Online Unsupervised Representation Learning of Waveforms in the Intensive Care Unit via a novel cooperative framework: Spatially Resolved Temporal Networks (SpaRTEn)

Abstract

Univariate high-frequency time series are dominant data sources for many medical, economic and environmental applications. In many of these domains, the time series are tied to real-time changes in state. In the intensive care unit, for example, changes and intracranial pressure waveforms can indicate whether a patient is developing decreased blood perfusion to the brain during a stroke, for example. However, most representation learning to resolve states is conducted in an offline, batch-dependent manner. In high frequency time-series, high intra-state and inter-sample variability makes offline, batch-dependent learning a relatively difficult task. Hence, we propose Spatial Resolved Temporal Networks (SpaRTeN), a novel composite deep learning model for online, unsupervised representation learning through a spatially constrained latent space. SpaRTeN maps waveforms to states, and learns time-dependent representations of each state. Our key contribution is that we generate clinically relevant representations of each state for intracranial pressure waveforms.

1. Introduction

In high-frequency time series data like intracranial pressure waveforms, rapidly predicting and detecting changes in state is a clinically important task. For example, if a patient in the intensive care unit starts exhibiting intracranial pressure decompensation, it may cause bilateral blindness (Mollan et al. (2016)). Consequently, early detection of state transitions may provide clinicians with the tools to intervene appropriately for better outcomes. For example, at early stages, cerebral vascular decompensation can be treated with a loop diuretic like furosemide (Llwyd et al. (2022)).

Second, a growing amount of research is indicating the need to redefine critical illness by biological state rather than a non-specific illness syndrome (Maslove et al. (2022)). Large scale cohort studies, enabled by the sheer volume of patients in the intensive care unit during the severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) pandemic—suggests that the current syndrome-based framework of critical illness should be reconsidered. Large variability in patient responses are inadequately characterized by over-simplified diagnostic codes such as ”acute respiratory distress syndrome”. More precise subphenotyping of underlying pathophysiology via data-driven clustering algorithms may inform more precise interventions at the bedside. SpaRTEn takes a key step towards personalized care in the intensive care unit by generating individualized state representations in real-time. Identifying the clinical correlates of these state representations enables an interpretable understanding of waveforms, which can propagate critical care research, and eventually allow for targeted therapies in the intensive care unit.

Many algorithms like shapelets, hierarchical latent factor models, hidden Markov Model-like methods, change point and anomaly detection techniques, and N-Beats are dedicated towards disentangling time series into their respective subcomponents (Li et al. (2021); Grabocka et al. (2015); Oreshkin et al. (2019a); Blazquez-Garcia and Conde (2022); Aminikhanghahi and Cook (2017); Van Den Oord and Vinyals (2017)) but few are dedicated towards disentangling states within a single time series (Franceschi et al. (2019)) or predicting future state transitions. For high-dimensional datasets, unsupervised methods like t-SNE, UMAP, and SOMs can be used to project samples into lower dimensions with spatial relationships (Van der Maaten and Hinton (2008); McInnes et al. (2018)). However, in time series, dimensionality is proportional to series length. As a result, state determination requires encoding time series into fixed-length vectors, followed by clustering algorithms like k-means. These methods can capture long-range dependencies but rely on non-differentiable function fitting.

Also, these methods are often offline, in that they learn from an entire training dataset at once, before being evaluated and deployed. This can be problematic, especially in the context of dataset shift or high inter-sample variability. Every time a new batch of data is received, the entire model needs to be retrained. High-frequency time series data like waveforms are often encountered in scenarios more suitable for online learning, wherein a learner attempts to tackle some predictive task by learning a sequence of data in the order they are received (Hoi et al. (2021)).

Extraction of states or state-transitions from a high-frequency time series requires online unsupervised representation learning, a relatively understudied field. Fuzzy neural networks create a set of modifiable rules (Luo et al. (2019)), but successive rule changes makes state inference relatively volatile and inconclusive. Another example of the state-of-the-art time series forecasting method is temporal fusion transformers (TFTs), which can provide interpretable risk prediction via attention mechanisms (Lim et al. (2021); Kamal et al.). This method combines feature attention with sequence attention to generate interpretable forecasts, and has shown great promise in time series forecasting.

In this work, we propose **Spa**tial **R**esolved **Te**mporal **N**etworks (SpaRTeN), a composite differentiable unsupervised deep learning network to learn a discrete spatial representation from a high frequency time series via temporal ensemble learning. We show that our method outperforms SOTA methods such as TFTs with the same number of parameters in benchmarks of online learning tasks. We also note that these other methods can be included within our method due to the flexibility of the composite model.

Generalizable Insights about Machine Learning in the Context of Healthcare

- We introduce SpaRTEn, a new framework for online learning of spatial representations from high-frequency time series.

- We demonstrate that introduction of a latent space improves rather than harms SpaRTEn's ability to forecast and cluster high frequency data in real-time, compared to state of the art models.

- We show that SpaRTeN can generate clinically meaningful representations of medical intracranial pressure waveforms.

2. Related Work

2.1. Twin Neural Networks

Twin neural networks contain two or more identical subnetworks (He et al. (2018)), and can learn semantic similarity between different samples. Subnetworks cast as recurrent neural networks have been used to learn and visualize time series similarities (Pei et al. (2016)). Like twin neural networks, our framework employs contrastive loss with subnetworks, but does not force subnetworks to share all the weights or even architectures.

2.2. Temporal Ensembles and Mixture of Experts

Ensemble Learning refers to a family of techniques where multiple learners are trained to solve the same problem (Zhou (2009)). Ensemble methods construct multiple hypotheses from these base learner algorithms and join them to generate a prediction that generalizes much better than the individual algorithms. Ensembles with base learner LSTMs have been used on financial time series forecasting to improve performance (Sun et al. (2018)). Our framework forces base learners to occupy a Euclidean space, which can subsequently be used to generate interpretable representations. Other online unsupervised methods with time series have developed composite or adaptive model approaches focused on anomaly detection followed by model adaptation (Karaahmetoglu et al. (2020); Savitha et al. (2020)). Having distinct sub-networks for each state allows for different models to uniquely represent distinct states.

The key advantage of utilizing the framework proposed in the paper over a mixture of experts is the idea of state separation,which allows visualization and explainability via representation. Utilizing the novel contrastive function to promote diversity in recurrent neural networks allows for state separation. This is particularly relevant in the medical setting - state separation allows for different interventions. Learning these representations can allow a provider to give a drug or an economist to change a fiscal policy. We provide the specific example of the ICU measurements, where if an individual belongs to a state where cerebral ischemia is identified, then an intervention targeting cerebral ischemia can be provided. Mixtures of expert models do not typically generate representations to interpret and therefore, limits explainability in state-dependent time-series analysis.

2.3. Discrete Latent Spaces

Discrete latent spaces have been utilized in the past with relatively high degrees of success. For example, VAEs can discretize the latent space with an encoder-decoder setup, and has been more heavily applied to interpreted disentangling of discrete representation learning **?**. There are a few key differences between the VAEs and SpaRTEn in terms of 1) generated output, and 2) task flexibility. While VAEs generate a representation in the latent space, SpaRTEn clearly identifies the representation in the space of the time series (Figure 1b). Generating a representation in the same space as the time series allows for improved explainability, and therefore, intervention. For example, if a patient has an ICP waveform that belongs to the state where there is cerebral ischemia, then clinicians can make an intervention relevant to cerebral ischemia. Current VAE based methods generate representations in a latent space, and the relevant clinical state must be extracted from additional data. Second, VAEs are typically constrained to reconstruction or KL-divergence based loss. In their current implementation, they have yet to be implemented for forecasting. Finally, SpaRTEn can take advantage of the diverse potential loss functions for the R-Block and improve individual sub-networks.

One extension of VAEs with a spatially resolved latent space encodes time series in a self-organizing map (Fortuin et al. (2018)). Self-organizing maps are an extension of discrete latent spaces that represents an input space with fixed dimensionality as a discrete two-dimensional Euclidean space. Each node in the two dimensional map is a single neuron, and the best matching neuron is adjusted towards input. This model learns state transitions via Markov modeling on the self-organizing map. We extend self-organizing maps differently, where nodes represent distinct subnetworks rather than a decodable state, which allows distinct weights and architectures. Using a separate block to predict the node, we can eliminate the Markov chain used in SOM-VAEs.

3. Methods

Mathematical notation is precisely defined in (Appendix 6)

3.1. Problem Formulation

For the purposes of time series forecasting, we examine the problem of simultaneously learning:

1. a function $S : \boldsymbol{x_t} \rightarrow \boldsymbol{s_t}$, which maps a time series $\boldsymbol{x_t}$ of length k, $\{x_{t-k}, x_{t-k+1}, \ldots, x_t\}$, to a discrete state $\boldsymbol{s_t}$, and
2. a set of functions $R_{\boldsymbol{s_t}} : \boldsymbol{x_t} \rightarrow \boldsymbol{\hat{y}_t}$ which, for each state $\boldsymbol{s_t}$, map the input time series of length k to a forecast time series $\boldsymbol{\hat{y}_t} = \{x_{t+1}, x_{t+2}, \ldots, x_{t+w}\}$ of length equal to the prediction window w.

S and R are optimized to maximize the probability of assigning the time series $\boldsymbol{x_t}$ to the most suitable function $R_{(.)}$ as determined by an objective function L:

$$\max_{S} \min_{R_{(.)}} E[L(R_{S(\boldsymbol{x_t})}(\boldsymbol{x_t}))]$$

where the expectation $E[L(.)]$ is taken over the set of all time series. In contrast to the minmax framework described for GANs in (Goodfellow et al. (2014)), our networks are collaborative - helping each other out to determine the best state the corresponding best predictions for that state.

3.2. Model architecture and the forward pass

We implement the above with a composite model architecture depicted in Figure 1, where S and R are depicted as analogously named blocks. The height a and width b, common

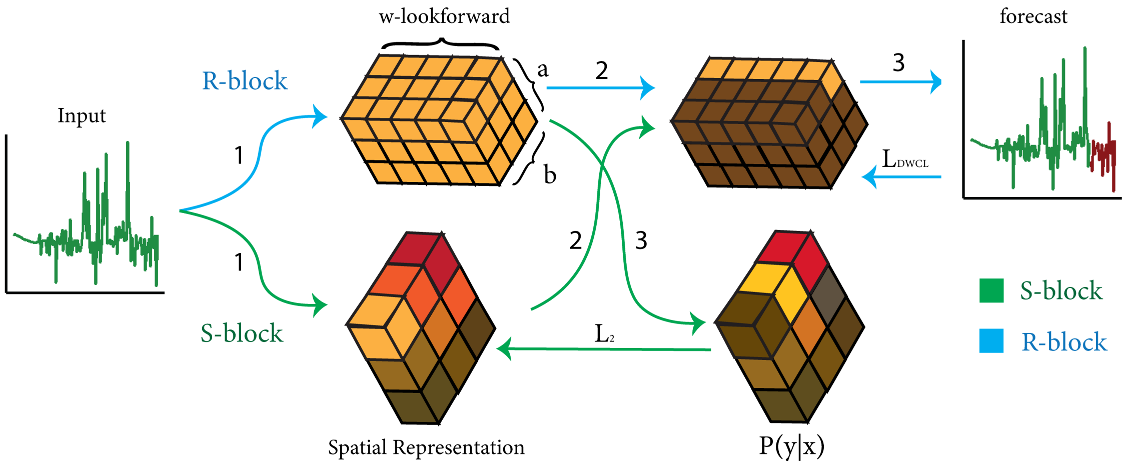


Figure 1: **Schematic representation of model architecture.** Blue is S-Block propagation and green is R-Block. Time series from the data space are forecast by the sub-networks in an R-block with a look-forward period of k. The dimensionality of the spatial representation is a $a \times b$. Simultaneously, an S-Block predicts the most relevant sub-network for the time series. The predicted sub-network is used to generate a forecast and is then back-propagated via Equation 1.

to the two blocks, represents the two dimensional discrete state space, which can also be considered to be the latent space for this model.

The S block implements convolutional filters (Appendix 6) to map an input time series $\boldsymbol{x_t}$ to a density over the discrete two dimensional space of states (green arrow 1). The R block consists of a spatially arranged ensemble of LSTM sub-networks, each of which makes a forecast for the input $\boldsymbol{x_t}$ with a prediction window of w (blue arrow 1). For each input $\boldsymbol{x_t}$, the sub-network in R corresponding to the greatest density output by S (green arrow 2) is used for generating the prediction $\hat{\boldsymbol{y}}_{\boldsymbol{t}}$ (blue arrows 2, 3).

We choose to place $\boldsymbol{s_t}$ in a two-dimensional discrete state-space (i, j), because it facilitates easy visualization of time series corresponding to individual states, which previous methods like SOM-VAE and TFT are unable to currently do. We can parameterize the number of states by adjusting the width and height of the latent space, a and b. The state space width and height are hyper-parameters that should be adjusted depending on the *a priori* assumptions of dataset complexity.

3.3. Loss functions and Training with backpropagation

When implemented as described here, the entire system can be trained with backpropagation. Two distinct loss functions have to be accounted for, for the S and R blocks.

Sub-networks of R can have a primary objective ($L_{\text{objective}}$) of forecasting, classification or reconstruction if we assign $\boldsymbol{y_t}$ to be the forecasting window, a labeled class or $\boldsymbol{x_t}$, respectively. For the examples in this paper, we consider the objective to be forecasting. To further impose structure, we introduce a second objective inspired by contrastive loss and self-organizing maps, the distance-weighted contrastive loss (L_{DWCL}) (Appendix 6). For a single sample, we define the loss function for to be

$$L_{R_{\boldsymbol{s_t}}} = L_{\text{objective}}(\boldsymbol{y_t}, R_{\boldsymbol{s_t}}(\boldsymbol{x_t}; \theta_{R_{\boldsymbol{s_t}}})) + \alpha \times L_{\text{DWCL}}(R_{\boldsymbol{s_t}}(\boldsymbol{x_t}; \theta_{R_{\boldsymbol{s_t}}}), R(\boldsymbol{x_t}; \theta_R)) \tag{1}$$

$$L_{\text{DWCL}}(R_{\boldsymbol{s_t}}, R) = -\log \frac{e^{R_{\boldsymbol{s_t}}}}{E_{\boldsymbol{z}\sim Z}\left[e^{(R_{\boldsymbol{z}}, R_{\boldsymbol{s_t}})} \times ||\boldsymbol{s_t} - \boldsymbol{z}||_2\right]} \tag{2}$$

where $\boldsymbol{y_t} = (x_{t+1}, x_{t+2}, ..., x_{t+w})$, the ground truth values from the time series, α is a learned hyper-parameter to modulate the relative effects of the two terms, $\boldsymbol{z}$ is a state drawn from the set of all states Z, is a metric of similarity between $R_{\boldsymbol{z}}$ and $R_{\boldsymbol{s_t}}$, such as a normalized dot product or cosine-similarity.

L_{DWCL} forces similar states to be closer and dissimilar states to be separated, causing the waveforms to cluster in the state space. During each forward propagation step, the predictions are generated by each forecasting network in the R-block. After the predictions are generated by each network in the R block, the distance-weighted contrastive loss is created by calculating the difference in the predictions between each of the network in the R-block and network selected by the S block, and weighting it by Euclidean distance of the network and the selected sub-network (blue leftwards arrow).

The S block has a separate loss function L_s, contingent on its objective, which is to predict the spatial state occupied by the next time step. It can be conceptualized as the distance between S block prediction of the best state and the network in the R block with the lowest error with respect to $\boldsymbol{y_t}$ (green leftwards arrow):

$$L_s = \|S(\boldsymbol{x_t}) - \arg\min_{\boldsymbol{s_t}} L(R_{\boldsymbol{s_t}}(\boldsymbol{x_t}))\|_2 \tag{3}$$

While an L_1 or L_2 norm may better capture the information about the density of the spatial networks, in practice it may not provide a sufficient gradient for S to learn well (Appendix 6). Early in learning, when the sub-networks in the R-matrix perform poorly, S displays highly unstable dynamics, which in turn hinders R-block learning. Rather than training S to directly maximize the spatial density, we can improve stability by treating the objective as a classification problem and minimize the negative log likelihood (details in Appendix 6).

The full training algorithm is provided in Algorithm 1.

3.4. Ensemble Weight Sharing

Inductive transfer learning leverages an inductive bias to improve performance on a target task and eliminates redundant learning of patterns in data structure (Zhuang et al. (2021)). To generate sub-networks with weights that represent distinct states rather than shared structure between states in the time series, we employ an inductive transfer learning framework. This procedure increases the gap between the sub-network posteriors, which further enhances the contrastive learning aspect of the network. From an information theoretical perspective, the process of learning a shared posterior can be thought of as a lossless compression of the hidden states by encoding them into a shared embedding. In turn, non-unique learning of state-independent behavior only needs to take place once rather than $a \times b$ times.

While mode collapse is a known problem, we find that the sharing of weights across the first few layers leads to robust performance as seen with the paradigm of transfer learning. This is quite unlike the mode collapse seen in the training of generative adversarial networks. Without the sharing of weights across the first few layers, we find that learning requires significantly more samples because the overall structure of the time series must be learned for each unit in the R block, in addition to learning of the relevant state. In contrast, with weight sharing, we find that learning of the overall structure of time series can be done jointly, and the state separation can be learned by each sub-network.

In our implementation of the SpaRTeN framework, this auxiliary network maps the hidden states of R, which is an $h \times a \times b$ embedding to a low-dimensional embedding h', which is subsequently appended to a dense layer of each sub-network. This procedure ensures that the shared weights are differentiable during training. After back-propagation, weights are copied to all sub-networks.

Algorithm 1 Spatial Projection of Time Series with Temporal Ensembles

```
Require: $\{x_0, \dots, x_T\}$, where $T$ is the length of the time series. $\boldsymbol{x_t}$ represents
  $\{x_{t-k}, \dots, x_t\}$ where $k$ is the look-back period, and $y_t$ represents the forecast
  $\{x_{t+1}, \dots, x_{t+w}\}$ where $w$ is the forecast window. Assign a width and height to the state
  space $a, b \in \mathbb{Z}^+$. Randomly initialize weights of the $R$ and $S$ block: $\theta_R, \theta_S \sim N(0,1)$.
  for $m = k$ to $m = T - w$ do
    $\boldsymbol{x_m} \leftarrow \{x_{m-k}, \dots, x_m\}$
    $\boldsymbol{y_m} \leftarrow \{x_{m+1}, \dots, x_{m+w}\}$
    $(\hat{i,j}) = S(\boldsymbol{x_m}; \theta_S)$
    $\boldsymbol{\hat{y}_{m}}_{(i,j)} = R_{(i,j)}(\boldsymbol{x_m}; \theta_R) \forall \{i : [0,a), j : [0,b)\}$
    Update $R_{(\hat{i,j})}$ via gradient descent on $L_{DWCL}(\boldsymbol{\hat{y}_{m}}_{(\hat{i,j})}, \boldsymbol{y_m})$
    Update $S$ via gradient descent on $L(S(\boldsymbol{x_m}), \arg\min_{i,j}(L(\boldsymbol{\hat{y}_{m}}_{(i,j)}, \boldsymbol{y_m}))$
  end for
```

4. Experiments

4.1. Results on Synthetic Experiments

For our applications, we focus on three distinct tasks involving high frequency time series — online forecasting, zero-shot clustering and clinically significant representation learning. First, we benchmark on standard time series data, using SOTA approaches on standard datasets. Second, we apply these results to intracranial pressure waveforms. We benchmark against SOTA online forecasting models with convolutional approaches such as N-Beats (Oreshkin et al. (2019b)) and attention based methods like Temporal Fusion Transformers (Lim et al. (2019)) and Autoformers (Wu et al. (2021)). N-Beats is a time series model that convolves on trends and seasonality. Temporal Fusion Transformers uses a discrete attention mechanism. Finally, autoformers adds an auto-correlation block to a transformer base. SpaRTEn outperforms on three out of the four datasets drawn from the UCI repository with utilizing simple LSTM subnetworks and a latent space dimensionality of 3×3 (Table 1).

We benchmark against the UCI electricity, UCI traffic dataset, the five-min sub-sampled realized volatility from the Oxford stocks dataset, and the kaggle retail dataset (Asuncion and Newman (2007)). We benchmark on long short-term memory networks (LSTMs), N-Beats, TFTs and Autoformer. We report root mean squared error (RMSE) (Oreshkin et al. (2019a); Lim et al. (2021)).

$$RMSE = \frac{\sum_{i=1}^{N} ||y(i) - \hat{y}(i)||^2}{N}$$

Table 1: RMSE of benchmarks on an online forecasting task

	Datasets			
Model	Electricity	Traffic	Stocks	Retail
LSTM	2.93	32.10	0.13	13.76
N-Beats	2.84	3.10	**0.10**	14.16
TFT	2.49	15.10	0.11	13.87
Autoformer	6.61	3.34	1.24	4.98
SpaRTEn	**1.57**	**1.58**	0.67	**1.59**

Second, we demonstrate that the clusters generated by the S-block of SpaRTEn generally represent the data better than adjacent algorithms. We demonstrate that SpaRTeN can generate prototypical waveforms, that can be utilized by K-Nearest Neighbors to perform state-of-the-art for zero-shot clustering methods. We benchmark on traditional clustering techniques such as KNN with random Silhouette score is calculated by

$$s(i) = \frac{b(i) - a(i)}{max\{a(i) - b(i)\}} \tag{4}$$

where $a(i)$ is the intra-cluster distance, and $b(i)$ is the mean nearest-cluster difference. Clusters are assigned by $SRTN$. Visual representations are the average of all waveforms of given length k that belong to any given cluster.

Table 2: Silhouette score of benchmarks on an unsupervised clustering task

	Datasets			
Model	Electricity	Traffic	Stocks	Retail
Random	0.023	0.22	0.012	0.011
Spectral	0.005	0.09	0.005	0.002
GMM	0.024	0.12	0.011	0.010
SpaRTEn	**0.028**	**0.24**	**0.026**	**0.027**

4.2. Ablation Studies

In this section, we run four different ablation studies. The baseline model contains a latent space of dimension 3×3, a negative log-likelihood loss, an S-Block and an R-Block. We perform three distinct experiments.

First, we ablate the S-Block. Ablation of the S-Block significantly decreases the performance of the model. We anticipate this is because the spread of patterns included in the time series analysis are subject to oversquashing (Alon and Yahav (2020)).

Second, we over-parameterize the latent space to a 10×10. We show that this slightly decreases the performance, but not by much in the online forecasting task across three of the four datasets. Because there are no constraints on how many coordinates the network needs to use, this may simply be the result of self-regularization where the network voluntarily learns a representation that under-utilizes an over-parameterized space. Nevertheless, over-specification of the latent space harms the clustering ability of SpaRTEn.

Third, we ablate the distance-weighted contrastive loss. The distance-weighted contrastive loss was implemented to improve clustering. We see that eliminating the distance-weighted contrastive loss can reduce online forecasting performance and clustering performance.

Table 3: Ablation study on benchmarked datasets

	Dataset							
	UCI Electricity		UCI Traffic		Oxford Stocks		Retail	
Ablation	RMSE	Silhouette	RMSE	Silhouette	RMSE	Silhouette	RMSE	Silhouette
S-block	2.93	**X**	32.10	**X**	**0.13**	**X**	13.76	**X**
10×10	2.58	0.019	**1.58**	0.15	0.57	0.005	1.61	0.001
DWCL	2.57	0.024	1.62	0.17	0.60	0.012	1.60	0.013
None	**1.58**	**0.028**	**1.58**	**0.24**	0.67	**0.026**	**1.59**	**0.027**

Table 4: Demographic details of 344 patients

Demographic	Value (95% CI)
% Female	42.96%
Age	53.22 (31.9, 74.5)
% Medicaid	9.14%
% Non-White	31.9%

5. Time is brain: Representation Learning of Intracranial Pressure Waveforms

5.1. Evaluation Approach/Study Design

In this section, we assess SpaRTeN's ability to learn a set of representations of a realistic waveform by first benchmarking it on standard time series datasets, and then demonstrating its practical application on intracranial pressure wave-forms from the MIMIC-III data-set.

A good interpretable representation of states should be a) able to demonstrate distinct properties within each state, and b) cluster wave-forms within a given time series into a given state.

In order to compare the distinct properties of each state and demonstrate the ability to cluster wave-forms within a given state, we train SpaRTeN with a latent state space of 3×3 on intracranial pressure wave-forms across 4000 time steps to generate a set of 9 distinct classes (Figure 2a). We visualize the aggregate of these in Figure 2b. We optimize hyper-parameters with a grid-search conducted over the state-space and learning rates for both the R and S-blocks, as well as α, and the depth and width of the LSTM sub-networks. We train a KNN with $k = 9$ on the 9 ($3 \times 3 = 9$) distinct waveforms aggregated by state, and evaluate its ability to cluster all the waveforms on the dataset using a silhouette score (Rousseeuw (1987)), which is widely used to evaluate the goodness of a clustering technique and report the results. We conduct 10 bootstraps for this experiment and report 95% confidence intervals.

To highlight the application of SpaRTeN to medical data, we use the MIMIC waveform database to benchmark performance on intracranial pressure (Table 1). The MIMIC waveform database is a dataset consisting of 22,317 waveform records for 10,282 ICU patients, which typically include ECG, arterial blood pressure, respiration and polyplethysmography (Moody et al.). We provide benchmarks on the 344 patients with identified coded intracranial pressure waveforms. This work is mostly methods-based and therefore future work will look at more careful cohort selection and identification of underlying pathophysiology of intracranial hypertension such as stroke, obesity, and pregnancy.

5.2. Data Extraction

We extracted 4,000 timepoints for members of the cohort, with a sampling frequency of 125 Hz. Extraction is conducted via a custom data querying method that excludes waveforms with fewer than 4,000 time points.

5.3. Feature Choices

We extract intracranial pressure as a use case because understanding various physiological processes associated intracranial hyper- and hypo-tension remain controversial (Hawryluk et al. (2022)). Further work into learning the role of cerebral perfusion pressure can guide discussion about indications for monitoring, treatment thresholds, and management of intracranial hypertension.

5.4. Results on Intracranial Pressure

We compare the SpaRTeN to other modes of unsupervised clustering, including k-means applied random sampling (Pedregosa et al. (2011)) as a baseline; spectral clustering (Hochreiter et al. (2010)), which imposes a graph-based approach to clustering and is typically used for sequential genomic data; and, a Gaussian mixture model. We show that the silhouette score for the SpaRTeN representations far outperforms other modes of clustering, and is robust to different sample sizes. Furthermore, SpaRTeN approaches the bounds set by a KNN trained on all the data, which is 0.422 (0.418, 0.424).

Table 5: Silhouette Score of representations of Intracranial Pressure Waveforms

	Sample Size			
Model	10	25	100	400
Spectral	0.131 ±0.023	0.165 ±0.008	0.109 ±0.007	0.156 ±0.003
Random	0.366 ±0.018	0.275 ±0.007	0.252 ±0.005	0.276 ±0.002
GMM	0.341 ±0.019	0.345 ±0.005	0.329 ±0.004	0.344 ±0.003
SpaRTeN	**0.415** ±0.020	**0.422** ±0.006	**0.385** ±0.005	**0.405** ±0.002

Second, upon visual inspection of the generated waveform patterns by clinicians with expertise in ICU care, distinct patterns emerge (Figure 2) Future work should ensure consistency via empirical clinical validation.

At point (0,0), the waveform is both stable and relatively constant. This indicates that the intracranial pressure does not require some form of intervention, and is a strong baseline for what non-pathological activity should look like. At point $(0, 1)$ we start seeing evidence of pathological neuro-vascular activity - the mean change in the intracranial pressure is relatively small, unlike the variation, which is relatively large. ICP variability is part of the response to injuries like trauma (Svedung Wettervik et al. (2020)). Following trauma, for example, high intracranial pressure variability is a physiological response, and suggests that some of the compensatory mechanisms are starting to be hyper-reactive. In contrast, at point $(2, 0)$, intracranial pressure waveform has a U-shaped, which indicates that the net change in the intracranial pressure over this time series is zero. However, the visibly slower dip and return to baseline suggests a relatively slow, hypo-reactive compensatory response to changes in intracranial pressure. A hypo-reactive intracranial pressure is associated with worse performance on the Glasgow Coma Scale (Tian et al. (2013)). A physician may try to shift a patient's state from $(0, 2)$ to $(0, 1)$ in order to improve outcomes by increasing the tone of the sympathetic nervous system (Schmidt et al. (2018)).

In the second row, we start to see acute cerebro-vascular dysregulation. At point $(1, 1)$, we see that the there are signs of instability, followed by a complete over-compensation, and an acute drop in the intracranial pressure. A patient in this state may warrant a CT scan to detect an early aneurysmal rupture. At point (1,1) and (1, 2), we see that there is complete dysregulation of the brain's vasculature with dramatic decreases and increases in intracranial pressure, respectively. These two waveforms are adjacent to each other and highlight intracranial pressure waveforms in a pathological state. In the context of waveform $(1, 2)$, we might clinically witness a hemorrhage. A hemorrhage can increase local volume of blood, and decrease intracranial pressure. If a patient is in a hemorrhagic state such as an intracranial hemorrhage, interventions include endotracheal intubation to protect the airway, blood pressure management and hypertonic saline to reduce intracranial pressure (Caceres and Goldstein (2012)). In $(1, 1)$, we see a rapid decrease in intracranial pressure as might be expected following treatment such as placement of an extra-ventricular drain (Kramer (2021)). Notably, the waveform in $(1, 1)$ is closer to the baseline state than that in $(1, 2)$, which makes sense because $(1, 1)$ involves a treatment designed to restore physiologic state.

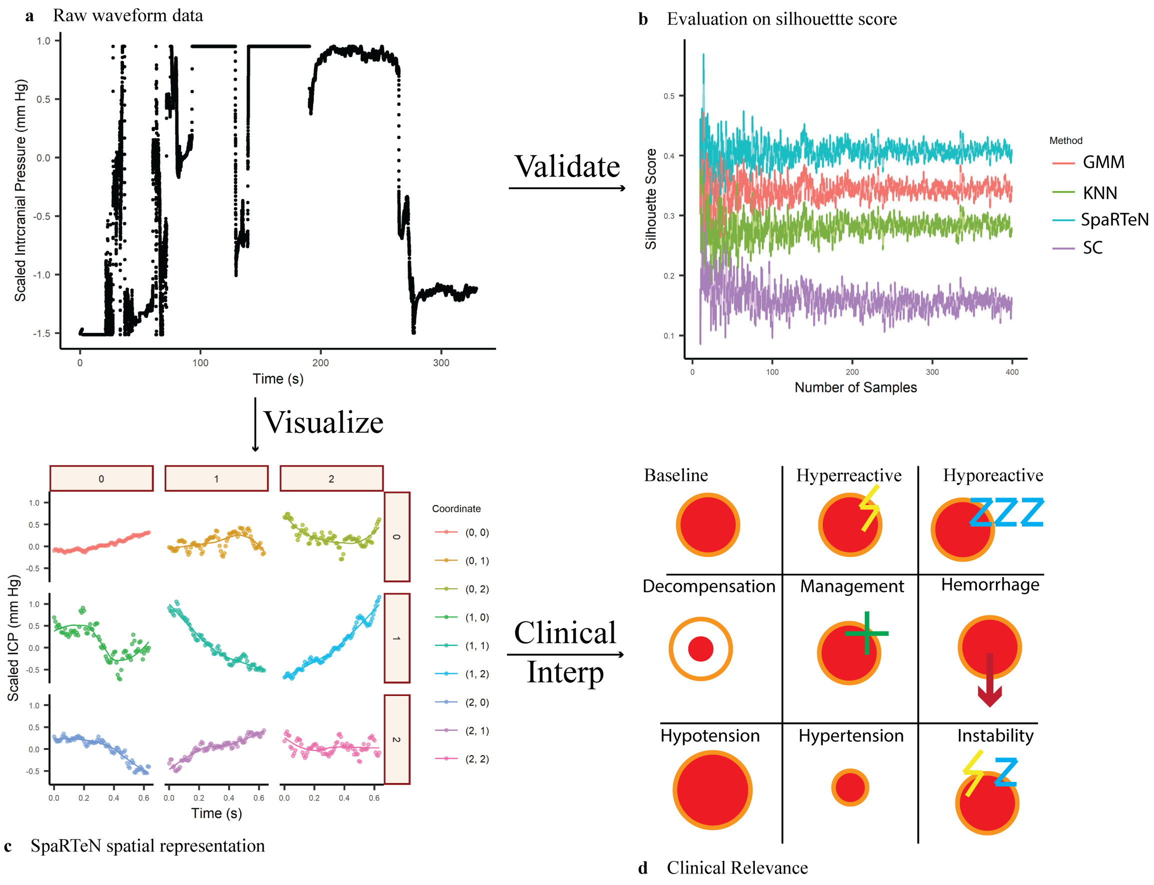


Figure 2: **Qualitatively and quantitatively evaluating model representations.** a) Raw waveforms of Intra-cranial Pressure with high intra-sample variability. b) Bootstrapped (10x) results of silhouette score across different sized samples demonstrates that SpaRTeN outperforms other clustering methods. 95% Confidence intervals reported in the figure, but may be too small to see. c) Clusters generated by SpaRTeN represent distinct trends within the time series. d) Clinical interpretation of each of the waveforms.

The bottom row, namely $(2,0)$ and $(2,1)$ represents decompensation but more chronically than acutely as was observed with $(1,1)$ and $(1,2)$. In $(2,0)$, we could see what chronic hypotension could look like, whereas in (2, 1), we might see what chronic hypertension would look like. Intracranial hypotension is associated with headaches (Luetzen et al. (2021)), and can either be acute or chronic. In $(2,1)$ we notice that there is some form of chronic hypertension, which can be treated clinically with a diuretic drug. In $(2,2)$, we see instability with respect to intracranial pressure, which can be a precursor to $(1,2)$ and $(1,1)$ (Oernbo et al. (2022)). These analyses demonstrate that SpaRTEn is able to decipher clinically meaningful states. Moreover, utilizing these state analyses to better disentangle states can improve the understanding of clinical treatment and associated outcomes (Samartsidis et al. (2018)).

5.5. Ablation Studies

We run three different ablation studies. Ablation of the S-Block, overparameterization of the latent space to a 10 × 10 space and ablation of the distance weighted contrastive loss all result in decreased performance. This is consistent with the ablation studies carried out on other datasets in Section 4.2.

Table 6: Ablation studies on ICU dataset

Ablation Experiment	RMSE	Silhouette Score
S-Block	5.58	**X**
10 ×10	7.59	0.137
DWCL	5.71	0.273
None	**5.41**	**0.401**

6. Discussion

We introduce a novel method called SpaRTeN (Spatially Resolved Temporal Networks) for discrete representations of time series via unsupervised learning and a forecasting objective. SpaRTeN stores models rather than samples in an embedding space, which allows for rapid interpretable learning of structured representations in high frequency time series. We show that it can be broadly applied to online forecasting and clustering. We anticipate that improvements in size, algorithmic and optimization details will only continue to further improve upon the SpaRTeN framework. Finally, we further intend to demonstrate the applicability of this model architecture to real-time, online clinical decision support in situations like the decoding states for patients in the ICU. Thus, the SpaRTeN framework can be generalized to different network blocks, optimization techniques and use cases. It takes a key step towards the goal of generating individualized state representations with online learning.

Analyses performed by trained clinicians demonstrate that SpaRTEn is able to decipher clinically meaningful states. Moreover, utilizing these state analyses to better disentangle states can improve the understanding of clinical treatment and associated outcomes (Samartsidis et al. (2018)).

Limitations SpaRTeN is a novel min-max framework for decoding states, and has many of the same advantages and disadvantages as other min-max frameworks. Without sufficient gradient-based optimizations like smoothing and replacing density-based losses with negative log-likelihood losses, the gradients and states learned by SpaRTeN can be highly unstable (Appendix D). Subsequently, a collapse in the gradients on one of the blocks can be highly detrimental to other blocks.

Second, many datasets, especially in the ICU contain multi-modal sources of information. Currently, models like temporal fusion transforms can better account for multi-modal trends in time series and combine categorical with continuous variables. We anticipate further development of the SpaRTeN framework by including R-blocks that are capable of accounting for different variable types and data modalities may further enhance the ability of SpaRTeN to generate multi-modal archetype waveforms, which can be subsequently used to qualitatively evaluate changing states in the clinical setting.

Third, we selected 2D geometry because it was computationally tractable in terms of the distance-weighted contrastive loss, and interpretable in the ICU setting. Ablation of the the distance-weighted contrastive loss leads to poorer representation learning and clustering. Future work could explore higher-dimensional latent spaces and hyperbolic geometry.

References

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*, 2020.

Samaneh Aminikhanghahi and Diane Cook. A survey of methods for time series change point detection, 2017.

Arthur Asuncion and David Newman. Uci machine learning repository, 2007.

Ane Blazquez-Garcia and Angel Conde. A review on outlier/anomaly detection in time series data, 2022.

J Alfredo Caceres and Joshua N Goldstein. Intracranial hemorrhage. *Emergency medicine clinics of North America*, 30(3):771–794, 2012.

Vincent Fortuin, Matthias Hüser, Francesco Locatello, Heiko Strathmann, and Gunnar Rätsch. SOM-VAE: Interpretable discrete representation learning on time series. June 2018.

Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. January 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z Ghahramani, M Welling, C Cortes, N Lawrence, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

Josif Grabocka, Martin Wistuba, and Lars Schmidt-Thieme. Scalable discovery of Time-Series shapelets. March 2015.

R Hadsell, S Chopra, and Y LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742, June 2006.

Gregory WJ Hawryluk, Giuseppe Citerio, Peter Hutchinson, Angelos Kolias, Geert Meyfroidt, Chiara Robba, Nino Stocchetti, and Randall Chesnut. Intracranial pressure: current perspectives on physiology and monitoring. *Intensive Care Medicine*, 48(10): 1471–1481, 2022.

Anfeng He, Chong Luo, Xinmei Tian, and Wenjun Zeng. A twofold siamese network for real-time object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4834–4843, 2018.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. November 2019.

Sepp Hochreiter, Ulrich Bodenhofer, Martin Heusel, Andreas Mayr, Andreas Mitterecker, Adetayo Kasim, Tatsiana Khamiakova, Suzy Van Sanden, Dan Lin, Willem Talloen, Luc Bijnens, Hinrich W H Göhlmann, Ziv Shkedy, and Djork-Arné Clevert. FABIA: factor analysis for bicluster acquisition. *Bioinformatics*, 26(12):1520–1527, June 2010.

Steven C H Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289, October 2021.

Sundreen Asad Kamal, Changchang Yin, Buyue Qian, and Ping Zhang. An interpretable risk prediction model for healthcare with pattern attention.

Oguzhan Karaahmetoglu, Fatih Ilhan, Ismail Balaban, and Suleyman Serdar Kozat. Unsupervised online anomaly detection on irregularly sampled or missing valued Time-Series data using LSTM networks. May 2020.

Andreas H Kramer. Critical icp in subarachnoid hemorrhage: how high and how long? *Neurocritical Care*, 34(3):714–716, 2021.

Yuening Li, Zhengzhang Chen, Daochen Zha, Mengnan Du, Denghui Zhang, Haifeng Chen, and Xia Hu. Learning disentangled representations for time series. May 2021.

Bryan Lim, Sercan O. Arik, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting, 2019. URL https://arxiv.org/abs/1912.09363.

Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *Int. J. Forecast.*, 37(4):1748–1764, October 2021.

Osian Llwyd, Jui-Lin Fan, and Martin Müller. Effect of drug interventions on cerebral hemodynamics in ischemic stroke patients. *Journal of Cerebral Blood Flow & Metabolism*, 42(3):471–485, 2022.

Niklas Luetzen, Philippe Dovi-Akue, Christian Fung, Juergen Beck, and Horst Urbach. Spontaneous intracranial hypotension: diagnostic and therapeutic workup. *Neuroradiology*, 63(11):1765–1772, 2021.

Chao Luo, Chenhao Tan, Xingyuan Wang, and Yuanjie Zheng. An evolving recurrent interval type-2 intuitionistic fuzzy neural network for online learning and time series prediction. *Appl. Soft Comput.*, 78:150–163, May 2019.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The M4 competition: Results, findings, conclusion and way forward. *Int. J. Forecast.*, 34(4):802–808, October 2018.

David M Maslove, Benjamin Tang, Manu Shankar-Hari, Patrick R Lawler, Derek C Angus, J Kenneth Baillie, Rebecca M Baron, Michael Bauer, Timothy G Buchman, Carolyn S Calfee, et al. Redefining critical illness. *Nature Medicine*, 28(6):1141–1148, 2022.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. UMAP: Uniform manifold approximation and projection, 2018.

Susan P Mollan, Fizzah Ali, Ghaniah Hassan-Smith, Hannah Botfield, Deborah I Friedman, and Alexandra J Sinclair. Evolving evidence in adult idiopathic intracranial hypertension: pathophysiology and management. *Journal of Neurology, Neurosurgery & Psychiatry*, 87 (9):982–992, 2016.

B Moody, M Craig, A Johnson, T Kyaw, G Moody, M Saeed, and M Villarroel. The MIMIC-III waveform database matched subset, physionet. org, 2017. doi: 10.13026.

Eva K Oernbo, Annette B Steffensen, Pooya Razzaghi Khamesi, Trine L Toft-Bertelsen, Dagne Barbuskaite, Frederik Vilhardt, Niklas J Gerkau, Katerina Tritsaris, Anja H Simonsen, Sara D Lolansen, et al. Membrane transporters control cerebrospinal fluid formation independently of conventional osmosis to modulate intracranial pressure. *Fluids and Barriers of the CNS*, 19(1):1–25, 2022.

Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. May 2019a.

Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting, 2019b. URL https://arxiv.org/abs/1905.10437.

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.

Wenjie Pei, David M J Tax, and Laurens van der Maaten. Modeling time series similarity with siamese recurrent networks. March 2016.

Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL https://www.sciencedirect.com/science/article/pii/0377042787901257.

Pantelis Samartsidis, Shaun R Seaman, Anne M Presanis, Matthew Hickman, and Daniela De Angelis. Assessing the causal effect of binary interventions from observational panel data with few treated units. April 2018.

Ramasamy Savitha, Arulmurugan Ambikapathi, and Kanagasabai Rajaraman. Online RBM: Growing restricted boltzmann machine on the fly for unsupervised representation. *Appl. Soft Comput.*, 92:106278, July 2020.

Eric A Schmidt, Fabien Despas, Anne Pavy-Le Traon, Zofia Czosnyka, John D Pickard, Kamal Rahmouni, Atul Pathak, and Jean M Senard. Intracranial pressure is a determinant of sympathetic activity. *Frontiers in physiology*, 9:11, 2018.

Shaolong Sun, Yunjie Wei, and Shouyang Wang. AdaBoost-LSTM ensemble learning for financial time series forecasting, 2018.

Teodor Svedung Wettervik, Timothy Howells, Per Enblad, and Anders Lewén. Intracranial pressure variability: relation to clinical outcome, intracranial pressure–volume index, cerebrovascular reactivity and blood pressure variability. *Journal of clinical monitoring and computing*, 34(4):733–741, 2020.

Ye Tian, Zengguang Wang, Ying Jia, Shengjie Li, Bin Wang, Shizhao Wang, Lin Sun, Jianning Zhang, Jieli Chen, and Rongcai Jiang. Intracranial pressure variability predicts short-term outcome after intracerebral hemorrhage: a retrospective study. *Journal of the neurological sciences*, 330(1-2):38–44, 2013.

Aaron Van Den Oord and Oriol Vinyals. Neural discrete representation learning., 2017.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *J. Mach. Learn. Res.*, 9(11), 2008.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

Zhi-Hua Zhou. Ensemble learning. *Encyclopedia of biometrics*, 1:270–273, 2009.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proc. IEEE*, 109(1): 43–76, January 2021.

Appendix A. Mathematical Notation

Table 7: Mathematical Notation

Symbol	Meaning
$\boldsymbol{x_t}$	A time series from $t-k$ to t. $\{x_{t-k}, x_{t-k+1}, \ldots, x_t\}$. k is the look-back period.
$\boldsymbol{y_t}$	A time series from $t+1$ to $t+w$. $\{x_{t+1}, \ldots x_{t+w}\}$. w is the forecast window.
$\boldsymbol{s_t}$	The state of a time series at time-point t.
	Corresponds to (i, j), a coordinate within the latent space.
	Within S, the range of possible states
$\boldsymbol{\hat{y}_{t_s}}$	Prediction of y_t given state s
T	Total length of time series.
i	The x-coordinate of a state. Constraint: $i < a$
j	The y-coordinate of a state. Constraint: $i < b$
Z	The latent space of states. Constrained to $\{\mathbb{Z}^{2+} : [0, a), [0, b)\}$
a	The width of the latent space.
b	The height of the latent space.
f_S	$f_S : X_t \rightarrow s_t$. The S-block.
f_R	$f_R : (s, X_t) \rightarrow y_t; \forall s \in S$. The R-Block.
R_s	A network in the R-block. Maps $X_t \rightarrow \hat{y}_{ts}$ given s
θ_f	Parameter of function f.
L	Loss function.

Appendix B. Model Architecture and Training

We employ a model architecture that utilizes the SpaRTeN framework. It consists of an S block and an R block with a spatial embedding space of $\mathbb{Z}^{2+} : [0, a), [0, b)$ (Figure 1).

Each step in training occurs in three progressive steps during forward propagation and two steps during back-propagation (Figure 1). During forward propagation, the R block maps an input time series to a set of forecasts $a \times b$ (Step 1 - Blue). The S block takes the input time series and generates a spatial density over the states (Step 1 - Green).

The second step is that the spatial densities corresponding to the predicted state over the S block is used to select the network in the R block to predict the time steps over the look-forward period (Step 2 - Blue, Green).

During backpropagation, there are two distinct loss functions that must be accounted for. First, the S block loss can be calculated by generating all the predictions in networks in the R blocks (Step 3 - Green). The difference between S-block prediction of the best state and the network in the R block with the lowest error with respect to the true future values (in the case of online forecasting), can be calculated (L_S), and the subsequently it can be either treated as a classification task with cross-entropy loss or a mean-squared error loss with the S block trying to approximate the density generated by the R networks.

Second, the R block loss can be calculated by utilizing the predictions generated by all the networks, and the predictions generated by the correct network, and weighting those such that networks with closer spatial distances to the correct network should have closer predictions, whereas, networks that are further away from the correct network have more leeway and should have further estimates (Step 3 - Blue). We can create inductive biases across the ensemble via weight-sharing across the initial layers (Appendix **??**), which improves performance (Appendix **??**)

For the R-Block, we minimize sMAPE (Symmetric Mean Absolute Percentage Error) as the primary objective in a forecasting task:

$$\text{sMAPE} = \frac{1}{N}\sum_{i=1}^{N} 2 \times \frac{|\boldsymbol{y_i} - \boldsymbol{\hat{y}_i}|}{|\boldsymbol{y_i}| + |\boldsymbol{\hat{y}_i}|} \tag{5}$$

where N is the number of examples used for training. sMAPE is a metric that has been typically reported in the past with competitions like the M4 time series forecasting competition Makridakis et al. (2018). For the S-Block, we utilize a standard cross-entropy loss for multi-class classification.

The goal of the S-block is to translate a high-frequency time series into a spatial coordinate system with a dimensionality of a, b. The flexibility of fully connected networks in conjunction with spatial constraints imposed by convolutional filters biases the network towards a spatial representation of the temporal networks.

The S-block consists of four key layers, a 1D-CNN, a fully connected network layer with $(a + 2) \times (b + 2)$ number of units, a layer that reshapes the fully connected network block into an $(a + 2) \times (b + 2)$ rectangle, followed by a 3×3 convolution with a stride length of 2, to produce an ultimate output layer of dimension ab (Figure 3).

A discrete state space was chosen to improve the interpretability of the model subnetworks to produce meaningful results. However, future work may replace the discrete state space output with a representation of a density distribution or a continuous vector space.

In order to compare the distinct properties of each state and demonstrate the ability to cluster waveforms within a given state, we train the S-Block with a latent state space of 3×3. We train a KNN with $k = 9$ on the 9 $(3 \times 3 = 9)$ distinct waveforms aggregated by state, and evaluate its ability to cluster all the waveforms on the dataset using a silhouette score, which is widely used to evaluate the goodness of a clustering technique. SpaRTEn outperforms all other methods used to cluster time series on all four datasets.

We used datasets from the UCI repository: Electricity, Traffic, Oxford Stocks, and Retail. We utilized an 80-20 train-test split. We included learning rates from $1 \times 10^{-}4$ to 1.0 for the grid-search, iterating by a factor of 2. For the intracranial pressure waveforms, we utilize a time series of length 4,000 with an equivalent train-test ratio of 80-20.

Appendix C. Distance-Weighted Contrastive Loss

We adapt contrastive loss and self-organizing maps in the second term to the distance-weighted contrastive loss (DWCL). For a single sample,

$$L_{\text{DWCL}}(R_{\boldsymbol{s_t}}, R) = -\log \frac{e^{R_{\boldsymbol{s_t}}}}{E_{\boldsymbol{z}\sim Z}\left[e^{\text{sim}(R_{\boldsymbol{z}}, R_{\boldsymbol{s_t}})} \times ||\boldsymbol{s_t} - \boldsymbol{z}||_2\right]} \tag{6}$$

where $\boldsymbol{z}$ is a state drawn from the set of all states Z, sim is a metric of similarity between $R_{\boldsymbol{z}}$ and $R_{\boldsymbol{s_t}}$, such as a normalized dot product or cosine-similarity.

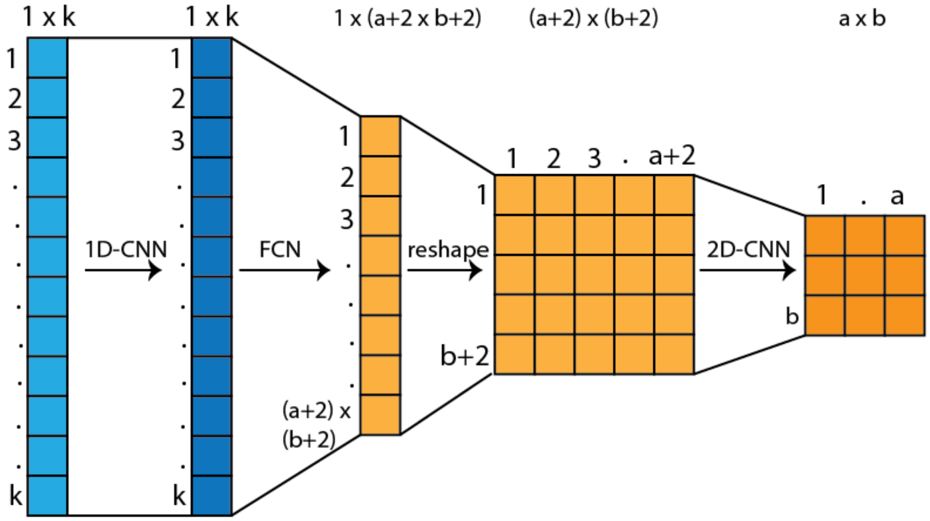


Figure 3: Network architecture of the S-Block. The S-block consists of four key layers, a 1D-CNN, a fully connected network layer with $(a+2) \times (b+2)$ number of units, a layer that reshapes the fully connected network block into an $(a+2) \times (b+2)$ rectangle, followed by a 3×3 convolution with a stride length of 2, to produce an ultimate output layer of dimension width length.

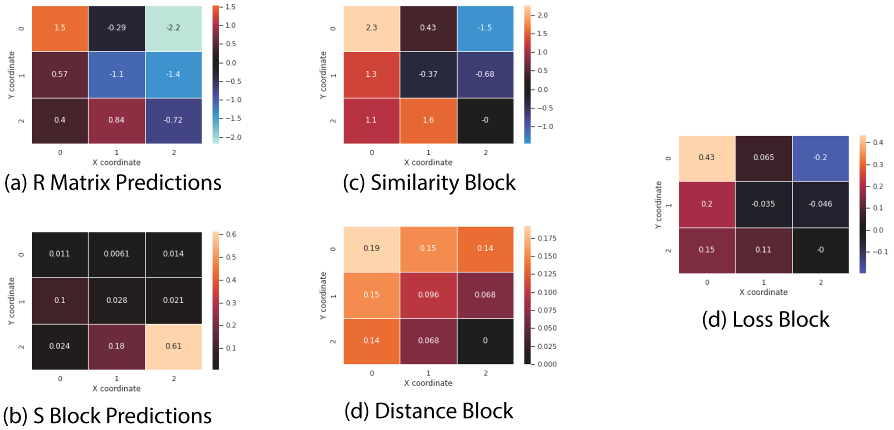
This loss pushes sub-networks to have distinct predictions. The distance-weighted contrastive loss for univariate time series learns similar and dissimilar pairs in a self-supervised manner. During each forward propagation step, the predictions are generated by each forecasting network in the R-block. After the predictions are generated by each network in the R block, the distance-weighted contrastive loss is created by calculating the difference in the predictions between each of the network in the R-block and network selected by the S block, and weighting it by Euclidean distance of the network and the selected sub-network. Similar pairs can be thought of networks that are closer to the selected network in euclidean space, and dissimilar pairs are those that are further apart in Euclidean space.

The overall computational cost is $O(c(f,b) + (K-1) \times c(f))$, where $c(f)$ is the cost of forward propagating, and $c(f,b)$ is the cost of forward and back-propagating, and K is the total number of blocks. Thus, computational cost scales with the number of sub-networks.

The general idea of contrastive loss is to preserve neighborhood relationships between data points by minimizing the distance between similar points and maximizing the distance between points of different classes Hadsell et al. (2006). The general form of the contrastive loss function is the following:

$$L_{\text{contrastive}}(x_i, x_j; \theta) = \frac{1[y_i = y_j]}{2} d(f_\theta(x_i), f_\theta(_j)) + \frac{1[y_i \neq y_j]}{2} \max(0, \epsilon - d(f_\theta(x_i), f_\theta(x_j)) \quad (7)$$

where x_i and x_j are two distinct samples, f is a function that maps $x \rightarrow R^k$, an embedding of dimensionality k, d is a distance metric, and ϵ is the distance to the margin.



(a) R Matrix Predictions

(c) Similarity Block

(d) Loss Block

(b) S Block Predictions

(d) Distance Block

Figure 4: Loss calculations for the R-matrix. (a) Is the calculation of the R-matrix predictions for the next time step. (b) S-Block predicts the appropriate state for the next time step. (c) The similarity block calculates similarity between the chosen state prediction and the other states. (d) The distance block calculates the distance between each other state and the selected substates. The loss block is the dot product of the distance and similarity block. The loss block is summed to produce the final loss value

In multi-class classification problems, this can be further extended as a classification problem with $K+1$ categories He et al. (2019).

$$L_q = -\log \frac{e^{\text{sim}(f_\theta(x_t), f_\theta(x_j))/\tau}}{\sum_{k=1}^{N} e^{\text{sim}(f_\theta(x_k), f_\theta(x_j))/\tau}} \tag{8}$$

where $\text{sim}(f(x_i), f(x_j))$ is a metric of similarity between $f(x_i)$ and $f(x_j)$ and τ is the normalization factor.

We can extend this to forecasting where the positive example can be thought of as the selected sub-network, whereas the negative examples are the irrelevant sub-networks. Finally, we add a normalized distance metric, to ensure sub-networks that are closer in euclidean space have closer representations.

$$L_{\text{DWCL}}(R_{i,j}, R) = -\log \frac{e^{R_{i,j}}}{E_{x,y\sim\mathbb{Z}^{2+}}\left[e^{\text{sim}(R_{x,y}, R_{i,j})} \times \frac{\sqrt{(x-i)^2+(y-j)^2}}{\sum_{x,y}^{\mathbb{Z}^{2+}} (x-i)^2+(y-j)^2}\right]}$$

We visualize this further in Figure 4.

Appendix D. Encouraging smoothness over time

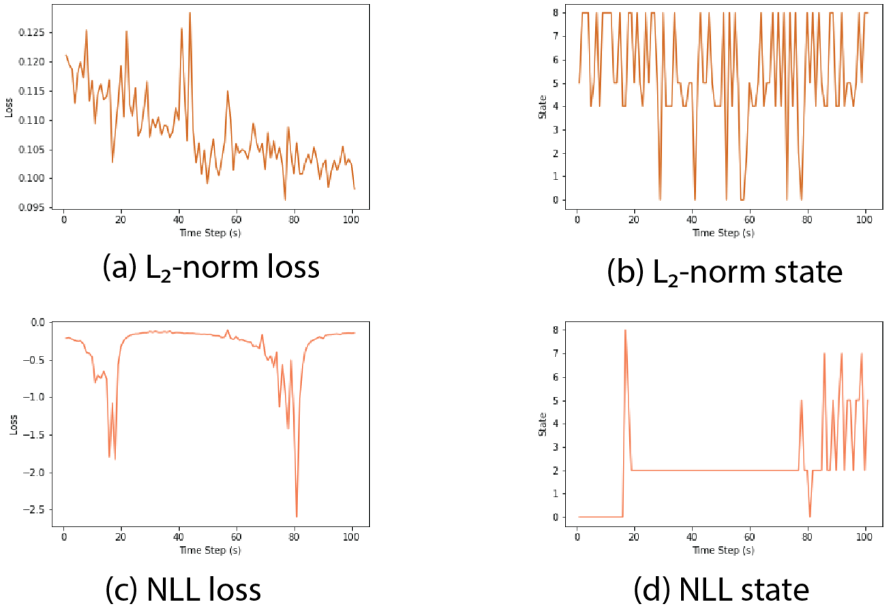
The goal is to predict the development of a time series in an interpretable way. This means that we may have a tradeoff between stable network dynamics and representation of a ground truth density. Learning a probabilistic model in a high-dimensional continuous space can be challenging, which necessitates the use of reductionist frameworks to improve interpretability.

Previous work in Markov chain modeling penalized state transitions via an additional smoothness term Fortuin et al. (2018). Other methods have focused on incorporating quantile outputs to maximize the signal-to-noise ratio Lim et al. (2021).

We find that by converting an L_2-norm-based loss function to cross-entropy loss, we can improve the stability of both the S-block representations, and by extension, the R-block ensemble:

$$L_S = -\sum_{i,j}^{\mathbb{Z}^{2+}:[a,b]} \arg\min_{i,j}(R_{i,j}(\boldsymbol{x_t}) - x_{t+1})^2 \times \log \sigma(S(\boldsymbol{x_t})) \tag{9}$$

where $\mathbb{Z}^{2+}$ is a discrete two-dimensional space of integers in $[a, b]$, the sum is over all the coordinates in the space, $R_{i,j}(\boldsymbol{x_t})$ is the prediction of the next time step by the network based on the previous time step, x_{t+1} is the next step. σ represents soft-max function, and $S(\boldsymbol{x_t})$ is the predicted state of the next time step. If S fails to provide strong initial gradients, as in the case with L_2-norm, then the instability of the network prevents a single sub-network from learning the characteristics of a given state (Figure 5). In turn, this causes the S-block to be increasingly volatile, which can in turn further destabilize the R-block.



(a) L_2-norm loss

(b) L_2-norm state

(c) NLL loss

(d) NLL state

Figure 5: Losses and associated states for L_2-norm and negative log-likelihood. (a) L_2-norm loss is significantly smaller and signal-to-noise ratio is smaller than (c) negative log likelihood (NLL) loss. The corresponding states calculated by the (b) L_2 loss are far more unstable than the states calculated by the (d) negative log likelihood S-block.