SOM-CPC: Unsupervised Contrastive Learning with Self-Organizing Maps for Structured Representations of High-Rate Time Series

Iris A.M. Huijben
Department of Electrical Engineering
Eindhoven University of Technology
Onera Health, Eindhoven
i.a.m.huijben@tue.nl

Arthur A. Nijdam
Department of Electrical Engineering
Eindhoven University of Technology

Sebastiaan Overeem
Sleep Medicine Center Kempenhaeghe, Heeze

Merel M. van Gilst
Sleep Medicine Center Kempenhaeghe, Heeze

Ruud J.G. van Sloun
Department of Electrical Engineering
Eindhoven University of Technology

Abstract

Continuous monitoring with an ever-increasing number of sensors has become ubiquitous across many application domains. Acquired data are typically high-dimensional and difficult to interpret, but they are also hypothesized to lie on a lower-dimensional manifold. Many deep learning (DL) models aim to identify this manifold, but do not promote structure nor interpretability. We propose the SOM-CPC model, which jointly optimizes Contrastive Predictive Coding (CPC), and a Self-Organizing Map (SOM) to find such an organized manifold. We address a largely unexplored and challenging set of scenarios comprising high-rate time series, and show on synthetic and real-life medical and audio data that SOM-CPC outperforms strong baseline models that combine DL with SOMs. SOM-CPC has great potential to expose latent patterns in high-rate data streams, and may therefore contribute to a better understanding of many different processes and systems.

1 Introduction

The improvement and abundance of sensor technology has led to large amounts of high-dimensional continuous data streams that are information-rich but challenging to interpret. Interpretability is, however, vital to truly gain actionable insights from acquired data. The main objective of this study is to develop an algorithm for acquiring a structured and interpretable representation of (high-rate) time series. According to the manifold hypothesis, high-dimensional real-world data lies on a low-dimensional manifold, comprising disentangled latent factors of variation. The area of unsupervised representation learning is concerned with models that learn this manifold from a set of training data, without the bias of human annotations. Traditional methods to find lower-dimensional embeddings, such as principle component analysis (PCA), have nowadays been outperformed by expressive non-linear Deep Neural Networks (DNNs). However, the resulting latent space is typically unstructured and hard to interpret. As a solution, DNNs have been combined with clustering algorithm such as K-means, spectral clustering or hierarchical clustering. Recently, models that combine DNNs with

Preprint. Under review.

Kohonen's Self-Organizing Maps (SOMs) have been developed [2, 6, 7, 9, 10, 13, 19, 26], which we dub deep-SOM models.

Most of these models have focused on autoencoders as feature extractors. However, similar to [20], we hypothesize that their reconstruction objective may hamper the clustering or structured representation learning objective: while within-cluster similarities should remain preserved for latent clustering, reconstruction demands a preservation of all factors of similarity. Moreover, in the context of time series representation learning, other unsupervised models - that take the temporal nature of the data into account during training - might be more suitable.

Contrastive self-supervised learning approaches have quickly become popular methods thanks to their superior representation learning performance in many domains (see [18] for a review). While many of these models rely on data augmentations during training in order to construct pairs of similar data points, Contrastive Predictive Coding (CPC) [21] explicitly leverages the temporal dimension for this purposes, making it a natural choice for unsupervised representation learning of time series. In CPC, the temporal dimension not only serves as a pretext task, but simultaneously enforces smoothness over time. The contributions of this work are as follows:

- We propose a new model in the deep-SOM family: *SOM-CPC*, which is suitable for learning structured and interpretable 2D representations of (high-rate) time series by encoding subsequent data windows to a topologically ordered set of quantization vectors.
- We show that SOM-CPC quantitatively and qualitatively outperforms autoencoder-based deep-SOM models in terms of both clustering and topological ordering.
- Compared to other deep-SOM models for temporal data, our SOM-CPC requires far less auxiliary loss functions (and associated hyperparameter tuning).
- We analyze the training behavior of SOM-CPC and observe that the SOM clustering objective better aligns with the CPC objective than with a reconstruction loss.

2 Preliminaries and related work

2.1 Kohonen Self-Organizing Maps

Kohonen's Self-Organizing Map (SOM) [17] is an algorithm to find a 2-dimensional (2D) topological data representation. It has been found useful to reveal intricate patterns and structure in a plethora of applications. The output of the algorithm, the 2D representation, is often referred to as a SOM as well.

We define a set of data points $\mathcal{Z}$, and quantized counterparts $\phi_j^{(n)} = q_\Phi(\boldsymbol{z}) \in \Phi$ for $\boldsymbol{z} \in \mathcal{Z}$, at a given point in training (denoted with superscript (n)). The set $\Phi : \{\phi_1, \dots, \phi_k\}$ is a trainable quantization codebook containing k vectors or prototypes $\phi_i \in \mathbb{R}^F, 1 \leq i \leq k$. Vector $\phi_j^{(n)}$ can be considered the 'winning vector' for data point $\boldsymbol{z}$. For interpretability, the learned codebook vectors are placed on a pre-defined 2D grid by assigning an xy-coordinate to each vector at initialization. During training, each ϕ_i is updated as follows [17], with $\boldsymbol{z} \in \mathcal{Z}$:

$$\phi_i^{(n+1)} = \phi_i^{(n)} + \eta^{(n)} \mathcal{S}_i\big(\phi_j^{(n)}\big)\big(\boldsymbol{z} - \phi_i^{(n)}\big), \tag{1}$$

where $\eta^{(n)}$ is a time-decreasing learning rate. A topological neighborhood structure is promoted via a Gaussian function:

$$\mathcal{S}_i\big(\phi_j^{(n)}\big) = \exp\Big(-\frac{(d_{j,i}^{(n)})^2}{2(\sigma^{(n)})^2}\Big), \qquad \text{with} \tag{2}$$

$$d_{j,i}^{(n)} = ||\mathcal{P}\{\phi_j^{(n)}\}, \mathcal{P}\{\phi_i^{(n)}\}||_2 \text{ and} \tag{3}$$

$$\sigma^{(n)} = \sigma^{(0)} \exp(-n/\lambda), \tag{4}$$

where $\mathcal{P}$ projects a codebook vector to its corresponding xy-coordinate on the grid, σ_0 denotes the initial standard deviation, and λ the decay factor. Setting $\lambda = -n_{\text{max}} / \log(\sigma^{(n_{\text{max}})}/\sigma^{(0)})$ ensures that σ varies between $\sigma^{(0)}$ and $\sigma^{n_{\text{max}}}$ in n_{max} steps. The distance equals zero for $\phi_i^{(n)} = \phi_j^{(n)}$, implying a scaling of 1 for the winning node.

2.2 Deep-SOM models

Sensor data usually exist in a high-dimensional space that is difficult to cluster. SOMs are, therefore, typically trained on (compressed) data features, rather than the raw data directly. To eliminate the need for heuristic feature selection, a line of research has focused on exploiting the expressive power of deep learning models to learn features, while jointly training a SOM on these features. In this joint training strategy, the SOM objective can be seen as a regularizer on the encoding procedure, as it promotes a cluster-friendly feature space. Resulting models - dubbed deep-SOM models here - are in fact representation learning models that aim to find a lower-dimensional, and interpretable, manifold in which latent factors of variation are supposedly disentangled.

Almost all deep-SOM research has focused on combining autoencoders [6, 7, 9, 10, 19, 26] with a SOM. These models can broadly be summarized as a vector-quantized (VQ) VAE [27], with a topological organization of the vectors in the quantization codebook: the SOM. The models are trained end-to-end using error backpropagation of both a reconstruction *task* loss $\mathcal{L}_{\text{task}}$ and a loss $\mathcal{L}_{\text{topo}}$ that encourages *topological* ordering in the SOM. In general, a deep-SOM training objective takes the following form:

$$\mathcal{L}_{\text{deep-SOM}} = \mathcal{L}_{\text{task}} + \alpha \mathcal{L}_{\text{topo}} \qquad \text{with} \tag{5}$$

$$\mathcal{L}_{\text{topo}}(\boldsymbol{z}^{(n)}) = \mathbb{E}_{\mathcal{Z}}\Big[\sum_{i=1}^{k} \mathcal{S}_i\big(\phi_j^{(n)}\big)||\boldsymbol{z}^{(n)} - \phi_i^{(n)}||_2^2\Big], \tag{6}$$

and α controls the task-topological ordering trade-off. The topological loss thus replaces the original update rule of the SOM algorithm (as in eq. (1)). The features $\boldsymbol{z} \in \mathcal{Z}$ are jointly optimized, and therefore also depend on n now. To prevent clutter, we will, however, further omit the (n)-superscript in the rest of this work.

2.2.1 SOM-VAE and its extensions

The authors of [10] propose the SOM-VAE model. With respect to the VQ-VAE model, the SOM-VAE has two decoders, as it also decodes the continuous latents, and it enforces topological organization of the codebooks vectors, by forcing neighbour nodes close to the winning node. The encoder parameters were, however, left uninfluenced by the quantization error of the neighbour nodes.

To facilitate the latter, the topological loss was split in a *commitment* loss (commiting the selected codebook vector to the corresponding continuous embedding and *vice versa*) and a *SOM* loss (enforcing the codebook vectors of the neighbours to move close to $\boldsymbol{z}$ as well): $\mathcal{L}_{\text{topo}} = \mathcal{L}_{\text{commitment}} + \frac{\beta}{\alpha}\mathcal{L}_{\text{SOM}}$. Formally:

$$\mathcal{L}_{\text{commitment}} = \mathbb{E}_{\mathcal{Z}}\Big[||\boldsymbol{z} - \phi_i||_2^2\Big], \qquad \text{and} \tag{7}$$

$$\mathcal{L}_{\text{SOM}} = \mathbb{E}_{\mathcal{Z}}\Big[\sum_{i=1, i\neq j}^{k} \mathcal{S}_i\big(\phi_j\big)||\,\text{sg}[\boldsymbol{z}] - \phi_j||_2^2\Big], \tag{8}$$

with as $\text{sg}[\cdot]$ a gradient blocker that impedes gradient updates to the encoder. Both reconstruction losses $\mathcal{L}_{\text{recon}}$ from the continuous and discrete decoder yield the total task loss $\mathcal{L}_{\text{task}}$, which is combined with the topological loss to create the training objective of the SOM-VAE model. Figure 1a visualizes the full architecture, including the gradient paths in green. The authors of the SOM-VAE [10] chose a static neighbourhood function that includes the selected node and its four direct neighbours on the grid, i.e. up, down, left, and right. Note that for $0 < \beta/\alpha < 1$, this *plus*-kernel is a coarse approximation of a Gaussian.

The SOM-VAE-prob model [10] and the (Temporal-)Deep Probabilistic SOM or (T)-DPSOM [19] are extensions of the SOM-VAE. SOM-VAE-prob enforces smoothness over time by adding a transition loss (multiplied by γ) to optimize a first-order Markov model to learn the node transition probabilities and a smoothness loss (multiplied by τ) to minimize the quantization MSE of highly probable transitions. The T-DPSOM is a probabilistic model, based upon the *variational* autoencoder [15] with soft cluster assignment and a cluster assignment hardening (CAH) loss [28].

The temporal variant of the model additionally incorporates a temporal smoothness loss, and an LSTM, which aims to predict the future latent space. Interestingly, this latter functionality is very

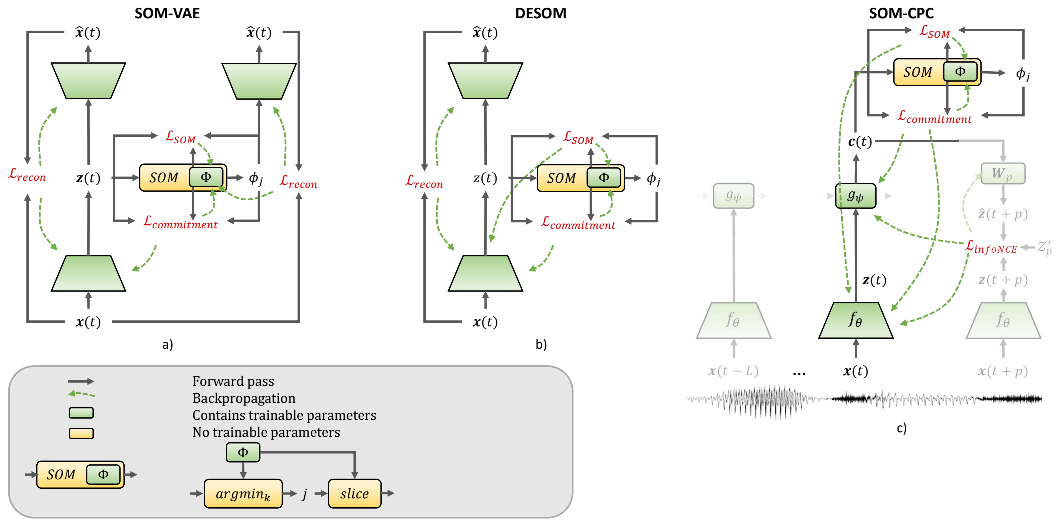

Figure 1: Architectures of different deep-SOM models including the gradient paths in green. a) SOM-VAE [10] b) DESOM [9] c) SOM-CPC (ours). The two decoders in the SOM-VAE model are independent and have their own trainable parameters, while the visualized encoders in the SOM-CPC model are all the same (i.e. parameters are shared). The g_ψ block in the SOM-CPC model indicates an autoregressive component (e.g. a GRU), and $\mathcal{Z}'_p$ refers to a set of drawn negative embeddings.

similar to the way in which an autoregressive module is also used in SOM-CPC training. As we focus on comparing autoencoders to CPC as feature extractors for joint SOM training, the probabilistic additions in the (T-)DPSOM model are orthogonal to the developments in this work.

2.2.2 (LSTM-)DESOM

The authors of [9] propose a variant on the SOM-VAE model, called Deep embedded SOM (DESOM). Compared to SOM-VAE, the decoder on the discrete space is omitted, there is no blocking of gradients, and the topological loss $\mathcal{L}_{\text{topo}}$ was adopted as given in eq. (2), with a Gaussian neighbourhood function with decaying variance over time, as suggested from the original SOM algorithm. Figure 1b visualizes the DESOM architecture including the gradient paths in green. In another short work, [7] speculate about adding an LSTM in the latent space to train a SOM on sequential data, and refer to this model as LSTM-DESOM.

3 SOM-CPC

3.1 Motivation

In this work, we propose the SOM-CPC model, a representation learning model that learns to map time series data to a structured 2D grid. The model jointly optimizes a temporal contrastive learning objective to extract features, and a topological loss that organizes the SOM space.

In order to learn features that are both suitable for SOM organization *and* accurately reflect the data, the model should ideally invert the original data generating process (which is in general unknown and implicit). Assuming that this generative process has been highly non-linear, feature learning can be formulated as a non-linear independent component analysis (ICA) problem, which has proven to be non-identifiable [11]. However, recent advances showed that the problem becomes identifiable under the assumed presence of an auxiliary variable [12]. Such an auxiliary variable (e.g. a temporal component) is not present in plain autoencoders, but the contrastive learning paradigm has shown to conform to this assumption [12, 29]. This theory is in line with the hypothesis stated by the authors of [20] that a reconstruction objective may hamper clustering performance in the latent space.

3.2 Algorithmic details

We introduce $\mathcal{X} = \{\ldots, \boldsymbol{x}(t), \boldsymbol{x}(t+1), \ldots\}$, a set of non-overlapping data windows $\boldsymbol{x}(t) \in \mathbb{R}^{\text{ch}\times T}$, with ch the number of channels, and T the number of samples in the window. For brevity we omit the time index when possible.

An encoder, parameterized by θ, maps each data window $\boldsymbol{x}$ to a latent representation $\boldsymbol{z} = f_\theta(\boldsymbol{x}) \in \mathbb{R}^F$, with F the number of features. The set $\mathcal{Z}$ includes the embeddings of all windows in $\mathcal{X}$. A causal auto-regressive (AR) module g_ψ parameterized by ψ, e.g. a gated-recurrent unit (GRU), subsequently aggregates the current and L previous embeddings, to generate a (current) context vector $\boldsymbol{c}(t) \in \mathbb{R}^F$. Given this context, the pretext task in our SOM-CPC model aims to minimize the prediction error for P future (or 'positive') embeddings $\boldsymbol{z}(t+p)$, for $p \in \{1, \ldots, P\}$, compared to this error for N 'negative' embeddings. These negatives may be sampled across the dataset, or within the same time-series, and are on the fly encoded to their latent representation during training. The training objective, being the InfoNCE loss [21], is defined as:

$$\mathcal{L}_{\text{task}} = \mathcal{L}_{\text{InfoNCE}} = \frac{1}{P}\sum_{p=1}^{P} \mathcal{L}_p, \qquad \text{with} \tag{9}$$

$$\mathcal{L}_p = -\mathop{\mathbb{E}}_{\mathcal{X}}\Big[\log \frac{\exp\Big(\boldsymbol{z}(t+p)\mathbf{W}_p\boldsymbol{c}(t)\Big)}{\sum_{\boldsymbol{z}'\in\mathcal{Z}'_p\cup\{\boldsymbol{z}(t+p)\}}\exp\Big(\boldsymbol{z}'\mathbf{W}_p\boldsymbol{c}(t)\Big)}\Big],$$

with $\mathcal{Z}'_p \subset \mathcal{Z}$ a set of embeddings of drawn negative samples ($|\mathcal{Z}'_p| = N$), and $\mathbf{W}_p \in \mathbb{R}^{F\times F}$ a trainable mapping between the context vector and a future embedding.

The context vector is not only used to predict future embeddings, it is also the input to the SOM module that selects the winning node. The SOM is optimized using the topological loss $\mathcal{L}_{\text{topo}}$, as defined in eq. (6), with a Gaussian neighbourhood kernel $\mathcal{S}$, as defined in eq. (2). Depending on the use case, it might not be necessary to use the AR module g_ψ to aggregate causal context into the current embedding. If the AR module is not used, the future predictions are made directly from the current (continuous) latent space $\boldsymbol{z}(t)$ instead of $\boldsymbol{c}(t)$. Likewise, $\boldsymbol{z}(t)$ rather than $\boldsymbol{c}(t)$ is being quantized by the SOM module. Depending on the presence of this AR module, both $\mathcal{L}_{\text{topo}}$ and $\mathcal{L}_{\text{task}}$ are thus computed on either $\boldsymbol{z}(t)$ or $\boldsymbol{c}(t)$.

The model is optimized in a one-step joint training approach, with the training objective being: $\mathcal{L}_{\text{SOM-CPC}} = \mathcal{L}_{\text{task}} + \alpha\mathcal{L}_{\text{topo}}$, which adheres to the general objective of a deep-SOM model as formulated in eq. (5). Figure 1c provides an overview of the SOM-CPC model, and its gradient paths in green. The initial standard deviation σ_0 of the Gaussian kernel (from eq. (4)) was fixed to yield half the squared-root of the number of SOM nodes k, since a square SOM grid was used in all cases. Algorithm 1 in appendix A.1 provides pseudocode of the full SOM-CPC algorithm.

3.3 Performance evaluation

Comparing the SOM-CPC model to other deep-SOM models requires performance metrics that are indicative for SOM performance. The authors of [8] provide a taxology of SOM metrics, in which a distinction is made between *external* vs *internal* and *topological* vs *clustering* metrics. External metrics are related to labels that are available for the data (but not used during unsupervised training), while internal metrics do not depend on such information.

Topological metrics assess the topological ordering (i.e. neighbourhood relations) of the SOM, while clustering metrics are more related to, for example, pureness of nodes. To evaluate clustering performance, linked to external labels, we leverage *purity* [4] and the *normalized mutual information* (NMI). The latter corrects for a high number of clusters (i.e. nodes), which could easily lead to high pureness, but leaves the NMI more conservative.

We, moreover, leverage *Cohen's kappa* [5], which is commonly used in classification problems, e.g. in sleep staging, and corrects for correctness by chance. This metric is computed by first 'coloring'

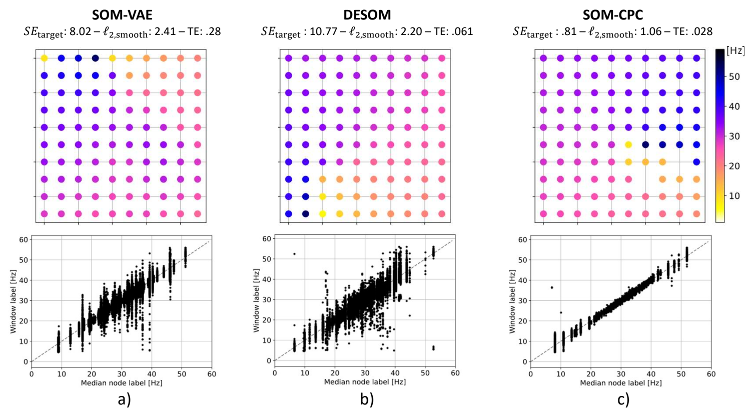

Figure 2: SOMs for the SOM-VAE (a), DESOM (b) and SOM-CPC (c) models indicated with an $*$ in table 1. Both the DESOM and SOM-CPC model show a gradual change of frequency over the grid, but the SE_{target} is lower for the SOM-CPC model, which can also be seen from the bottom plots, where the labels (in Hz) of all test set windows are plotted (y-axis) against the training set median label (x-axis) for the node on which the window was mapped.

(or labelling) the nodes with the most occurring label from the training set. The test set predictions, to be compared to the test set labels, are then converted from node indices to class labels by using these colorings. The highest score for the clustering and classification metrics yields 1. For the experiments in section 4.1 that leverage continuous labels, we compute the squared regression errors (SE) between the test set label predictions and the node labels assigned using the training set (similar procedure as for Cohen's kappa), and denote this metric with SE_{target}.

Topographic performance is measured using the (internal) *topographic error* (TE) [16], which reports the fraction of windows (i.e. between 0 and 1) for which the winning and second-best winning node are not neighbours in the SOM. The lower this error, the better the topology of the SOM. Finally, to measure whether a time series conveys a smooth trajectory through SOM space, we measure the average Euclidean distance (denoted $\ell_{2,\text{smooth}}$) between two subsequent windows in each time series, which is an internal metric as well. The lower this value, the less frequently large jumps in the 2D map occur. However, note that in extreme cases where the model performed very poorly and many windows collapsed to the same node, both the TE and the average $\ell_{2,\text{smooth}}$ metric are artificially pushed down. We can thus only interpret these metrics in conjunction with earlier-mentioned clustering and classification metrics.

The main goal of the experiments is to investigate the effect of the contrastive pretext task versus a reconstruction task (i.e. CPC vs autoencoder) on SOM performance in a joint training setup. To make the fairest possible comparison, the same encoder architecture was used for all models that were compared within a single experiment. In some cases, slight adaptations to the encoder were, however, made when it was found to highly benefit a specific model. All models were run with the same seed for randomization. The details on model architectures for experiments on simulation data, sleep data and audio data can respectively be found in appendix A.2.1, A.3.2, and A.4.1.

4 Experiments

This section describes the experiments and corresponding results of the SOM-CPC model and earlier-introduced deep-SOM models on synthetic data (section 4.1), sleep recordings (section 4.2), and audio data (section 4.3).

4.1 Synthetic data

We created a synthetic dataset to analyze the properties of the SOM-VAE, DESOM and SOM-CPC in a controlled environment. We generated sinusoids with an amplitiude of 1, sampled at 128 Hz, and an initial frequency sampled from a uniform distribution between 20 and 40 Hz. The frequency of the signals was altered over time according to a random walk process with a step size of 0.1 Hz. As such, at each time step (i.e. sample), the signal's frequency either increased with 0.1 Hz (with probability $p_{\text{up}} = 0.1$), decreased with 0.1 Hz ($p_{\text{down}} = 0.1$), or remained constant ($p_{\text{c}} = 0.8$). In case the random walk crossed either 1 or 60 Hz, the probabilities were (temporarily) altered to [p_{up}, p_{c}, p_{down}] = $[0.5, 0.5, 0]$ or [p_{up}, p_{c}, p_{down}] = $[0.0, 0.5, 0.5]$, respectively, to keep the sinusoid's frequency in the set range. Each series was finally corrupted with additive white Gaussian noise. Formally, each generated signal took the form:

$$\boldsymbol{x}[n] = \sin\left(2\pi \frac{f[n-1] + \Delta f}{f_s} n\right) + \epsilon,$$

with $U[a, b]$ a uniform random variable between a and b, $f[n = 0] \sim U[20, 40]$, $\Delta f \sim \text{Categorical}([p_{\text{up}}, p_{\text{c}}, p_{\text{down}}])$, $f_s = 128$ Hz the sampling frequency, and $\epsilon \sim \mathcal{N}(0, 0.01)$. A total of 200 of such time-series, each of 5 minutes, were generated, and the set was randomly divided into a training ($n = 100$), validation ($n = 50$), and test split ($n = 50$). Labels were created per 1-second window, by taking the median frequency of each window.

Table 1 show the test set performance for various models, run with different values of α (the multiplier of $\mathcal{L}_{\text{topo}}$). Results below a horizontal dashed line are ablations. The standard deviation, denoted with $\pm$ for $\ell_{2,\text{smooth}}$, is measured across the 5-minute sequences in the test set. Comparing the CPC objective to the reconstruction task (SOM-CPC vs the rest), we see that for all α values, the SOM-CPC models outperformed the autoencoder-based models in terms of TE. A comparison of the best models per category (based on the performance of all three metrics, and denoted in bold), also shows lower SE_{target}, $\ell_{2,\text{smooth}}$, and TE for the SOM-CPC models compared to the other models. Figure 2 displays the resulting SOMs (colored with the median test set labels) for the three models denoted with a $*$ in the table. Uncolored nodes in the SOM were not assigned in the test set. Interestingly, although the SOM for the DESOM and SOM-CPC model look similar, the SE_{target} is higher for the DESOM model, which can also be seen from the graphs below the SOMs in fig. 2. These graphs plot the spread of windows' labels on a node (y-axis) against the median node label as assigned by the training set (x-axis). An SE_{target} of 0 would be achieved when all dots in the plot are perfectly aligned on the diagonal. It is clear that the SOM-CPC model is best able to achieve this. Figure 6 in appendix A.2.3 shows the principle component analysis (PCA) projections of the latent spaces of the same three models as for which the SOMs were visualized in fig. 2. These PCA plots show that the latent space disentanglement of the SOM-CPC model is much better than for the SOM-VAE and DESOM models, explaining the better performance of the SOM-CPC model. Note the gradient in the colorings of the SOM and PCA plots of the SOM-CPC model, which follow a spiral trajectory from low (yellow) to high (blue) frequencies. We created a 1D synthetic problem by only varying the frequency of the signal over time, so the most natural mapping in 2D is indeed to spiral this continuous frequency across the map.

Table 1 shows that using a plus neighbourhood kernel instead of a Gaussian in the SOM-CPC model decreased performance on all three metrics. The increase in TE, is well explainable by the fact that a plus kernel takes into account fewer neighbours (at least at the start of training) and therefore has more difficulty to find a good topological mapping. Whether or not the gradients of the neighbour nodes with respect to the encoder were blocked in our model ($\mathcal{L}_{\text{SOM}}$ $\text{sg}[\cdot]$ column) did not seem to result in large differences. While we expected that the topological ordering would benefit from the SOM loss gradients, the already low TE metric might be hard to improve by allowing gradient flow in this use case.

The addition of two temporal losses in the SOM-VAE-prob model, as compared to vanilla SOM-VAE, did deteriorate the given metrics, even though a range of values for multipliers γ and τ was tested (see table 3 in appendix A.2.3 for the full sweep). The deterioration of the results can be explained by the difficulty of finding the correct scaling factors for these additional losses. Note that the SOM-CPC model automatically incorporates smoothness over time thanks to the nature of the CPC task loss, therewith preventing additional hyperparameter tuning. The test set regression performance of SOM-CPC, as compared to vanilla CPC, was found to be slightly better. We suspect that prediction via SOM nodes automatically incorporated some filtering of outlier predictions, by coloring the nodes with the median train set label. This effect could explain the slightly lower SE_{target} for SOM-CPC.

Finally, we study the optimization behavior of the SOM-VAE model versus SOM-CPC. To make a fair comparison, we compare it with the SOM-VAE model, trained with a Gaussian neighbourhood kernel. The training of the task versus topological loss is depicted in fig. 5 in appendix A.2.3. Each line depicts one run with a different value for α, and the line color's gradient denotes the training

α	$\mathcal{S}$	$\mathcal{L}_{\text{SOM}}$ sg[$\cdot$]	SE_{target}	$\ell_{2,\text{smooth}}$	TE
CPC + supervised classifier [21]					
-	-	-	2.62	-	-
SOM-VAE (with $\beta = \alpha/5$) [10]					
1e-3	Plus	✓	11.59	2.60±.46	.38
1e-2	Plus	✓	14.57	2.98±.84	.38
* .1	Plus	✓	**8.02**	**2.41±.68**	**.28**
1	Plus	✓	13.43	2.75±.45	.33
1e-5	Gaussian	✓	11.12	1.93±.29	.069
1e-4	Gaussian	✓	**10.52**	1.95±.36	.075
1e-3	Gaussian	✓	11.60	**1.92±.34**	**.056**
1e-2	Gaussian	✓	18.13	2.03±.42	.086
SOM-VAE-prob (with $\beta = \alpha/5$, $\gamma =$ 3.3e-4, $\tau =$ 1e-2) [10]					
.1	Plus	✓	20.10	3.15±.73	.63
DESOM [9]					
1e-5	Gaussian	✗	14.00	1.99±.36	.13
1e-4	Gaussian	✗	19.09	1.95±.28	.11
1e-3	Gaussian	✗	12.58	1.92±.31	.077
1e-2	Gaussian	✗	13.66	**1.89±.33**	.065
* .1	Gaussian	✗	**10.77**	2.20±.44	**.061**
1	Gaussian	✗	39.4	2.19±.33	.067
SOM-CPC (ours)					
1e-5	Gaussian	✗	.95	1.24±.31	.048
1e-4	Gaussian	✗	.72	1.37±.37	**.022**
* 1e-3	Gaussian	✗	.81	**1.06±.28**	.028
1e-2	Gaussian	✗	**.62**	1.08±.28	.059
.1	Gaussian	✗	1.90	1.18±.30	.039
1e-5	Gaussian	✓	.64	1.04±.26	**.039**
1e-4	Gaussian	✓	.68	1.19±.43	.057
1e-3	Gaussian	✓	.75	1.12±.32	.059
1e-2	Gaussian	✓	**.47**	**.99±.24**	.069
.1	Gaussian	✓	.89	1.16±.32	.069
1e-3	Plus	✗	1.36	2.27±.44	.22
1e-2	Plus	✓	1.16	1.85±.26	.12

Table 1: Test set performance of various deep-SOM models on our synthetic data. Models below dashed lines are ablations. The SOM-CPC model clearly outperforms the autoencoder-based models. SOMs of models with a * are visualized in fig. 2. More results of the SOM-VAE-prob model can be found in table 3 in appendix A.2.3.

iteration. It can be seen that both losses seem to be counteracting each other during training of the SOM-VAE model, while the optimization trajectory of SOM-CPC is more smooth.

4.2 Sleep recordings

We analyse SOM-CPC on real-world medical time-series from subset 3 of the publicly available Montreal Archive of Sleep Studies (MASS) database [22], consisting of whole-night polysomnography recordings. Details on the data preprocessing and the model training can be found in appendix A.3.1 and A.3.3, respectively.

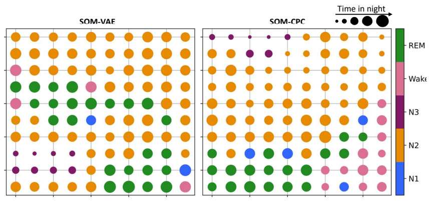

Figure 3: SOMs trained on sleep recordings by SOM-VAE and SOM-CPC (indicated with a $*$ in table 5, appendix A.3.4). SOM-CPC finds much more logical clusters: deep sleep N3 is isolated from light sleep N1, Wake and REM sleep with a thick cluster of medium-deep sleep N2 in orange.

Table 5 in appendix A.3.4 shows the results of different deep-SOM models, and CPC followed by a supervised classifier. The SOM-CPC model outperformed SOM-VAE and DESOM on all metrics. Its classification performance was found on par with a CPC model followed by a supervised classifier. Whether or not the gradients of the SOM loss were stopped towards the encoder did not greatly influence this result. Changing the neighbourhood kernel to a plus instead of a Gaussian (see below the dashed lines in the table), deteriorated all metrics, but mainly the TE. Figure 3 shows the SOMs of the two models indicated with a $*$ in table 5 (appendix A.3.4). The nodes were colored with the most-occurring sleep stage labels of the test set windows on each node. Moreover, the size of each node denotes the average time in the night. SOM-CPC resulted in a SOM that shows very logical clusters, i.e. deep sleep N3 is isolated from lighter forms of sleep (i.e. N1, Wake and REM sleep) by a thick cluster of medium-deep sleep N2. This pattern is, however, not visible for the SOM created by the SOM-VAE model. Interestingly, the size of the nodes may inform us about different sub-categories of sleep within each pre-defined sleep stage. A difference is, e.g. , seen in node sizes within both the N2 and N3 cluster, suggesting that there might be differences in N2 and N3 sleep across the night.

4.3 Audio

For the audio experiments, we use a subset of the publicly available LibriSpeech dataset [24]. The dataset contains multiple minute-long English voice recordings, sampled at 16 KHz and of 251 different speakers. We used the publicly available train-test split, as provided by [21], and created an additional validation set by randomly selecting 25% of the training set. We ensured that at least one recording of each speaker was present in all three splits. We selected the recordings of the ten speakers with the longest recording time and created a subset which we refer to as *LibriSpeech-10s*. This finally contained a total of 150.9, 54.6, and 46.5 minutes in the training, validation, respectively test set. The used train-validation-test split, including the necessary code will be made available upon publication. The full model and training details can be found in appendix A.4.

Table 7 in appendix A.4.2 shows the results of different models trained on the *Librispeech-10s* dataset. It can be seen that the SOM-CPC model outperforms all variants of the DESOM model (DESOM, GRU-DESOM that reconstructs only the last window, GRU-DESOM that reconstructs the full sequence) by a wide margin and for all choices of the α parameter. The GRU slightly increased the performance of the DESOM model in terms of clustering metrics, but it was still unable to reliably identify and cluster speakers in its SOM. The difference in performance between the GRU-DESOM and the SOM-CPC models is also visible in fig. 4. The SOM-CPC model has clustered the SOM nodes belonging to the same speaker, and seems to group male and female speakers closer together.

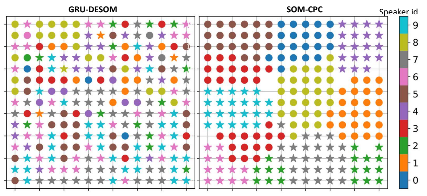

Figure 4: SOMs of audio data. The SOM-CPC model is able to cluster different speakers, while the GRU-DESOM model results in a less clustered SOM. The star versus the dot denotes the gender.

5 Discussion

In this work we proposed a new member of the deep-SOM family: SOM-CPC, suitable for interpretable representation learning of high-rate data streams. In general, the CPC objective implicitly enforces temporal smoothness, while autoencoder-based models require additional losses and hyperparameter tuning to achieve this. Moreover, CPC's pre-text task alignes better with the topological SOM objective, and the non-linear ICA community has provided proof of identifiability for contrastive learning.

Table 1, table 5 (appendix A.3.4) and table 7 (appendix A.4.2) showed that the SOM-CPC model outperformed autoencoder-based deep-SOM models for both synthethic and real time series, on clustering (NMI, purity), classification (Cohen's kappa), and topographic metrics ($\ell_{2,\text{smooth}}$, TE). Resulting SOMs were shown to reveal more logical clustering patterns (see Figures 2 to 4 (appendix A.2.3)), with fewer sensitivity to the topological-task trade-off multiplier α (see table 1 and fig. 5). Figure 5 (appendix A.2.3) also showed that the reconstruction objective may indeed counteract the topological objective, as already hypothesized by [20], while the InfoNCE loss, as used in our SOM-CPC model, seemed to coincide more with this topological loss. Even better, the SOM objective did not hamper CPC optimization at all, given the fact that the regression/classification performance of SOM-CPC was found on par with CPC followed by a supervised classifier. The use of a Gaussian neighbourhood kernel, as opposed to a plus kernel, was found to improve the topological ordering in the SOM, measured by the TE metric. One should note, that due to the decreasing variance of this Gaussian kernel over time during training, the topological loss value automatically reduces, due to summing the contribution of fewer neighbours. This is something to consider when training these models and interpreting the loss curves. No decisive conclusions could be made regarding whether or not the gradients of the part of the topological loss that acts on the neighbour nodes in the SOM (i.e. the SOM loss) should be detached from the encoder. Using these gradients for encoder updates did at least not hurt performance, so for coding simplicity, we would advice to not detach the SOM loss.

We believe that the SOM-CPC model opens up new research directions for interpretable representation learning of time series. Directions for further research include investigation to whether additions like the soft-cluster assignment, cluster hardening loss or a Gaussian latent prior - which have shown to improve the SOM-VAE model [19] - improve SOM-CPC performance as well.

References

[1] R. B. Berry, R. Brooks, C. E. Gamaldo, S. M. Harding, C. Marcus, B. V. Vaughn, et al. The AASM manual for the scoring of sleep and associated events. *Rules, Terminology and Technical Specifications, Darien, Illinois, American Academy of Sleep Medicine*, 176:2012, 2012.

[2] P. H. M. Braga, H. R. Medeiros, and H. F. Bassani. Deep categorization with semi-supervised self-organizing maps. In *International Joint Conference on Neural Networks (IJCNN)*, 2020. ISBN 9781728169262.

[3] S. Chambon, M. N. Galtier, P. J. Arnal, G. Wainrib, and A. Gramfort. A Deep Learning Architecture for Temporal Sleep Stage Classification Using Multivariate and Multimodal Time Series. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 26(4):758–769, 2018. ISSN 15344320. doi: 10.1109/TNSRE.2018.2813138.

[4] D. Christopher. Manning, introduction to information retrieval. *Journal of the American Society for Information Science and Technology*, 43(3):824–825, 2008.

[5] J. Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20 (1):37–46, 1960.

[6] C. Ferles, Y. Papanikolaou, and K. J. Naidoo. Denoising Autoencoder Self-Organizing Map (DASOM). *Neural Networks*, 105:112–131, 9 2018. ISSN 18792782. doi: 10.1016/j.neunet.2018.04.016.

[7] F. Forest, M. Lebbah, H. Azzag, and J. Lacaille. Deep Architectures for Joint Clustering and Visualization with Self-organizing Maps. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, volume 11607 LNAI, pages 105–116. Springer Verlag, 2019. ISBN 9783030261412. doi: 10.1007/978-3-030-26142-9{_}10.

[8] F. Forest, M. Lebbah, H. Azzag, and J. Lacaille. A survey and implementation of performance metrics for self-organized maps. *arXiv preprint arXiv:2011.05847*, 2020.

[9] F. Forest, M. Lebbah, H. Azzag, and J. Lacaille. Deep embedded self-organizing maps for joint representation learning and topology-preserving clustering. *Neural Computing and Applications*, 33 (24):17439–17469, 12 2021. ISSN 14333058. doi: 10.1007/s00521-021-06331-w.

[10] V. Fortuin, M. Hüser, F. Locatello, H. Strathmann, and G. Rätsch. Som-Vae: Interpretable discrete representation learning on time series. *7th International Conference on Learning Representations, ICLR 2019*, pages 1–18, 2019.

[11] A. Hyvärinen and P. Pajunen. Nonlinear independent component analysis: Existence and uniqueness results. *Neural networks*, 12(3):429–439, 1999.

[12] A. Hyvarinen, H. Sasaki, and R. E. Turner. Nonlinear ICA Using Auxiliary Variables and Generalized Contrastive Learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, volume 89, pages 859–868, 2018. URL `http://arxiv.org/abs/1805.08651`.

[13] L. Khacef, L. Rodriguez, and B. Miramond. Improving Self-Organizing Maps with Unsupervised Feature Extraction. In *In International Conference on Neural Information Processing*, pages 474–486, 9 2020. URL `http://arxiv.org/abs/2009.02174`.

[14] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[16] K. Kiviluoto. Topology preservation in self-organizing maps. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 1, pages 294–299. IEEE, 1996.

[17] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[18] P. H. Le-Khac, G. Healy, and A. F. Smeaton. Contrastive representation learning: A framework and review. *IEEE Access*, 2020.

[19] L. Manduchi, M. Hüser, M. Faltys, J. Vogt, G. Rätsch, and V. Fortuin. *T-DPSOM: An interpretable clustering method for unsupervised learning of patient health states*, volume 1. Association for Computing Machinery, 2021. ISBN 9781450383592. doi: 10.1145/3450439.3451872.

[20] N. Mrabah, M. Bouguessa, and R. Ksantini. Adversarial deep embedded clustering: on a better trade-off between feature randomness and feature drift. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[21] A. v. d. Oord, Y. Li, and O. Vinyals. Representation Learning with Contrastive Predictive Coding. *arXiv preprint arXiv:1807.03748*, 2019. URL `http://arxiv.org/abs/1807.03748`.

[22] C. O'reilly, N. Gosselin, J. Carrier, and T. Nielsen. Montreal archive of sleep studies: an open-access resource for instrument benchmarking and exploratory research. *Journal of sleep research*, 23(6):628–635, 2014.

[23] C. O'Reilly, N. Gosselin, J. Carrier, and T. Nielsen. Montreal Archive of Sleep Studies: an open-access resource for instrument benchmarking and exploratory research. *Journal of Sleep Research*, 23(6):628–635, 12 2014. ISSN 09621105. doi: 10.1111/jsr.12169. URL `http://doi.wiley.com/10.1111/jsr.12169`.

[24] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[26] M. Pesteie, P. Abolmaesumi, and R. Rohling. Deep neural maps. *arXiv preprint arXiv:1810.07291*, 2018.

[27] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural Discrete Representation Learning. In *Conference on Neural Information Processing System*, 2017.

[28] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR, 2016.

[29] R. S. Zimmermann, Y. Sharma, S. Schneider, M. Bethge, and W. Brendel. Contrastive Learning Inverts the Data Generating Process. *arxiv*, 2021. URL `http://arxiv.org/abs/2102.08850`.

A Experimental details

This appendix contains all necessary information for full reproducability of the experiments. General details are provided in appendix A.1, while experiment-specific settings are discussed in the subsequent sections.

A.1 General details

In the experiments section, we investigated the effect of various differences as mentioned in section 2.2 by means of ablation studies. For this purpose, models were run that use a Gaussian neighbourhood function, while disabling gradient updates of the topological loss from the neighbours of the winning node to the encoder, equivalent as in the SOM-VAE model [10]. In this case, the same split in a SOM and a commitment loss is made as given in eq. (7) and eq. (8), but now with the neighbourhood function $\mathcal{S}$ being a Gaussian.

To restrict the search space of hyperparameters in the SOM-VAE(-prob) model, which has many loss multipliers, we fixed $\beta/\alpha = 1/5$. Multiplier β scales the SOM loss that sums the quantization error of the four neighbour nodes in the plus kernel. It is, therefore, expected to be at least 4 times larger than the commitment loss, which only reflects the quantization error of the winning node. Choosing $\beta = \alpha/4$ would imply that the weighing of the summed quantization error of the four neighbours is equal to the weighing of this error for the winning node. To give the winning node a slightly higher importance, we set $\beta = \alpha/5$.

We have used several metrics to quantify SOM performance, as explained in section 3.3. The purity implementation is taken from the Github page of Fortuin et al [10], while NMI implementation was taken from the sklearn library [25]. The topographic error implementation comes from the *SOMperf* python library [8]. Finally, the $\ell_{2,\text{smooth}}$ distance is computed using the *norm* function in the *linalg* library of numpy.

Pseudocode of SOM-CPC, when considering the presence of an AR module, is provided in algorithm 1.

Algorithm 1 SOM-CPC

```
Input: Dataset $\mathcal{X}$, model comprising $f_\theta$, $g_\psi$, $\{\mathbf{W}_p\}_{p=1}^{P}$, number of past windows $L$, # positive samples $P$, # negative samples $N$, # SOM nodes $k$, Gaussian neighbourhood function $\mathcal{S}$ with $\sigma^{(n_{\max})}$, $n_{\max}$, loss trade-off parameter $\alpha$.
Output: A trained SOM: topologically ordered codebook $\Phi$ that represents data $\mathcal{X}$ in 2D.
  for $n$ in $n_{\max}$ do
    - Sample a sequence of datapoints: $[\boldsymbol{x}(t-L), \ldots, x(t)] \sim \mathcal{X}$
    - Define $P$ positive samples: $\{\boldsymbol{x}(t+p)\}_{p=1}^{P}$
    - Sample $P \times N$ negative samples $\mathcal{X}' \subset \mathcal{X}$, with $|\mathcal{X}'| = P \times N$
    - Encode
        - data sequence: $\boldsymbol{c}(t) = g_\psi\Big(f_\theta\big([\boldsymbol{x}(t-L), \ldots, x(t)]\big)\Big)$
        - positive samples independently: $\{\boldsymbol{z}(t+p)\}_{p=1}^{P} = f_\theta\big(\{\boldsymbol{x}(t+p)\}_{p=1}^{P}\big)$
        - negative samples independently: $\mathcal{Z}' = f_\theta\big(\mathcal{X}'\big)$
    - Predict future: $\hat{\mathbf{z}}(t+p) = \mathbf{W}_p\boldsymbol{c}(t)$, with $1 \le p \le P$
    - Quantize: $\phi_j = \text{SOM}_\Phi\left(\boldsymbol{c}(t)\right)$
    - Compute $\sigma^{(n)}$ according to eq. (4)
    - Compute $\mathcal{L}_{\text{topo}}\big(\boldsymbol{c}(t)\big)$ and $\mathcal{L}_{\text{infoNCE}}$ according to eq. (6) and eq. (9)
    - Update trainable parameters $\propto \alpha\mathcal{L}_{\text{topo}}$ and $\mathcal{L}_{\text{infoNCE}}$
  end for
```

A.2 Synthetic experiments

A.2.1 Architecture details

Table 2 summarizes the encoder and decoder architectures used in this experiment. The *output size* column in the table uses channels-first notation. The SOM-VAE and DESOM models were found to benefit from a convolutional part of the encoder that did not fully reduce the temporal dimension to size 1. As such, the last convolutional layer of the SOM-CPC encoder was changed for a fully connected layer preceded by a flattening operation for the autoencoder-based models. The SOM-CPC model is run without an AR module, to make the fairest comparison to the SOM-VAE(prob) and DESOM models, which also do not incorporate such a component.

Layer type	Output size	Channels	Activation	Kernel size	Strides	Dilation	Padding
Encoder for SOM-CPC and CPC							
Conv1D	bs $\times 16 \times 128$	16	Leaky ReLU (0.01)	9	1	1	same
MaxPool1D	bs $\times 16 \times 32$	-	-	4	4	-	-
Dropout (0.1)	bs $\times 16 \times 32$	-	-	-	-	-	-
Conv1D	bs $\times 32 \times 32$	32	Leaky ReLU (0.01)	7	1	1	same
MaxPool1D	bs $\times 32 \times 8$	-	-	4	4	-	-
Dropout (0.1)	bs $\times 32 \times 8$	-	-	-	-	-	-
Conv1D	bs $\times 64 \times 8$	64	Leaky ReLU (0.01)	3	1	1	same
MaxPool1D	bs $\times 64 \times 2$	-	-	4	4	-	-
Dropout (0.1)	bs $\times 64 \times 2$	-	-	-	-	-	-
Conv1D	bs $\times 128 \times 2$	128	Leaky ReLU (0.01)	3	1	1	same
MaxPool1D	bs $\times 128 \times 1$	-	-	2	2	-	-
Encoder for SOM-VAE and DESOM							
Conv1D	bs $\times 16 \times 128$	16	Leaky ReLU (0.01)	9	1	1	same
MaxPool1D	bs $\times 16 \times 32$	-	-	4	4	-	-
Dropout (0.1)	bs $\times 16 \times 32$	-	-	-	-	-	-
Conv1D	bs $\times 32 \times 32$	32	Leaky ReLU (0.01)	7	1	1	same
MaxPool1D	bs $\times 32 \times 8$	-	-	4	4	-	-
Dropout (0.1)	bs $\times 32 \times 8$	-	-	-	-	-	-
Conv1D	bs $\times 64 \times 8$	64	Leaky ReLU (0.01)	3	1	1	same
Flatten	bs $\times 512$	-	-	-	-	-	-
Fully Connected	bs $\times 128$	128	Leaky ReLU (0.01)	-	-	-	-
Decoder for SOM-VAE and DESOM							
Fully Connected	bs $\times 512$	512	Leaky ReLU (0.01)	-	-	-	-
Unflatten	bs $\times 64 \times 8$	-	-	-	-	-	-
Conv1D	bs $\times 32 \times 8$	32	Leaky ReLU (0.01)	3	1	1	same
ConvTranspose1D	bs $\times 32 \times 32$	32	None	4	4	1	0
Conv1D	bs $\times 16 \times 32$	16	Leaky ReLU (0.01)	7	1	1	same
ConvTranspose1D	bs $\times 16 \times 128$	16	None	4	4	1	0
Conv1D	bs $\times 1 \times 128$	1	Tanh	9	1	1	same

Table 2: Model details for the synthetic experiments in section 4.1.

A.2.2 Training details

For the SOM-CPC model, $P = 3$ future predictions (i.e. positive samples) were used, and $N = 3$ negative samples were drawn for each positive sample. The latter were drawn randomly from the entire training set. We used the Adam optimizer [14], with a learning rate of 0.001 and a batch size of 128. Each model was trained for maximally 1000 epochs. The best model was selected based on the lowest task loss, being $\mathcal{L}_{\text{recon}}$ for SOM-VAE and DESOM and $\mathcal{L}_{\text{InfoNCE}}$ for SOM-CPC, on the validation set. We did not use the full training objective $\mathcal{L}_{\text{deep-SOM}}$ as model selection criterion, as both the commitment and SOM loss showed to be low initially (possibly due to low values of the random initialization of the model), while both increased and reached a steady-state later in training. The (non-activated) linear classifier on the CPC embeddings was trained until convergence, which happened after 550 epochs.

The value of $\sigma^{(n_{\max})}$ for the Gaussian neighbourhood kernel was set to 2. Choosing a lower value at the end of training induced instable optimization behavior for this synthethic case.

A.2.3 Extended results

Table 3 shows the results of different deep-SOM models on our synthetic case. Additional to table 1 is the loss multiplier sweep on the SOM-VAE-prob model [10]. This model is expected to show more smooth trajectories over time, captured in the $\ell_{2,\text{smooth}}$ distance metric, thanks to the additional transition and smoothness loss (multiplier by γ and τ, respectively). It can be seen that tuning these hyperparameters is a complex process, and even a sweep did not result in one run that performed better. In contrary, the addition of the extra losses possibly interfered with the optimization process, and only very delicate settings of γ and τ might improve model performance eventually.

Figure 5 shows the optimization behavior of SOM-VAE (with Gaussian neighbourhood) versus SOM-CPC. The latter shows an optimization trajectory in which both task and topological loss seem to be more in line than for the SOM-VAE model.

$\boldsymbol{\alpha}$	$\mathcal{S}$	$\mathcal{L}_{\text{SOM}}$ sg$[\cdot]$	SE_{target}	$\ell_{2,\textbf{smooth}}$	**TE**
CPC + supervised classifier [21]					
-	-	-	2.62	-	-
SOM-VAE (with $\beta = \alpha/5$) [10]					
1e-3	Plus	✓	11.59	2.60±.46	.38
1e-2	Plus	✓	14.57	2.98±.84	.38
* .1	Plus	✓	**8.02**	**2.41±.68**	**.28**
1	Plus	✓	13.43	2.75±.45	.33
1e-5	Gaussian	✓	11.12	1.93±.29	.069
1e-4	Gaussian	✓	**10.52**	1.95±.36	.075
1e-3	Gaussian	✓	11.60	**1.92±.34**	**.056**
1e-2	Gaussian	✓	18.13	2.03±.42	.086
SOM-VAE-prob (with $\beta = \alpha/5$) [10]					
1e-3 (γ = 5e-5, τ = 1e-3)	Plus	✓	21.26	3.78±.52	.93
1e-3 (γ = 4e-5, τ = 1e-3)	Plus	✓	21.30	4.23±.57	.88
1e-3 (γ = 3.3e-5, τ = 1e-3)	Plus	✓	22.52	3.81±.51	.98
1e-2 (γ = 5e-4, τ = 1e-2)	Plus	✓	**14.65**	3.70±.51	.86
1e-2 (γ = 4e-4, τ = 1e-2)	Plus	✓	27.25	3.54±.67	.94
1e-2 (γ = 3.3e-4, τ = 1e-2)	Plus	✓	21.62	3.71±.76	.97
.1 (γ = 5e-5, τ = 1e-2)	Plus	✓	26.82	3.23±.80	.68
.1 (γ = 4e-5, τ = 1e-2)	Plus	✓	22.34	3.11±.73	.74
.1 (γ = 3.3e-5, τ = 1e-2)	Plus	✓	20.10	3.15±.73	**.63**
.1 (γ = 5e-4, τ = .1)	Plus	✓	40.85	3.06±.55	.84
.1 (γ = 4e-4, τ = .1)	Plus	✓	29.96	**2.81±.34**	.82
.1 (γ = 3.3e-4, τ = .1)	Plus	✓	30.96	3.95±.83	.85
DESOM [9]					
1e-5	Gaussian	✗	14.00	1.99±.36	.13
1e-4	Gaussian	✗	19.09	1.95±.28	.11
1e-3	Gaussian	✗	12.58	1.92±.31	.077
1e-2	Gaussian	✗	13.66	**1.89±.33**	.065
* .1	Gaussian	✗	**10.77**	2.20±.44	**.061**
1	Gaussian	✗	39.4	2.19±.33	.067
SOM-CPC (ours)					
1e-5	Gaussian	✗	.95	1.24±.31	.048
1e-4	Gaussian	✗	.72	1.37±.37	**.022**
* 1e-3	Gaussian	✗	.81	**1.06±.28**	.028
1e-2	Gaussian	✗	**.62**	1.08±.28	.059
.1	Gaussian	✗	1.90	1.18±.30	.039
1e-5	Gaussian	✓	.64	1.04±.26	**.039**
1e-4	Gaussian	✓	.68	1.19±.43	.057
1e-3	Gaussian	✓	.75	1.12±.32	.059
1e-2	Gaussian	✓	**.47**	**.99±.24**	.069
.1	Gaussian	✓	.89	1.16±.32	.069
1e-3	Plus	✗	1.36	2.27±.44	.22
1e-2	Plus	✓	1.16	1.85±.26	.12

Table 3: This is the extended version of table 1 on the synthetic data results, with a multiplier sweep of the SOM-VAE-prob model included.

Figure 6 shows PCA projections of the latent space of the SOM-VAE, DESOM, and SOM-CPC models that were indicated with a * in table 1, and for which the SOMs were visualized in fig. 2. Disentanglement of the signal frequencies is much better for the SOM-CPC model, providing an explanation for the better (i.e. lower) SE_{target} of this model, as compared to SOM-VAE and DESOM.

A.3 Sleep experiments

A.3.1 Data processing

This set contains whole-night polysomnography recordings - including among others electroencephalography (EEG), chin electromyography (EMG), and electrooculography (EOG) - of 62 healthy sleepers (we refer the reader to [22] for more details regarding this dataset). We selected the channels that are typically used in clinical practice, comprising three EEG channels (F4, C4, O2), the two EOG channels, and one chin EMG derivation, and downsampled the data to 128 Hz. Sleep stage labels that follow the guidelines of the American Academy of Sleep Medicine (AASM) [1] (being wakefulness (W), rapid-eye movement (REM) sleep, or non-REM1 till non-REM3 (N1, N2, N3)) were available for every non-overlapping 30-second window, the common label resolution in clinical practice.

Some processing had already been done by the distributors of the MASS dataset [23]. The 60 Hz powerline interference was, however, not fully suppressed, and we wanted to down sample each signal to 128 Hz to reduce computational complexity. As such, before downsampling, all derivations were additionally filtered with a zero-phase (i.e. two-directional) 5th order Butterworth band-pass filter ($0.3 - 59$ Hz), followed by another zero-phase 5th order Butterworth notch filter ($59 - 61$ Hz). Channels were normalized within-patient and per channel, yielding mean subtraction, followed by

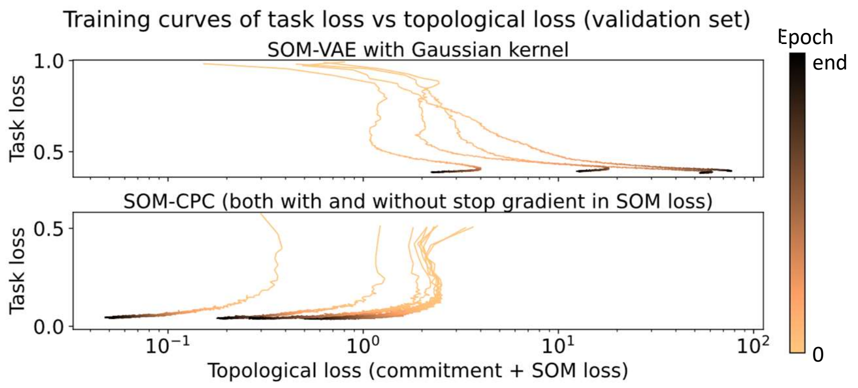

Figure 5: Task loss versus the topological loss for the SOM-VAE model with Gaussian neighbourhood kernel and the SOM-CPC models (both variants; either with or without the SOM loss detached from the encoder). The SOM-CPC models follow a more smooth optimization curve, minimizing both the task and topological loss, while these losses seem to be more conflicting in the SOM-VAE model. The different curves display various values of trade-off parameter α, for which the SOM-VAE model seems more sensitive.

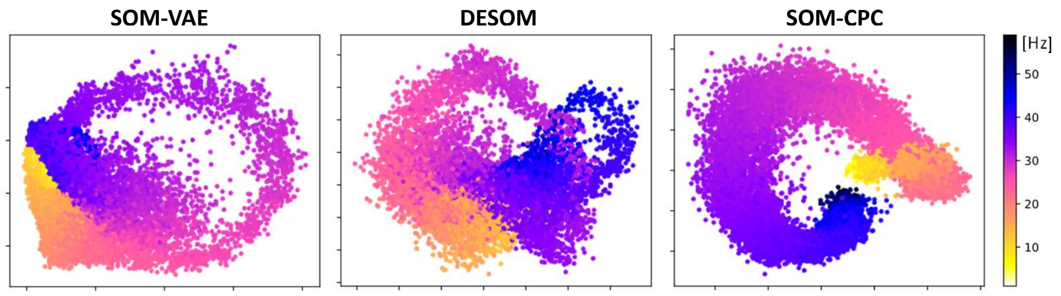

Figure 6: PCA projections of the latent spaces of the SOM-VAE, DESOM, and SOM-CPC models that were also visualized in fig. 2. Disentanglement of the signal frequencies is much better in the SOM-CPC model.

normalization such that amplitudes of 95% of the samples were mapped between -1 and +1. The 62 recordings (numbered $1-64$, with number 43 and 49 missing) were split into a training set including patients $1-48$ ($n=47$), a validation set including patients $50-57$ ($n=8$), and hold-out test set that included patients $58-64$ ($n=7$).

A.3.2 Architecture details

Table 4 summarizes the used encoder and decoder architectures. The encoder architecture is based upon an earlier sleep staging model [3]. The SOM-VAE and DESOM benchmark models were, however, found to benefit from a higher-capacity encoder, with twice as many channels (16 instead of 8), and no dimensionality reduction in the last fully connected layer before the latent space. As a result, the feature vectors used in the SOM-CPC model were of size $F=128$, while these were of size $F=1440$ for the benchmark models. This altered dimensionality also induced larger codebook vectors in the latter case. The decoder architecture (see table 4) was used both for the continuous and discrete decoding in the SOM-VAE model. No AR-component was used in the SOM-CPC model to make a fair comparison to the SOM-VAE and DESOM model that also did not include such a component.

A.3.3 Training details

For the SOM-CPC model, $P=3$ future predictions (i.e. positive samples) were used, and $N=3$ negative samples were drawn for each positive sample. The latter were drawn from the same subject as the positive sample.

The σ of the Gaussian neighbourhood kernel was annealed to $\sigma^{(n_{\max})} = 0.1$ during training.

For SOM-CPC and the benchmarks, we all used the Adam optimizer [14], with a learning rate of 0.0001 and a batch size of 64. Each model was trained for maximally 500 epochs, and the best model was selected based on the lowest $\mathcal{L}_{\text{recon}}$ (for SOM-VAE and convDESOM) or $\mathcal{L}_{\text{InfoNCE}}$ (for SOM-CPC) on the validation set. The softmax-activated linear classifier trained on the CPC embeddings was trained with a batch size of 128, and was stopped upon convergence, which happened after 250 epochs.

Layer type	Output size	Channels	Activation	Kernel size	Strides	Dilation	Padding
Encoder for SOM-CPC and CPC							
Unsqueeze (dim=-1)	bs $\times 1 \times 6 \times 3840$	-	-	-	-	-	-
Conv2D	bs $\times 6 \times 1 \times 3840$	6	None	$(6, 1)$	$(1, 1)$	1	0
Permute $(0, 2, 1, 3)$	bs $\times 1 \times 6 \times 3840$	-	-	-	-	-	-
Conv2D	bs $\times 8 \times 6 \times 3840$	8	ReLU	$(1, 64)$	$(1, 1)$	1	same
MaxPool2D	bs $\times 8 \times 6 \times 240$	-	-	$(1, 16)$	$(1, 16)$	-	-
Conv 2D	bs $\times 8 \times 6 \times 240$	8	ReLU	$(1, 64)$	$(1, 1)$	1	same
MaxPool2D	bs $\times 8 \times 6 \times 15$	-	-	$(1, 16)$	$(1, 16)$	-	-
Flatten	bs $\times 1440$	-	-	-	-	-	-
Dropout (0.5)	bs $\times 1440$	-	-	-	-	-	-
Fully Connected	bs $\times 128$	128	None	-	-	-	-
Encoder for SOM-VAE and DESOM							
Unsqueeze (dim=-1)	bs $\times 1 \times 6 \times 3840$	-	-	-	-	-	-
Conv2D	bs $\times 6 \times 1 \times 3840$	6	None	$(6, 1)$	$(1, 1)$	1	0
Permute $(0, 2, 1, 3)$	bs $\times 1 \times 6 \times 3840$	-	-	-	-	-	-
Conv2D	bs $\times 16 \times 6 \times 3840$	16	ReLU	$(1, 64)$	$(1, 1)$	1	same
MaxPool2D	bs $\times 16 \times 6 \times 240$	-	-	$(1, 16)$	$(1, 16)$	-	-
Conv 2D	bs $\times 16 \times 6 \times 240$	16	ReLU	$(1, 64)$	(1,1)	1	same
MaxPool2D	bs $\times 16 \times 6 \times 15$	-	-	$(1, 16)$	$(1, 16)$	-	-
Flatten	bs $\times 1440$	-	-	-	-	-	-
Dropout (0.5)	bs $\times 1440$	-	-	-	-	-	-
Fully Connected	bs $\times 1440$	1440	None	-	-	-	-
Decoder for SOM-VAE and DESOM							
Fully Connected	bs $\times 1440$	1440	None	-	-	-	-
Unflatten	bs $\times 16 \times 6 \times 15$	-	-	-	-	-	-
ConvTranspose2D	bs $\times 16 \times 6 \times 240$	16	None	$(1, 16)$	$(1, 16)$	1	0
Conv2D	bs $\times 16 \times 6 \times 240$	16	ReLU	$(1, 64)$	$(1, 1)$	1	same
ConvTranspose2D	bs $\times 16 \times 6 \times 240$	16	None	$(1, 16)$	$(1, 16)$	1	0
Conv2D	bs $\times 1 \times 6 \times 3840$	1	ReLU	$(1, 64)$	$(1, 1)$	1	same
Permute $(0, 2, 1, 3)$	bs $\times 6 \times 1 \times 3840$	-	-	-	-	-	-
ConvTranspose2D	bs $\times 6 \times 6 \times 3840$	6	None	$(6, 1)$	$(6, 1)$	1	0
Conv2D	bs $\times 1 \times 6 \times 3840$	1	None	$(6, 1)$	$(1, 1)$	1	same
Squeeze (dim=1)	bs $\times 6 \times 3840$	-	-	-	-	-	-

Table 4: Model details for the sleep experiments in section 4.2.

A.3.4 Extended Results

The extended results of the sleep experiments can be found in table table 5.

A.4 Audio experiments

A.4.1 Architecture details

Table 6 provides the model details of the encoder, and the adopted decoder for the DESOM benchmark. The reconstruction loss of the DESOM model was found to stay constant when using the encoder architecture, as adopted for the SOM-CPC model. As such, the downsampling factor of the encoder was reduced for the DESOM model. For training, we followed the settings from the CPC framework [21] and set $P = 12$. The number of negative samples was set to $N = 10$, which were drawn randomly from the entire training set. A GRU aggregated $L = 127$ past windows with the current window to acquire the context vector, as used in both the SOM-CPC model and the GRU-DESOM benchmarks.

We used the Adam optimizer [14], with a learning rate of 0.0001 and a batch size of 8. Each model was trained for maximally 3000 epochs, and the best model was selected based on the lowest validation task loss. The softmax-activated linear classifier on the CPC embeddings was trained until convergence, which happened after 200 epochs, and a batch size of 128.

A.4.2 Extended Results

The extended results of the audio experiments can be found in table 7.

α	$\mathcal{S}$	$\mathcal{L}_{\text{SOM}}$ $\text{sg}[\cdot]$	**Purity**	**NMI**	**Cohen's kappa**	$\ell_{2,\textbf{smooth}}$	**TE**
CPC + supervised classifier [21]							
-	-	-	-	-	.68	-	-
SOM-VAE (with $\beta = \alpha/5$) [10]							
* 1e-3	Plus	✓	.64	.14	.39	4.38±.20	**.47**
1e-2	Plus	✓	**.65**	**.15**	**.43**	4.48±.26	.50
.1	Plus	✓	.63	.14	.37	4.39±.18	**.47**
1	Plus	✓	.58	.11	.25	**4.31±.21**	.54
DESOM [7]							
1e-6	Gaussian	✗	.52	.04	.04	**.06±.02**	**.11**
1e-5	Gaussian	✗	.62	**.18**	.35	1.78±.16	.32
1e-4	Gaussian	✗	**.64**	.14	**.40**	3.93±.24	.58
1e-3	Gaussian	✗	**.64**	.14	.39	4.12±.16	.54
1e-2	Gaussian	✗	.63	.15	.36	3.68±.30	.87
SOM-CPC (ours)							
1e-5	Gaussian	✗	**.79**	**.28**	.64	**1.18±.16**	.060
1e-4	Gaussian	✗	**.79**	**.28**	**.65**	1.22±.23	.053
* 1e-3	Gaussian	✗	**.79**	**.28**	**.65**	1.24±.19	**.039**
1e-2	Gaussian	✗	.78	.27	.60	1.29±.21	.32
.1	Gaussian	✗	.77	.27	.61	1.40±.10	.64
1e-3	Gaussian	✓	**.79**	**.28**	**.67**	1.22±.21	.042
1e-2	Gaussian	✓	**.79**	**.28**	.65	1.22±.23	.051
.1	Gaussian	✓	.78	.27	.65	**1.18±.18**	**.023**
1	Gaussian	✓	**.79**	**.28**	.63	1.23±.14	.28
1e-3	Plus	✗	.79	.28	.63	1.97±.37	.31
.1	Plus	✓	.79	.27	.63	1.95±.34	.30

Table 5: Test set performance of different deep-SOM models trained on sleep recordings. The SOM-CPC model outperforms the autoencoder-based models on all metrics. SOMs of models with a * are visualized in fig. 3.

Layer type	**Output size**	**Channels**	**Activation**	**Kernel size**	**Strides**	**Dilation**	**Padding**
Encoder for SOM-CPC and CPC							
Conv1D	bs $\times$ 512 $\times$ 32	512	ReLU	10	5	1	3
Conv1D	bs $\times$ 512 $\times$ 8	512	ReLU	8	4	1	2
Conv1D	bs $\times$ 512 $\times$ 4	512	ReLU	4	2	1	1
Conv1D	bs $\times$ 512 $\times$ 2	512	ReLU	4	2	1	1
Conv1D	bs $\times$ 512	512	ReLU	4	2	1	1
Encoder for SOM-VAE and DESOM							
Conv1D	bs $\times$ 512 $\times$ 32	512	ReLU	10	5	1	3
Conv1D	bs $\times$ 512 $\times$ 8	512	ReLU	8	4	1	2
Conv1D	bs $\times$ 512 $\times$ 4	512	ReLU	4	2	1	1
Conv1D	bs $\times$ 512 $\times$ 2	512	ReLU	4	2	1	1
Conv1D	bs $\times$ 512 $\times$ 2	512	ReLU	4	1	1	same
Flatten	bs $\times$1024	-	-	-	-	-	-
Decoder for SOM-VAE and DESOM							
Unflatten	bs $\times$ 512 $\times$ 2	-	-	-	-	-	-
Conv1D	bs $\times$ 512 $\times$ 2	512	ReLU	4	1	1	same
ConvTranspose1D	bs $\times$ 512 $\times$ 4	512	ReLU	4	2	1	1
ConvTranspose1D	bs $\times$ 512 $\times$ 8	512	ReLU	4	2	1	1
ConvTranspose1D	bs $\times$ 512 $\times$ 32	512	ReLU	8	4	1	2
ConvTranspose1D	bs $\times$ 1 $\times$ 160	512	ReLU	10	5	1	3 (+ output pad = 1)

Table 6: Model details for the audio experiments in section 4.3.

α	$\mathcal{S}$	$\mathcal{L}_{\text{SOM}}$ $\text{sg}[\cdot]$	**Purity**	**NMI**	**Cohen's kappa**	**TE**
CPC + supervised classifier [21]						
-	-	-	-	-	1.00	-
DESOM [9]						
1e-5	Gaussian	✗	.18	.03	.06	**.14**
1e-4	Gaussian	✗	.23	.08	.11	.34
1e-3	Gaussian	✗	**.31**	**.13**	**.20**	.68
1e-2	Gaussian	✗	.13	.00	.00	1.0
GRU-DESOM (reconstructing last window) [7]						
1e-5	Gaussian	✗	.19	.04	.08	**.13**
1e-4	Gaussian	✗	.26	.09	.15	.20
1e-3	Gaussian	✗	.31	.13	.21	.42
1e-2	Gaussian	✗	**.32**	**.14**	**.22**	.46
0.1	Gaussian	✗	.13	.00	.00	.91
GRU-DESOM (reconstructing full sequence) [7]						
1e-5	Gaussian	✗	.19	.05	.08	**.34**
1e-4	Gaussian	✗	.30	.12	.20	.59
* 1e-3	Gaussian	✗	**.33**	**.14**	**.22**	.78
1e-2	Gaussian	✗	.29	.12	.19	.58
SOM-CPC (ours)						
1e-5	Gaussian	✗	.99	**.73**	.99	**.15**
1e-4	Gaussian	✗	**1.00**	.63	**1.00**	.24
* 1e-3	Gaussian	✗	**1.00**	.61	**1.00**	.33
1e-2	Gaussian	✗	**1.00**	.61	.99	.34
.1	Gaussian	✗	**1.00**	.61	.99	.42
1	Gaussian	✗	**1.00**	.60	.99	.45
1e-3	Gaussian	✓	**1.00**	**.61**	.99	**.28**
1e-2	Gaussian	✓	**1.00**	**.61**	.99	.35
.1	Gaussian	✓	**1.00**	**.61**	**1.00**	.35
1	Gaussian	✓	**1.00**	**.61**	**1.00**	.38

Table 7: Clustering, classification and topographic metrics on the test set for the audio experiments with different variants of the DESOM model and SOM-CPC. SOM-CPC has superior performance. SOMs of models with a * are visualized in fig. 4.