preserving the original time-series structure. However, removing the clustering and balance losses results in misaligned cluster prototypes or many unused clusters, which also degrades performance. The decorrelation and mask losses have smaller effects on NMI in the evaluated settings, suggesting that they become important for more difficult datasets, whereas for easier tasks, the other components compensate for their absence.

Evaluation. We compare NRTSC performance to existing non-redundant clustering methods after adapting them to the time-series case. For the deep-clustering methods (ENRC Miklautz et al. (2020), DivClust Metaxas et al. (2023), iMClusts Ren et al. (2022), and CMClusts Zhang et al. (2025)), we replace the encoder architecture with a temporal one that exactly matches the encoder used in NRTSC. For methods that require an autoencoder architecture, we use our encoder in addition to a 3-layer convolutional decoder. For the remaining methods (ISAAC Ye et al. (2016) and AutoNR Leiber et al. (2022)), we pretrain a temporal autoencoder using the MSE loss, then feed the learned embeddings to the clustering algorithm. Additionally, we compare to k-Shape Paparrizos & Gravano (2015), the pipeline of TS2Vec Yue et al. (2022) followed by k-Means, and DTCC Zhong et al. (2023) as another group of time-series clustering methods that return a single clustering. We train each algorithm with every distinct number of clusters from the available ground-truth labelings separately, using the output obtained with its corresponding number of clusters. For all comparison methods, we

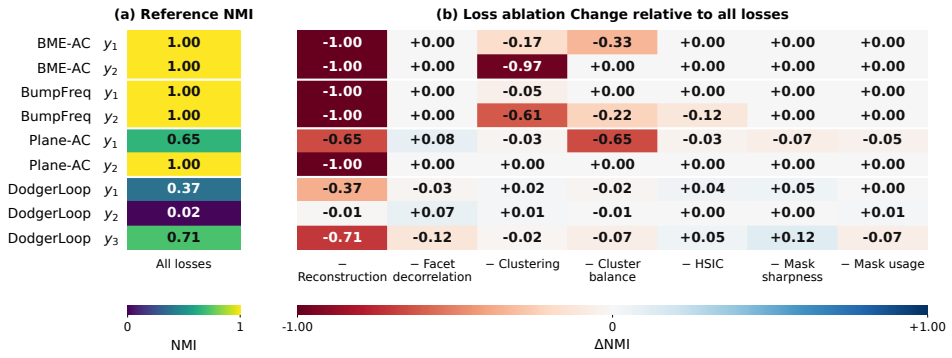


Figure 4: Ablation study showing the NMI drop in the absence of individual loss terms.

Table 2: Mean ± standard deviation of NMI scores across 5 runs. The highest mean in each row is bold and the second-highest is underlined. For average rank, lower is better. TS2+KM: TS2Vec+k-Means. Datasets' names, abbreviations, and labels' meaning are in Talbe 3

Data	Label	k_f	NRTSC	AutoNR	DivClust	ENRC	ISAAC	iMClusts	CMClusts	KShape	TS2+KM	DTCC
BME	y_1	2	**1.00 ± 0.00**	0.20 ± 0.45	0.06 ± 0.07	0.84 ± 0.33	0.18 ± 0.04	0.04 ± 0.08	0.00 ± 0.00	0.32 ± 0.06	0.00 ± 0.00	0.00 ± 0.01
	y_2	2	**1.00 ± 0.00**	0.80 ± 0.45	0.79 ± 0.29	**1.00 ± 0.00**	0.23 ± 0.06	**1.00 ± 0.00**	0.06 ± 0.09	0.32 ± 0.06	**1.00 ± 0.00**	0.93 ± 0.15
BF	y_1	2	**1.00 ± 0.00**	0.60 ± 0.55	0.80 ± 0.45	**1.00 ± 0.00**	0.33 ± 0.08	0.53 ± 0.43	0.00 ± 0.00	0.20 ± 0.45	0.00 ± 0.00	0.18 ± 0.06
	y_2	2	**1.00 ± 0.00**	0.40 ± 0.55	0.20 ± 0.45	**1.00 ± 0.00**	0.34 ± 0.05	0.08 ± 0.07	0.00 ± 0.00	0.80 ± 0.45	**1.00 ± 0.00**	0.00 ± 0.00
DP	y_1	3	**0.42 ± 0.01**	0.32 ± 0.02	0.27 ± 0.01	0.35 ± 0.06	0.37 ± 0.02	0.04 ± 0.05	0.01 ± 0.00	0.31 ± 0.02	0.38 ± 0.01	0.14 ± 0.13
	y_2	6	**0.62 ± 0.01**	0.51 ± 0.03	0.46 ± 0.01	0.54 ± 0.06	0.53 ± 0.04	0.08 ± 0.11	0.02 ± 0.01	0.50 ± 0.01	0.56 ± 0.01	0.49 ± 0.06
DL	y_1	7	0.37 ± 0.07	**0.44 ± 0.06**	0.32 ± 0.04	0.41 ± 0.05	0.03 ± 0.04	0.28 ± 0.13	0.19 ± 0.10	0.40 ± 0.02	0.30 ± 0.03	0.33 ± 0.15
	y_2	2	0.02 ± 0.01	0.27 ± 0.03	0.17 ± 0.07	**0.32 ± 0.05**	0.01 ± 0.02	0.19 ± 0.11	0.04 ± 0.03	0.01 ± 0.00	0.19 ± 0.03	0.04 ± 0.03
	y_3	2	0.71 ± 0.18	0.63 ± 0.13	0.29 ± 0.05	0.55 ± 0.20	0.01 ± 0.02	0.29 ± 0.32	0.31 ± 0.19	**0.84 ± 0.00**	0.26 ± 0.03	0.16 ± 0.18
GP	y_1	2	0.07 ± 0.08	0.01 ± 0.01	0.02 ± 0.01	**0.22 ± 0.10**	0.18 ± 0.04	0.05 ± 0.08	0.00 ± 0.00	0.05 ± 0.01	0.03 ± 0.01	0.05 ± 0.03
	y_2	2	0.19 ± 0.11	0.38 ± 0.00	0.21 ± 0.41	0.38 ± 0.09	0.19 ± 0.02	0.14 ± 0.10	0.01 ± 0.00	0.59 ± 0.02	**0.64 ± 0.03**	0.41 ± 0.24
	y_3	2	0.79 ± 0.23	0.74 ± 0.26	0.01 ± 0.02	**0.91 ± 0.19**	0.20 ± 0.04	0.12 ± 0.08	0.01 ± 0.00	0.03 ± 0.00	0.02 ± 0.00	0.07 ± 0.10
MP	y_1	3	0.39 ± 0.01	0.35 ± 0.05	0.27 ± 0.01	0.37 ± 0.03	0.37 ± 0.02	0.02 ± 0.01	0.01 ± 0.00	**0.40 ± 0.00**	**0.40 ± 0.00**	0.32 ± 0.01
	y_2	6	**0.49 ± 0.01**	0.44 ± 0.05	0.36 ± 0.01	0.46 ± 0.03	0.43 ± 0.03	0.02 ± 0.01	0.02 ± 0.00	0.42 ± 0.01	0.42 ± 0.01	0.35 ± 0.04
PLA	y_1	7	0.62 ± 0.04	0.59 ± 0.01	0.55 ± 0.05	**0.71 ± 0.20**	0.21 ± 0.03	0.35 ± 0.08	0.01 ± 0.00	0.51 ± 0.08	0.55 ± 0.01	0.41 ± 0.07
	y_2	2	**1.00 ± 0.00**	0.54 ± 0.01	0.45 ± 0.01	0.74 ± 0.24	0.17 ± 0.04	0.81 ± 0.24	0.11 ± 0.04	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
PP	y_1	3	**0.54 ± 0.01**	0.42 ± 0.02	0.35 ± 0.03	0.45 ± 0.02	0.40 ± 0.12	0.01 ± 0.01	0.01 ± 0.00	0.45 ± 0.01	**0.54 ± 0.00**	0.45 ± 0.07
	y_2	6	**0.62 ± 0.02**	0.53 ± 0.01	0.45 ± 0.03	0.57 ± 0.02	0.45 ± 0.14	0.02 ± 0.00	0.02 ± 0.01	0.49 ± 0.03	0.52 ± 0.01	0.52 ± 0.03
SH	y_1	2	**0.12 ± 0.03**	0.11 ± 0.02	0.08 ± 0.02	0.11 ± 0.02	0.11 ± 0.02	0.05 ± 0.03	0.01 ± 0.00	0.06 ± 0.00	0.06 ± 0.01	0.00 ± 0.00
	y_2	6	**0.29 ± 0.03**	0.26 ± 0.05	0.16 ± 0.02	0.23 ± 0.02	0.16 ± 0.02	0.19 ± 0.05	0.01 ± 0.00	0.22 ± 0.01	0.17 ± 0.01	0.00 ± 0.00
	y_3	5	0.26 ± 0.05	0.27 ± 0.06	**0.30 ± 0.05**	0.21 ± 0.03	0.16 ± 0.01	0.20 ± 0.04	0.01 ± 0.00	0.26 ± 0.07	0.23 ± 0.01	0.00 ± 0.00
SC	y_1	2	0.09 ± 0.09	0.00 ± 0.00	0.01 ± 0.02	0.06 ± 0.10	0.00 ± 0.00	**0.16 ± 0.13**	0.00 ± 0.01	0.00 ± 0.00	0.00 ± 0.00	0.01 ± 0.01
	y_2	2	**1.00 ± 0.00**	0.99 ± 0.01	**1.00 ± 0.00**	0.64 ± 0.49	0.01 ± 0.01	0.52 ± 0.35	0.06 ± 0.02	0.83 ± 0.37	**1.00 ± 0.00**	0.81 ± 0.11
TP	y_1	2	0.05 ± 0.12	0.00 ± 0.00	0.30 ± 0.20	0.00 ± 0.00	0.30 ± 0.28	0.01 ± 0.01	0.00 ± 0.00	0.21 ± 0.03	**0.32 ± 0.06**	0.00 ± 0.00
	y_2	2	0.09 ± 0.05	0.00 ± 0.00	0.09 ± 0.07	0.01 ± 0.01	**0.43 ± 0.16**	0.02 ± 0.03	0.00 ± 0.00	0.17 ± 0.02	0.25 ± 0.07	0.00 ± 0.00
Average Rank			**2.66**	4.68	5.84	3.24	6.36	6.74	8.96	4.90	4.56	6.66

use the default training parameters suggested by the original authors and fix the number of epochs for all algorithms. Additionally, we use the original implementation provided by the authors, but for ENRC Miklautz et al. (2020) and AutoNR Leiber et al. (2022), we use the implementation provided by the ClustPy package Leiber et al. (2023). The implementation of NRTSC is available in `https://anonymous.4open.science/r/NRTSC-E9AD/`

We set the training epochs to 600 epochs: 300 epochs for the warm-up phase and 300 epochs for the joint training phase; batch size to 64, learning rate to 0.001, number of restarts to 10, and embedding dimension to 32 × the number of non-redundant clusterings. We fix these parameters for all datasets. For each dataset, we run each algorithm 5 times using 5 random seeds and calculate the NMI score for each run. The mean and standard deviation for the runs are reported in Table 2, and a summary of the names of the datasets, the abbreviations used, and the meaning of each label is provided in Table 3. We calculate the average rank across all clusterings for all datasets; a lower value means a higher rank, and NRTSC has the best rank of 2.66. The adjusted rand index scores are also calculated and reported in Appendix A.3 Table 4, they show similar performance. Additionally, we calculate the average NMI between the predicted clusterings for all non-redundant clustering algorithms, and NRTSC performs well in finding diverse predictions. Further details and results are provided in Appendix A.3 and Table 5.

Decomposition. NRTSC shows the ability to perform signal decomposition. Figure 1 shows the isolation of independent patterns; similarly, the learned facet representation in Figure 8 shows that the selected features for facet-1 are turned off for facet-2 and vice versa. This is reflected in the facet embeddings (Figure 7), where clustering is detectable only in a single facet representation, although it is difficult to identify in the original embeddings.

6 Conclusion

In this paper, we introduced NRTSC for non-redundant time-series clustering. NRTSC employs multiple facets, each responsible for a single clustering. A facet has a clustering layer for assigning clusters and a decoder that reconstructs a part of the signal that provides insight into the underlying clustering criterion. We also collect and introduce a set of univariate, non-redundant time-series datasets that serve as a benchmark for future non-redundant time-series clustering algorithms. A future direction is to collect multivariate datasets and test how the algorithms perform in this setting.

AI USE STATEMENT

In this work, we used generative AI tools mainly for rephrasing, sentence refinement, grammar checking, and writing complex LaTeX syntax. We have reviewed all AI-assisted work. We have NOT used generative AI tools for creating images, creating datasets, coding, writing proofs, designing methods, or providing feedback on the research, and there are no other disclosures applicable to this work. We take responsibility for the final content of this work, including anything produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Our code is included in the submission, Section 5 - Evaluation: `https://anonymous.4open.science/r/NRTSC-E9AD/`, and will be made publicly available upon acceptance. We refer to Section 3 and Appendix A.1 for the method and training details, Section 4 and Appendix A.2 for dataset construction, and Section 5 and Appendix A.3 for evaluation.

REFERENCES

Ali Alqahtani, Mohamed Ali, and Xunxian Xiang. Deep time-series clustering: A review. *Electronics*, 10(23), December 2021. doi: 10.3390/electronics10233001.

Eunjin Bae and James Bailey. Coala: A novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM)*, pp. 53–62. IEEE, December 2006. doi: 10.1109/ICDM.2006.37.

James Bailey. Alternative clustering analysis: A review. In Charu C. Aggarwal and Chandan K. Reddy (eds.), *Data Clustering: Algorithms and Applications*, pp. 533–548. CRC Press, 2014.

Inci M. Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K. Jain, and Jiayu Zhou. Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pp. 65–74, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 978-1-4503-4887-4. doi: 10.1145/3097983.3097997.

Donald J. Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *Workshop on Knowledge Discovery in Databases, AAAI-94*, pp. 359–370, 1994.

Rich Caruana, Mohamed Elhawary, Nam Nguyen, and Casey Smith. Meta clustering. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM)*, pp. 107–118, 2006.

Yale Chang, Junxiang Chen, Michael H. Cho, Peter J. Castaldi, Edwin K. Silverman, and Jennifer G. Dy. Multiple clustering views from multiple uncertain experts. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 674–683. PMLR, 2017. URL `https://proceedings.mlr.press/v70/chang17a.html`.

Irene Y. Chen, Rahul G. Krishnan, and David Sontag. Clustering interval-censored time-series for disease phenotyping. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6211–6221, 2022.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15745–15753, 2021. doi: 10.1109/CVPR46437.2021.01549.

Ziye Chen, Yiqun Duan, Riheng Zhu, Zhenbang Sun, and Mingming Gong. Agent-centric personalized multiple clustering with multi-modal LLMs, 2025.

Ying Cui, Xiaoli Zhang Fern, and Jennifer G. Dy. Non-redundant multi-view clustering via orthogonalization. In *Proceedings of the 2007 IEEE International Conference on Data Mining (ICDM)*, pp. 133–142. IEEE, 2007.

Xuan-Hong Dang and James Bailey. Generation of alternative clusterings using the CAMI approach. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pp. 118–129. Society for Industrial and Applied Mathematics, 2010. doi: 10.1137/1.9781611972801.11.

Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

José G. Dias, Jeroen K. Vermunt, and Sofia Ramos. Clustering financial time series: New insights from an extended hidden Markov model. *European Journal of Operational Research*, 243(3): 852–864, 2015. doi: 10.1016/j.ejor.2014.12.041.

Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee-Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, pp. 2352–2359, 8 2021. doi: 10.24963/ijcai.2021/324.

Fabian Falck, Haoting Zhang, Matthew Willetts, George Nicholson, Christopher Yau, and Chris C. Holmes. Multi-facet clustering variational autoencoders. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

David Gondek and Thomas Hofmann. Non-redundant data clustering. In *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM)*, pp. 75–82. IEEE, 2004.

Illia Horenko. On clustering of non-stationary meteorological time series. *Dynamics of Atmospheres and Oceans*, 49(2-3):164–187, 2010. doi: 10.1016/j.dynatmoce.2009.04.003.

Juhua Hu, Qi Qian, Jian Pei, Rong Jin, and Shenghuo Zhu. Finding multiple stable clusterings. *Knowledge and Information Systems*, 51:991–1021, 2017.

Prateek Jain, Raghu Meka, and Inderjit S. Dhillon. Simultaneous unsupervised learning of disparate clusterings. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 1(3):195–210, 2008. doi: 10.1002/sam.10007.

Ali Javed, Byung Suk Lee, and Donna M. Rizzo. A benchmark study on time series clustering. *Machine Learning with Applications*, 1:100001, 2020. doi: 10.1016/j.mlwa.2020.100001.

Baptiste Lafabregue, Jonathan Weber, Pierre Gaçarski, and Germain Forestier. End-to-end deep representation learning for time series clustering: A comparative study. *Data Mining and Knowledge Discovery*, 36:29–81, 2022. doi: 10.1007/s10618-021-00796-y.

Collin Leiber, Dominik Mautz, Claudia Plant, and Christian Böhm. Automatic parameter selection for non-redundant clustering. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, pp. 226–234. SIAM, 2022. doi: 10.1137/1.9781611977172.26.

Collin Leiber, Lukas Miklautz, Claudia Plant, and Christian Böhm. Benchmarking deep clustering algorithms with clustpy. In *2023 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 625–632. IEEE, 2023. doi: 10.1109/ICDMW60847.2023.00087.

Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. Contrastive clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8547–8555, 2021.

Yulin Luo, Ruichuan An, Bocheng Zou, Yiming Tang, Jiaming Liu, and Shanghang Zhang. Llm as dataset analyst: Subpopulation structure discovery with large language model. In *Computer Vision – ECCV 2024*, volume 15089 of *Lecture Notes in Computer Science*, pp. 235–252. Springer, 2024. doi: 10.1007/978-3-031-73414-4_14.

Qianli Ma, Jiawei Zheng, Sen Li, and Gary W. Cottrell. Learning representations for time series clustering. In *Advances in Neural Information Processing Systems*, volume 32, pp. 3781–3791, 2019.

Dominik Mautz, Wei Ye, Claudia Plant, and Christian Böhm. Non-redundant subspace clusterings with nr-kmeans and nr-dipmeans. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14(5):1–24, 2020.

Mike A. Merrill, Mingtian Tan, Vinayak Gupta, Thomas Hartvigsen, and Tim Althoff. Language models still struggle to zero-shot reason about time series. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 3512–3533, 2024.

Ioannis Maniadis Metaxas, Georgios Tzimiropoulos, and Ioannis Patras. Divclust: Controlling diversity in deep clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3418–3428, 2023.

Lukas Miklautz, Dominik Mautz, Muzaffer Can Altinigneli, Christian Böhm, and Claudia Plant. Deep embedded non-redundant clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5174–5181, April 2020. doi: 10.1609/aaai.v34i04.5961.

Donglin Niu, Jennifer G. Dy, and Michael I. Jordan. Multiple non-redundant spectral clustering views. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pp. 831–838, 2010.

John Paparrizos and Luis Gravano. k-Shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pp. 1855–1870. ACM, 2015.

John Paparrizos, Fan Yang, and Haojun Li. Bridging the gap: A decade review of time-series clustering methods. *arXiv preprint – arXiv:2412.20582*, December 2024. doi: 10.48550/arXiv.2412.20582.

Furong Peng, Jiachen Luo, Xuan Lu, Sheng Wang, and Feijiang Li. Cross-domain contrastive learning for time series clustering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38:8921–8929, 03 2024. doi: 10.1609/aaai.v38i8.28740.

François Petitjean, Alain Ketterlin, and Pierre Gaçarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, 44(3):678–693, 2011.

Zijie Qi and Ian Davidson. A principled and flexible framework for finding alternative clusterings. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 717–726. ACM, 2009.

Liangrui Ren, Guoxian Yu, Jun Wang, Lei Liu, Carlotta Domeniconi, and Xiangliang Zhang. A diversified attention model for interpretable multiple clusterings. *IEEE Transactions on Knowledge and Data Engineering*, 35(9):8852–8864, November 2022. doi: 10.1109/TKDE.2022.3218693.

Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

Alexander Strehl and Joydeep Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

Tomoki Tokuda, Okito Yamashita, and Junichiro Yoshimoto. Multiple clustering for identifying subject clusters and brain sub-networks using functional connectivity matrices without vectorization. *Neural Networks*, 142:269–287, October 2021. doi: 10.1016/j.neunet.2021.05.016.

Patrick Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Dieter Kreiseler, Fatima I. Lunze, Wojciech Samek, and Tobias Schaeffter. PTB-XL, a large publicly available electrocardiography dataset. *Scientific Data*, 7:154, 2020.

Congyu Wang, Mingjing Du, and Xiang Jiang. Time-frequency augmented multi-level contrastive clustering for time series. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(31): 26142–26150, Mar. 2026. doi: 10.1609/aaai.v40i31.39817. URL `https://ojs.aaai.org/index.php/AAAI/article/view/39817`.

Jie Wang, Xiao Wang, Guoxian Yu, Carlotta Domeniconi, Zhengwen Yu, and Zili Zhang. Discovering multiple co-clusterings with matrix factorization. *IEEE Transactions on Cybernetics*, 51:3576–3587, 2021. doi: 10.1109/TCYB.2019.2950568.

Xianchao Wang, Jialu Wang, Carlotta Domeniconi, Guoxian Yu, Guojiang Xiao, and Maozu Guo. Multiple independent subspace clusterings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5353–5360, 2019.

Zhiyuan Wang, Xovee Xu, Weifeng Zhang, Goce Trajcevski, Ting Zhong, and Fan Zhou. LaST: Learning latent seasonal-trend representations for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Shixin Wei, Jun Wang, Guoxian Yu, Carlotta Domeniconi, and Xiangliang Zhang. Multi-view multiple clusterings using deep matrix factorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 6348–6355, 2020.

Shixin Wei, Guoxian Yu, Jun Wang, Carlotta Domeniconi, and Xiangliang Zhang. Multiple clusterings of heterogeneous information networks. *Machine Learning*, 110(6):1505–1526, 2021.

Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. CoST: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. In *International Conference on Learning Representations (ICLR)*, 2022.

Sen Yang and Lijun Zhang. Non-redundant multiple clustering by nonnegative matrix factorization. *Machine Learning*, 106(5):695–712, 2017. doi: 10.1007/s10994-016-5601-9.

Jiawei Yao and Juhua Hu. Dual-disentangled deep multiple clustering. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pp. 679–687, 2024. doi: 10.1137/1.9781611978032.78.

Jiawei Yao, Enbei Liu, Maham Rashid, and Juhua Hu. AugDMC: Data augmentation guided deep multiple clustering. *Procedia Computer Science*, 222:571–580, 2023. doi: 10.1016/j.procs.2023.08.195.

Jiawei Yao, Qi Qian, and Juhua Hu. Multi-modal proxy learning towards personalized visual multiple clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14066–14075, June 2024a.

Jiawei Yao, Qi Qian, and Juhua Hu. Customized multiple clustering via multi-modal subspace proxy learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 82705–82725, 2024b.

Shixin Yao, Guoxian Yu, Jun Wang, Carlotta Domeniconi, and Xiangliang Zhang. Multi-view multiple clustering. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 4121–4127, 2019. doi: 10.24963/ijcai.2019/572.

Wei Ye, Samuel Maurus, Nina Hubig, and Claudia Plant. Generalized independent subspace clustering. In *Proceedings of the 2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 569–578. IEEE, dec 2016. ISBN 978-1-5090-5473-2. doi: 10.1109/ICDM.2016.0068.

Guoxian Yu, Liang Ren, Jun Wang, Carlotta Domeniconi, and Xiangliang Zhang. Multiple clusterings: Recent advances and perspectives. *Computer Science Review*, 52:16, May 2024. doi: 10.1016/j.cosrev.2024.100621.

Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. TS2Vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8980–8987, 2022. doi: 10.1609/aaai.v36i8.20881.

Shan Zhang, Liangrui Ren, Jun Wang, Yanyu Xu, Carlotta Domeniconi, and Guoxian Yu. Aligning contrastive multiple clusterings with user interests. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2025*, pp. 7011–7019, 2025. doi: 10.24963/ijcai.2025/780.

Shan Zhang, Liangrui Ren, Qiaoyu Tan, Carlotta Domeniconi, Wei Du, Jun Wang, and Guoxian Yu. Mllm enriched explainable multiple clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 28373–28381. Association for the Advancement of Artificial Intelligence, March 2026. doi: 10.1609/aaai.v40i33.40066.

Ying Zhong, Dong Huang, and Chang-Dong Wang. Deep Temporal Contrastive Clustering. *Neural Processing Letters*, May 2023. doi: 10.1007/s11063-023-11287-0.

A Appendix

A.1 Implementation Details

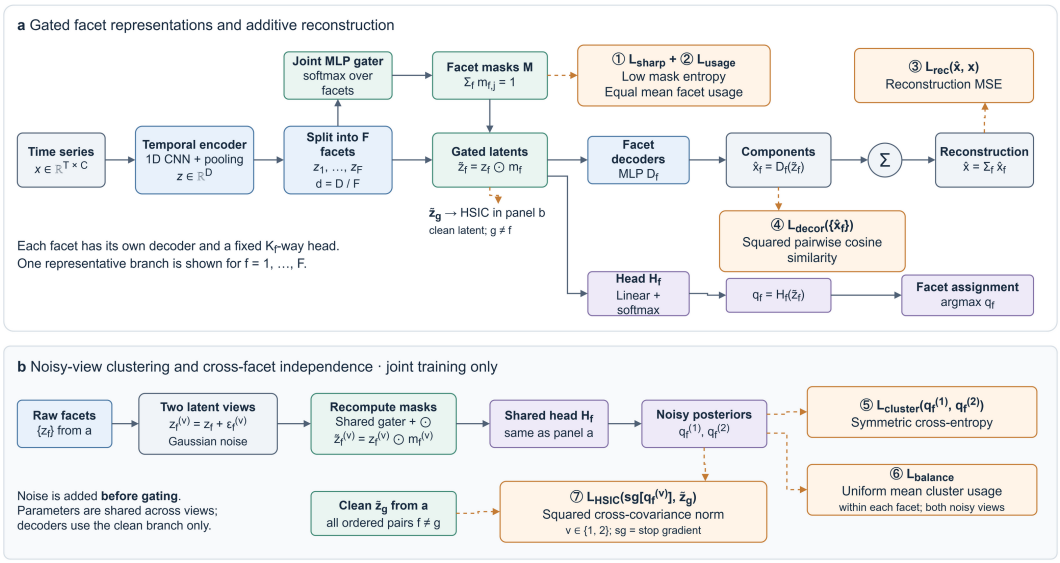


Figure 5: Overview of the method architecture and loss computation.

In this section, we provide further implementation details for NRTSC. Figure 5 also provides a detailed overview.

Importance of warm-up phase. During the warm-up epochs, training uses the embedding preparation loss Eq 1 and the component decorrelation loss:

$$\mathcal{L}_{\text{warm}} = \mathcal{L}_{\text{prep}} + \mathcal{L}_{\text{decorr}}. \tag{7}$$

The reconstruction term encourages the facet representations to preserve the input information, while the mask losses encourage selective and balanced gating. Component decorrelation is also active in this phase, encouraging the decoders to reconstruct different signal components. The clustering consistency, cluster balance, and HSIC losses are inactive, and the clustering heads receive no gradient updates. Consequently, the initial representations are learned without relying on predictions from randomly initialized clustering heads. Additionally, the clustering heads are initialized using k-Means after this warm-up. These two aspects together help to overcome the cold-start problem.

Clustering-head initialization. After the warm-up phase, the model computes the facet representations of all training samples. Then, k-Means is applied to each facet representation using its predefined number of clusters k_f. The resulting cluster centers are then used to initialize the rows of the corresponding clustering head. This provides a data-driven starting point for the clustering heads instead of a random initialization. It is also known that training using the reconstruction loss alone can make the model learn an embedding suitable for clustering Miklautz et al. (2020), this is also confirmed by the relatively good performance of the non-deep clustering methods (Section 5), where they can still learn good non-redundant clusterings from the embeddings learned by MSE and a convolutional autoencoder. Therefore, the warm-up and clustering-head initialization provide a reliable base for the upcoming joint optimization. A summary of this information is presented in Figure 5 (a).

Joint optimization. Following initialization, the encoder, gater, decoders, and clustering heads are optimized together using the complete objective (Eq.6). For each batch, the facet representations are used to compute the reconstruction, the mask losses, and the component decorrelation loss. For the clustering step, NRTSC creates two additional latent views by adding independent Gaussian noise:

$$Z = Z + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2) \tag{8}$$

The gater is applied to each augmented view, and the resulting masked representations are passed through each facet clustering head; the predictions are then used for the consistency and balance losses (Eq 3). The HSIC loss compares each noisy prediction matrix with the clean masked representations of the other facets. Finally, all losses are combined into a single objective function. A summary for this second phase is presented in Figure 5 (b).

Optimization details. The gating temperature decreases linearly across the entire training run, including both warm-up and joint training phases for more stable training. For epoch $e \in \{0, \dots, E-1\}$, it is given by:

$$\tau_e = \tau_{\text{start}} + \frac{e}{\max(E-1, 1)} \left(\tau_{\text{end}} - \tau_{\text{start}}\right). \tag{9}$$

A higher initial temperature speeds up facet training at the beginning, while a lower temperature encourages more stable masks and predictions. The schedule remains the same from the start of warm-up to the end of training.

Training uses Adam with a lower learning rate for the gater and a relatively higher learning rate for the autoencoder and the clustering heads. The lower learning rate makes the changes to the masks slower while the representations and decoders are being trained. Both learning rates follow cosine annealing over the full range of the number of epochs, and gradient norm clipping limits large parameter updates. The optimizer and learning-rate scheduler are retained at the transition between phases.

Default configuration. The implementation defaults to 600 epochs: 300 warm-up epochs and 300 joint-training epochs, with a batch size of 64. The initial learning rates are 10^{-3} for the autoencoder and the clustering heads, and 3×10^{-4} for the gater. We clip the gradient norm at 5 for more stable gradients, and set the latent noise standard deviation to $\sigma = 0.03$. The gating temperature decreases from 2.0 to 0.7. k-Means at the clustering head initialization step uses 10 initializations and a maximum of 300 iterations.

Unsupervised evaluation score. Training can be repeated with different random seeds. Each restart follows the complete schedule independently, and the final model is selected using an unsupervised score that combines the mean silhouette score across facets with the mean inter-facet diversity, inverse NMI, measured by $1 - NMI(predicted\ clusterings)$. The diversity contribution has a default weight of 0.5. We want to emphasize that ground-truth labels are not used for training or restart selection.

A.2 Datasets

In this section, we explain in detail the different datasets for the non-redundant time-series benchmark. In the following paragraphs, we describe each dataset and the meaning of each grouping according to the UCR archive Dau et al. (2019). We also compute the NMI scores between the different grouping ground-truth labels; all pairs are listed in Figure 6.

DistalPhalanx contains 539 bone-outline series of length 80, extracted from images of the distal bone of the middle finger. Identical observations from the UCR DistalPhalanxOutlineAgeGroup and DistalPhalanxTW Dau et al. (2019) tasks were matched exactly. The first annotation assigns subjects to three chronological age groups—0–6, 7–12, and 13–19 years—while the second records six observed Tanner–Whitehouse skeletal-maturity categories. Because chronological age and skeletal development are related, the two partitions are moderately associated rather than independent (Figure 6).

MiddlePhalanx is constructed similarly from 553 length-80 outlines of the middle phalanx. Exact signal matching combines the three-class chronological age-group annotation with the six-class Tanner–Whitehouse maturity annotation. The two labels describe distinct but biologically related aspects of development and consequently exhibit moderate agreement (Figure 6).

ProximalPhalanx contains 605 length-80 outlines of the proximal phalanx. The same observations from the age-group and Tanner–Whitehouse UCR tasks were aligned exactly, yielding a three-class chronological-age partition and a six-class skeletal-maturity partition. This dataset has the strongest association among the three phalanx pairs, reflecting the relationship between age and bone development while still preserving two non-equivalent classifications (Figure 6).

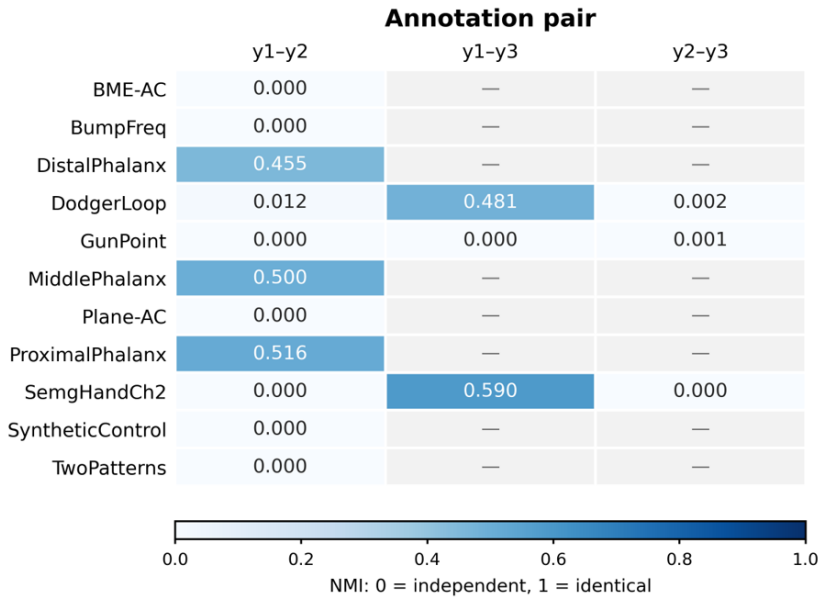


Figure 6: NMI score between true labels of different clusterings per dataset.

DodgerLoop comprises 144 complete daily traffic profiles of length 288, recorded by a freeway-ramp loop sensor near Dodger Stadium. The same profiles were matched across three UCR tasks Dau et al. (2019), and 14 observations containing missing values were excluded without imputation. The resulting annotations describe the day of the week, whether the day was a normal or game day, and whether it was a weekday or weekend. Day of the week and weekend status are naturally related, whereas game status is nearly independent of the other two labels (Figure 6).

GunPoint contains 451 motion series of length 150, obtained from actors' horizontal hand positions during five-second pointing actions. Exact matches across three UCR variants provide three binary annotations: holding a gun versus pointing with a finger, female versus male actor, and the earlier versus later recording era. The construction changes neither the observations nor their labels; it simply reunites the three documented descriptions of the same recordings.

SemgHandCh2 contains 900 channel-two surface-electromyography power spectra of length 1,500. The same measurements were aligned across three UCR classification tasks to obtain annotations for gender, six hand-grasp movements—cylindrical, hook, tip, palmar, spherical, and lateral—and five individual subjects. Movement is independent of both gender and subject in the balanced collection, while gender and subject identity are related because each subject belongs to one gender (Figure 6).

SyntheticControl is formed from the increasing-trend, decreasing-trend, upward-shift, and downward-shift classes of the original Synthetic Control Chart dataset, which contains 600 series. However, we exclude the normal and cyclic classes because they lack the required directional factor, which we use as a non-redundant clustering criterion. The remaining 400 series of length 60 are relabeled along two binary dimensions: trend versus level shift and upward versus downward direction. This factorization produces two balanced, independent annotations without modifying the generated signals.

TwoPatterns contains 5,000 simulated series of length 128, each containing two successive directional patterns. Its four original classes—down–down, up–down, down–up, and up–up—are decomposed into two binary annotations describing the direction of the first and second patterns separately. This converts the documented factorial class definition into two nearly independent clustering perspectives while leaving the original series unchanged.

Table 3: Dataset names, abbreviations, and clustering label meanings.

Dataset	Short	Label	Meaning	Dataset	Short	Label	Meaning
BME-AC	BME	y_1	Peak position	Plane-AC	PLA	y_1	Aircraft
		y_2	Plateau level			y_2	Trend
BumpFreq	BF	y_1	Direction	ProximalPhalanx	PP	y_1	Age group
		y_2	Frequency			y_2	TW
DistalPhalanx	DP	y_1	Age group	SemgHandCh2	SH	y_1	Gender
		y_2	TW			y_2	Movement
DodgerLoop	DL	y_1	Day of week			y_3	Subject
		y_2	Game day	SyntheticControl	SC	y_1	Shape type
		y_3	Weekend			y_2	Direction
GunPoint	GP	y_1	Action	TwoPatterns	TP	y_1	First direction
		y_2	Actor			y_2	Second direction
		y_3	Recording era				
MiddlePhalanx	MP	y_1	Age group				
		y_2	TW				

Table 4: Mean ± standard deviation of Adjusted Rand Index (ARI) scores across 5 runs. The highest mean in each row is bold and the second-highest is underlined. For average rank, lower is better. TS2+KM: TS2Vec+k-Means. Datasets' names, abbreviations, and labels' meaning are in Table 3.

Data	Label	k_f	NRTSC	AutoNR	DivClust	ENRC	ISAAC	iMClusts	CMClusts	KShape	TS2+KM	DTCC
BME	y_1	2	**1.00 ± 0.00**	0.20 ± 0.45	0.07 ± 0.08	0.81 ± 0.38	0.11 ± 0.09	0.05 ± 0.09	0.00 ± 0.00	0.24 ± 0.02	−0.00 ± 0.00	0.01 ± 0.01
	y_2	2	**1.00 ± 0.00**	0.80 ± 0.45	0.78 ± 0.30	**1.00 ± 0.00**	0.21 ± 0.13	**1.00 ± 0.00**	0.07 ± 0.12	0.24 ± 0.02	**1.00 ± 0.00**	0.94 ± 0.14
BF	y_1	2	**1.00 ± 0.00**	0.60 ± 0.55	0.80 ± 0.45	**1.00 ± 0.00**	0.27 ± 0.12	0.56 ± 0.43	0.00 ± 0.00	0.20 ± 0.45	−0.00 ± 0.00	0.13 ± 0.03
	y_2	2	**1.00 ± 0.00**	0.40 ± 0.55	0.20 ± 0.45	**1.00 ± 0.00**	0.23 ± 0.08	0.08 ± 0.07	0.00 ± 0.00	0.80 ± 0.45	**1.00 ± 0.00**	0.00 ± 0.00
DP	y_1	3	**0.46 ± 0.02**	0.33 ± 0.11	0.16 ± 0.01	0.32 ± 0.14	0.37 ± 0.02	0.03 ± 0.05	0.01 ± 0.01	0.33 ± 0.12	0.45 ± 0.00	0.16 ± 0.18
	y_2	6	0.69 ± 0.05	0.55 ± 0.17	0.30 ± 0.01	0.50 ± 0.19	0.54 ± 0.03	0.06 ± 0.10	0.01 ± 0.00	0.36 ± 0.04	**0.74 ± 0.01**	0.65 ± 0.22
DL	y_1	7	0.16 ± 0.05	**0.22 ± 0.07**	0.20 ± 0.04	0.20 ± 0.04	0.00 ± 0.01	0.10 ± 0.06	0.09 ± 0.06	0.20 ± 0.03	0.12 ± 0.03	0.17 ± 0.12
	y_2	2	0.01 ± 0.01	**0.26 ± 0.04**	0.10 ± 0.04	0.23 ± 0.08	0.01 ± 0.01	0.15 ± 0.13	0.02 ± 0.01	0.00 ± 0.00	0.14 ± 0.02	0.04 ± 0.04
	y_3	2	0.80 ± 0.17	0.62 ± 0.21	0.14 ± 0.03	0.40 ± 0.30	0.01 ± 0.01	0.32 ± 0.34	0.37 ± 0.28	**0.92 ± 0.00**	0.38 ± 0.02	0.17 ± 0.27
GP	y_1	2	0.05 ± 0.05	0.01 ± 0.01	0.02 ± 0.01	**0.28 ± 0.13**	0.09 ± 0.06	0.03 ± 0.04	0.00 ± 0.00	0.06 ± 0.01	0.04 ± 0.01	0.06 ± 0.04
	y_2	2	0.14 ± 0.07	0.31 ± 0.00	0.22 ± 0.42	0.30 ± 0.14	0.09 ± 0.02	0.12 ± 0.11	0.01 ± 0.01	0.60 ± 0.03	**0.67 ± 0.04**	0.43 ± 0.26
	y_3	2	0.81 ± 0.24	0.74 ± 0.30	0.01 ± 0.02	**0.91 ± 0.20**	0.15 ± 0.03	0.09 ± 0.07	0.00 ± 0.00	0.03 ± 0.00	0.02 ± 0.01	0.08 ± 0.14
MP	y_1	3	**0.44 ± 0.01**	0.32 ± 0.10	0.20 ± 0.01	0.38 ± 0.10	0.37 ± 0.02	0.01 ± 0.01	0.00 ± 0.01	0.43 ± 0.00	0.43 ± 0.00	0.40 ± 0.01
	y_2	6	0.51 ± 0.00	0.41 ± 0.10	0.28 ± 0.01	0.46 ± 0.10	0.39 ± 0.02	0.01 ± 0.02	−0.00 ± 0.01	0.35 ± 0.14	**0.57 ± 0.00**	0.39 ± 0.12
PLA	y_1	7	0.38 ± 0.06	0.34 ± 0.01	0.36 ± 0.05	**0.54 ± 0.31**	0.07 ± 0.02	0.18 ± 0.04	0.00 ± 0.00	0.26 ± 0.09	0.26 ± 0.01	0.18 ± 0.07
	y_2	2	**1.00 ± 0.00**	0.34 ± 0.02	0.24 ± 0.01	0.62 ± 0.35	0.09 ± 0.04	0.74 ± 0.35	0.10 ± 0.06	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
PP	y_1	3	0.55 ± 0.01	0.36 ± 0.10	0.23 ± 0.02	0.37 ± 0.11	0.43 ± 0.10	0.00 ± 0.01	0.00 ± 0.00	0.43 ± 0.09	**0.56 ± 0.00**	0.52 ± 0.04
	y_2	6	**0.59 ± 0.06**	0.45 ± 0.10	0.31 ± 0.02	0.47 ± 0.11	0.45 ± 0.11	0.00 ± 0.00	0.01 ± 0.01	0.36 ± 0.07	0.42 ± 0.01	**0.59 ± 0.10**
SH	y_1	2	0.09 ± 0.02	0.06 ± 0.01	0.06 ± 0.01	0.10 ± 0.03	**0.11 ± 0.05**	0.02 ± 0.01	0.01 ± 0.01	0.10 ± 0.01	0.10 ± 0.02	0.00 ± 0.00
	y_2	6	**0.18 ± 0.01**	0.15 ± 0.03	0.11 ± 0.01	0.13 ± 0.01	0.09 ± 0.01	0.10 ± 0.03	0.00 ± 0.00	0.14 ± 0.02	0.08 ± 0.01	0.00 ± 0.00
	y_3	5	0.16 ± 0.03	0.19 ± 0.06	**0.22 ± 0.03**	0.14 ± 0.01	0.10 ± 0.03	0.13 ± 0.04	0.00 ± 0.00	0.19 ± 0.07	0.15 ± 0.01	0.00 ± 0.00
SC	y_1	2	0.12 ± 0.12	0.00 ± 0.00	0.01 ± 0.02	0.06 ± 0.10	0.00 ± 0.00	**0.18 ± 0.14**	0.00 ± 0.00	−0.00 ± 0.00	−0.00 ± 0.00	0.01 ± 0.01
	y_2	2	**1.00 ± 0.00**	0.99 ± 0.01	**1.00 ± 0.00**	0.65 ± 0.48	0.00 ± 0.00	0.57 ± 0.36	0.04 ± 0.02	0.84 ± 0.35	**1.00 ± 0.00**	0.85 ± 0.11
TP	y_1	2	0.07 ± 0.15	0.00 ± 0.00	**0.38 ± 0.23**	0.00 ± 0.00	0.24 ± 0.24	0.02 ± 0.02	0.00 ± 0.00	0.28 ± 0.03	0.28 ± 0.03	−0.00 ± 0.00
	y_2	2	0.11 ± 0.07	0.00 ± 0.00	0.12 ± 0.09	0.01 ± 0.02	**0.34 ± 0.12**	0.03 ± 0.03	0.00 ± 0.00	0.22 ± 0.03	0.22 ± 0.03	0.00 ± 0.00
Average Rank			**3.06**	4.88	5.84	3.80	6.20	6.78	9.08	4.70	4.44	5.70

A.3 Further Evaluation

To evaluate the clustering performance of the comparison algorithms, Table 4 presents the mean and standard deviation of the Adjusted Rand Index scores averaged over 5 runs. The results demonstrate that the proposed approach, NRTSC, achieved the highest performance with an average rank of 3.06. ENRC emerges as the most competitive baseline, securing the second-best average rank of 3.80.

We also assess the diversity of the non-redundant clusterings generated by each algorithm. Table 5 quantifies this redundancy using the average NMI. In this context, a lower NMI score is better as it indicates the algorithm successfully discovers distinct, non-redundant clusterings. The findings

Table 5: Average NMI per non-redundant clustering algorithm showing redundancy among predicted clusterings. A small value indicates that an algorithm returns distinct, non-redundant clusterings; a large value indicates similar clusterings, therefore, the lower the better.

Dataset	NRTSC	AutoNR	DivClust	ENRC	ISAAC	iMClusts	CMClusts
BME-AC	0.000	0.000	0.500	0.000	0.038	0.525	0.000
BumpFreq	0.000	0.000	1.000	0.000	0.126	0.512	0.000
DistalPhalanx	0.016	0.008	0.779	0.054	0.172	0.007	0.010
DodgerLoop	0.318	0.077	0.742	0.053	0.004	0.490	0.064
GunPoint	0.387	0.032	0.784	0.037	0.044	0.776	0.003
MiddlePhalanx	0.336	0.013	0.773	0.000	0.094	0.007	0.008
Plane-AC	0.398	0.013	0.815	0.271	0.035	0.317	0.001
ProximalPhalanx	0.004	0.000	0.800	0.013	0.091	0.006	0.010
SemgHandCh2	0.348	0.098	0.743	0.171	0.079	0.431	0.007
SyntheticControl	0.021	0.000	0.737	0.000	0.000	0.231	0.006
TwoPatterns	0.004	0.000	0.749	0.002	0.164	0.319	0.000

reveal that NRTSC effectively avoids redundant outputs with scores at or near 0.000 for datasets such as BME-AC, BumpFreq, ProximalPhalanx, SyntheticControl, and TwoPatterns. While AutoNR and CMClusts also demonstrate a strong capacity for generating non-redundant clustering, they have lower clustering performance in general. Moreover, the gater-learned feature-selection criterion supports NRTSC 's ability to find non-redundant clustering. Figure 7 shows how the learned embeddings and the gater selections serve the goal of non-redundant clustering. The learned embeddings show the different clusterings in the different subspaces, and the gater selection (Figure 8 for each sample are complementary, i.e. the masks exhibit complementary weights across corresponding positions in the facet-specific latent blocks..

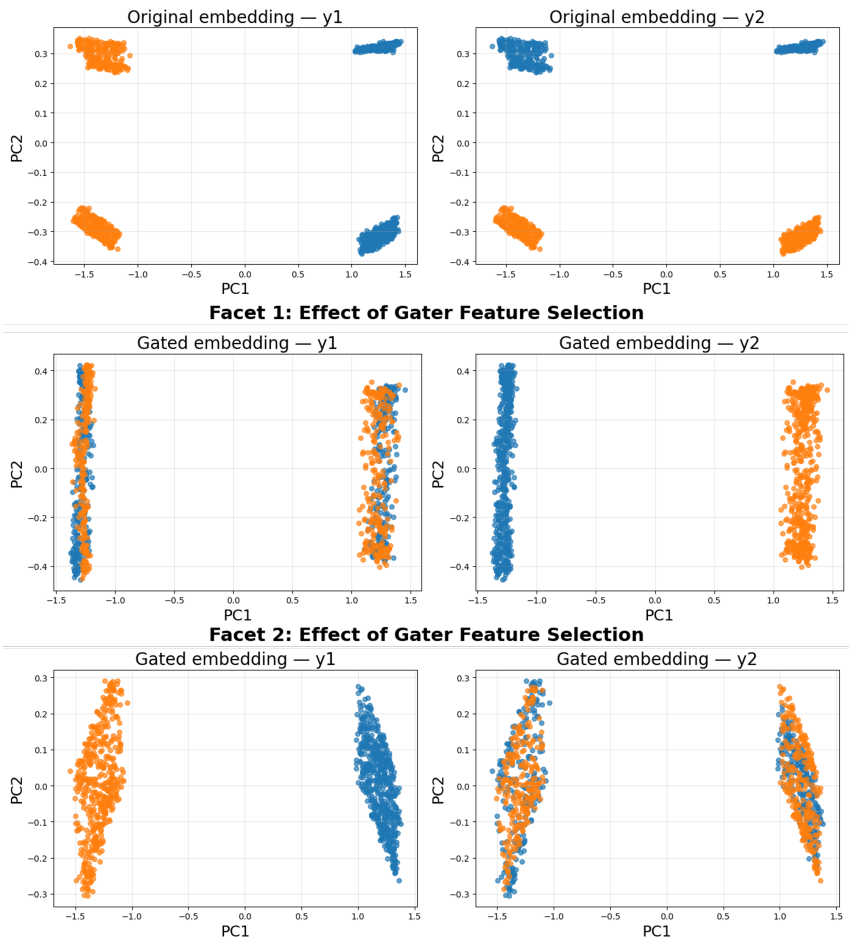


Figure 7: Facets latent space of NRTSC on the BumpFreq dataset, in addition to the facet representation after selecting relevant features. The figure shows the effect of the gater for feature selection.

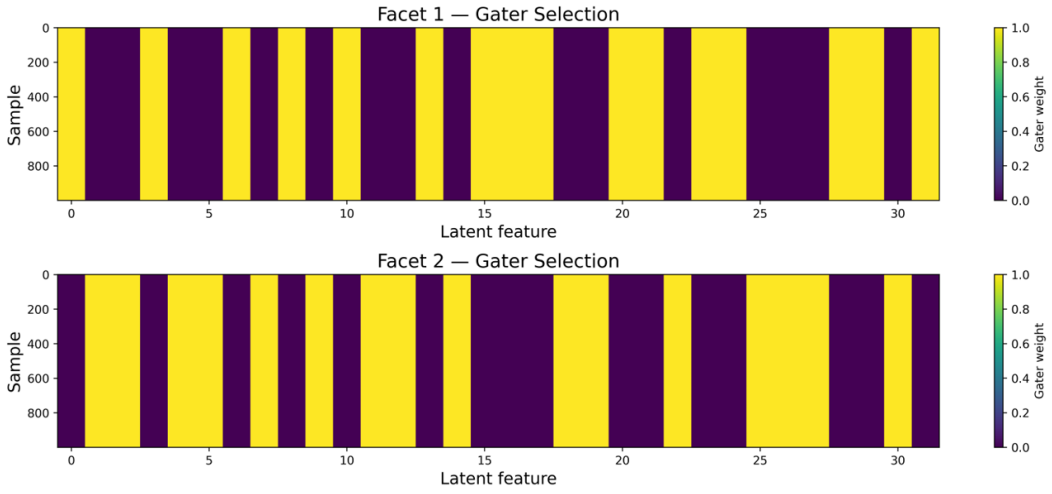


Figure 8: Gater mask predictions for the BumpFreq dataset.