SPATIALLY RESOLVED TEMPORAL NETWORKS: ONLINE UNSUPERVISED REPRESENTATION LEARNING OF HIGH FREQUENCY TIME SERIES

Anonymous authors
Paper under double-blind review

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 in an electrocardiogram signal can indicate a heart attack, and intracranial pressure waveforms can indicate whether a patient is developing decreased blood perfusion to the brain. 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. We simultaneously train two distinct blocks: a recurrent neural network ensemble f_R that captures states in high frequency time series, and a spatial block f_S that spatially resolves state changes from the predictions generated by f_R. The spatial block f_S identifies the block in f_R that best fits the current state of the time series, and the training procedure for f_R optimizes that block. This procedure corresponds to a minimax framework. When f_S and f_R are deep neural networks, the entire system can be trained via back-propagation. Finally, we demonstrate the application of this framework to online forecasting and interpretable, zero-shot clustering. We compare and demonstrate that SpaRTeN outperforms spectral clustering and a Gaussian mixture model.

1 INTRODUCTION

Univariate high-frequency time series arise in several domains including economics, medicine, and environmental studies Thorsen-Meyer et al. (2020); Shakeel & Srivastava (2021); Yao et al. (2019). High-frequency time series often exhibit different states: for example, an electrocardiogram (ECG) waveform can represent atrial fibrillation or sinus bradycardia and a stock market trend can represent a bullish or bearish market. 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 & Conde (2022); Aminikhanghahi & Cook (2017); Van Den Oord & Vinyals (2017) but few are dedicated towards disentangling states within a single time series Franceschi et al. (2019) or predicting future state transitions.

In high-frequency time series data like ECGs and arterial 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 changes in heart rate variability, it may be an indicator that the patient is entering a septic state Aboab et al. (2008). Early detection of state transitions may provide clinicians with the tools to intervene earlier for better outcomes. Supervised learning falls short because a) determining which and how many models to apply on an real-time basis is a clinically taxing, b) patients display variable responses to different clinical conditions, and c) real-time data collection in practice can generate noisy waveforms that are uninterpretable for algorithms trained on preprocessed and standardized datasets.

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 & 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.

Second, these methods are often static or offline in that they learn from an entire training dataset at once, before being evaluated and deployed. However, high-frequency time series data like waveforms are often encountered in scenarios requiring 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). This is different from batch or offline learning where data can be shuffled, batched, and is trained on a 'training dataset' before validating on a validation dataset. Most forecasting models are pretrained on a separate training dataset and evaluated on a separate validation dataset. However, in situations such as stock market, ICU waveform measurements and weather calculations, training data is received sequentially. Online learning accounts for the receipt of data sequentially. In the situation of offline learning, every time a new batch of data is received, the entirety of the model needs to be retrained. This can be problematic, especially in the context of dataset shift or high inter-sample variability. ECG and intracranial pressure waveform data in the intensive care unit (ICU) are examples of clinically relevant data that are presented to algorithms in an unsupervised online manner, where learning and predicting state-based representations are tasks important for decision support Sutton et al. (2019).

To extract states or state-transitions from a high-frequency time series, online unsupervised representation is required. Online unsupervised representation learning of high-frequency time series is a relatively understudied field. Fuzzy neural networks create a set of modifiable rules Luo et al. (2019). Fuzzy logic-based rule insertion and deletion can adapt well to time series patterns being currently observed but continuous 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, 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. First, we benchmark against this method, showing that our method outperforms given the same number of parameters in online learning tasks. Second, we also note that this method can be included within our method due to the interoperability of the composite neural network.

We propose a composite differentiable unsupervised deep learning network to learn a discrete spatial representation from a high frequency time series via temporal ensemble learning–a method called **Spa**tial **R**esolved **Te**mporal **N**etworks (SpaRTeN). While we demonstrate one variant of this network, the overall framework can be generalized to many use cases, different network blocks and optimization procedures. We can train both parts of the composite network via back-propagation and generate a spatial representation via forward propagation.

1.1 Contributions

- We introduce SpaRTEn, a new framework for learning spatial representations from high-frequency time series.
- We demonstrate that introduction of a latent space improves rather than harms SpaRTEn's ability to forecast medical high frequency waveform real world data in an online setting, improving upon state-of-the-art methods.
- We show that SpaRTeN generate archetypal waveforms, which perform state of the art for zero-shot clustering tasks.

2 Background

For reference, we include the mathematical notation in Appendix A.1.

2.1 PROBLEM SETUP.

We consider mapping the end point X_t in a time series of length k, $\{x_{t-k}, x_{t-k+1}, \ldots, x_t\}$ to a state s_t:

$$s_t = f(X_t; \theta) \tag{1}$$

where θ represents a networks parameters. We further constrain s_t to represent a discrete point in the positive space of integers $\mathbb{Z}^{2+}$:

$$s_t = \{(i, j); i : [0, a), j : [0, b); i, j \in \mathbb{Z}^{+}\} \tag{2}$$

where i represents the x-coordinate of s_t and j the y-coordinate of s_t. i and j are constrained to be positive integers less than the width a and height b of the latent space. This problem setup imposes a spatial constraint on associative memories learned by Hopfield networks Ramsauer et al. (2020). Self-organizing feature maps (SOMs) also map inputs onto two dimensional Euclidean spaces, but are offline and constrained to inputs of fixed dimension Kohonen (1990).

2.2 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.

2.3 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 adaptationKaraahmetoglu 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.

2.4 DISCRETE LATENT SPACES

Deep Learning Networks with 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 Williams et al. (2021). 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 2b). 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. In contrast, SpaRTEn can take advantage of the diverse potential loss functions for the R-Block and improve individual sub-network performance.

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 Model Framework

3.1 Composite Neural Network

A schematic overview of our composite model is depicted in Figure 1. The architecture we construct consists of a two-block model - an *S* block and an *R* block. The first block, *S*, generates a mapping from the time series to a state s_t. The second block, *R*, includes a spatially constrained ensemble of sub-networks each of which corresponds to a state $R_s \in R$ and makes a forecast y_t with an arbitrary forecast window w, $\{x_{t+1}, \ldots, x_{t+w}\}$.

$$f_S : X_t \rightarrow s_t$$
$$f_{R_{s_t}} : X_t \rightarrow y_t$$

We train *S* to maximize the probability of assigning the time series to the most suitable sub-network in *R*, which corresponds to the minmax framework proposed inGoodfellow et al. (2014) (Equation 3) .

$$\min_R \max_S E_{t \in \tau} L(R(S(X_t)) \tag{3}$$

where L is an objective function, R is the ensemble of state networks, S is a function that maps a time series to a density across the states, and t is drawn from the set of all time series τ.

We choose to place s 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 expanded depending on the *a priori* assumptions of dataset complexity.

Sub-networks can have a primary objective ($L_{\text{objective}}$) of forecasting, classification, or reconstruction if we assign y_t to be the forecasting window, a labeled class or x_t, respectively. However, to further impose structure, we introduce a secondary loss for time series inspired by contrastive loss and self-organizing maps. The secondary objective ensures that a state learned by a sub-network in the R-block is unique. This provides a distinct advantage over previous methods that can embed similar states with different embeddings — unique states with unique embeddings improve interpretation. For a single input, we define the loss function for to be

$$L_{R_{s_t}} = L_{\text{objective}}(y_t, R_{s_t}(X_t; \theta_{R_{s_t}})) + \alpha \times L_{\text{DWCL}}(R_{s_t}(X_t; \theta_{R_{s_t}}), R(x_t; \theta_R)) \tag{4}$$

where α is a learned hyper-parameter to modulate the relative effects of the two losses. For the examples in this paper, we consider the primary objective to be forecasting.

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

$$L_{\text{DWCL}}(R_{s_t}, R) = -\log \frac{e^{R_{s_t}}}{E_{z \sim Z}\left[e^{\text{sim}(R_z, R_{s_t})} \times ||s_t - z||_2\right]} \tag{5}$$

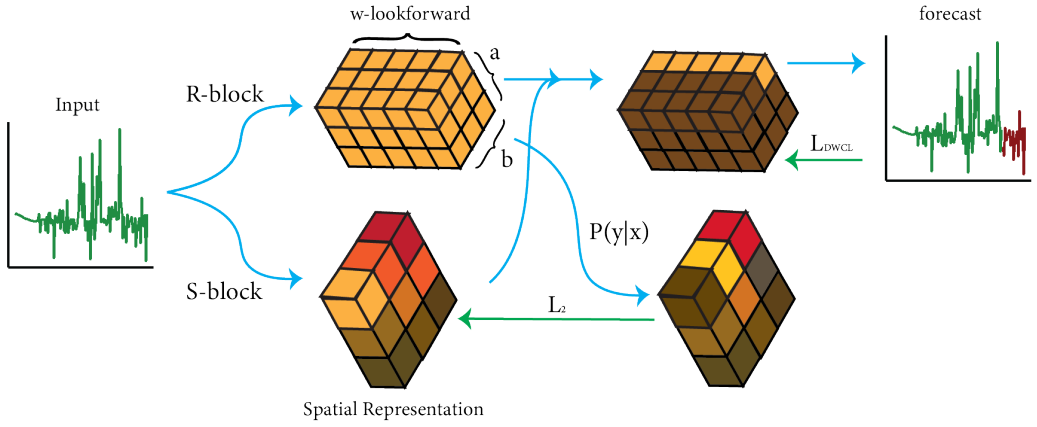

Figure 1: **Schematic representation of model architecture.** Blue is forward propagation and green is back-propagation. Time series from the data space are forecast by the sub-networks in an R-block with a look-forward period of k. The dimensionality of the spatial representation is a $a \times b$. 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 5.

where z is a state drawn from the set of all states Z, sim is a metric of similarity between R_z and R_{s_t}, such as a normalized dot product or cosine-similarity. This loss pushes sub-networks to have distinct predictions via the contrastive loss aspect of it, and is further discussed in Appendix A.2. Furthermore, the denominator is penalized by the distance function, which permits similar states to be neighbors and distant states to be farther. 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. Computational cost scales with the number of sub-networks.

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. S outputs a density function across the state-space domain, and the target for the S-block is the best sub-network in the R-block.

$$L_s = \|S(X_t) - L(R(X_t))\|_2 \tag{6}$$

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 A.3). 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 (Appendix A.3).

3.2 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). R is an $a \times b$ matrix of sub-networks, where each sub-network is an LSTM with a forecast window of w. Notably, the sub-networks within R can be exchanged for other networks including transformers or random forests. During forward propagation, each LSTM in R computes a forecast, generating a complete output matrix of $w \times a \times b$, where a and b are the dimensions of the spatial constraint. We implement S as a 1D-CNN that generates a probability distribution over the discrete coordinates (i,j) i.e. $i : [0,a), j : [0,b), i,j \in \mathbb{Z}^+$ (Appendix A.4). For the next time step, $f_S : X_t \rightarrow P_{(i,j)}$ over the discrete coordinate space. The sub-network with the greatest density maps to the outcome of interest: $f_R : R_{\underset{(i,j)}{\arg\max} P_{(i,j)}}(X_t) \rightarrow y_t$.

During back-propagation, the sub-network with the greatest density $R_{\underset{(i,j)}{\arg\max} P_{(i,j)}}$ is back-propagated with respect to the ground-truth y_t via a gradient-descent based rule using the distance-weighted contrastive loss L_{DWCL}. To update S, first the L_2-norm of each sub-network and the ground truth are calculated to generate a target matrix with dimension $a \times b$. The best performing network $\underset{(i,j)}{\arg\min} L(R_{(i,j)}, y_t)$ is used as the class label for the output vector of S, and subsequently cross-entropy loss is calculated and used to back-propagate S.

3.3 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.

In our implementation of the SpaRTeN framework, this auxiliary network is modeled via a simple LSTM, which is shared between sub-networks. The LSTM 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 the fully connected network 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. The full training algorithm is provided in Algorithm 1. Gradient based updates can use any standard

Algorithm 1 Spatial Projection of Time Series with Temporal Ensembles

Require: $\{x_0, \ldots, x_T\}$, where T is the length of the time series. X_t represents $\{x_{t-k}, \ldots, x_t\}$ where k is the look-back period, and y_t is the forecast period $\{x_{t+1}, \ldots, x_{t+w}\}$ where w is the forecast window. Assign a width and height to the latent 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**

  $X_m \leftarrow \{x_{m-k}, \ldots, x_m\}$

  $y_m \leftarrow \{x_{m+1}, \ldots, x_{m+w}\}$

  $(\hat{i, j}) = S(X_m; \theta_S)$

  $\hat{y_{m(i,j)}} = R_{(i,j)}(X_m; \theta_R) \forall \{i : [0, a), j : [0, b)\}$

  Update $R_{(\hat{i,j})}$ via gradient descent on $L_{DWCL}(\hat{y_{m(\hat{i,j})}}, y_m)$

  Update S via gradient descent on $L(S(X_m), \underset{i,j}{\arg\min}(L(\hat{y_{m(i,j)}}, y_m))$

end for

gradient-based learning rule. We used Adam optimization for our experiments.

4 Application

For our applications, we focus on two distinct tasks involving high frequency time series — online forecasting and zero-shot clustering. Online forecasting involves looking at a window in a time series and using that window to predict the next set of elements. First, we benchmark online forecasting tasks on standard time series data, and then apply it to ECG data. Online forecasting for ECG data is an especially important in an ICU setting, where predicting short-term changes in ECG can indicate that a patient is going into sepsis, but noisy data acquisition in conjunction with high intra-class and inter-patient variability has mitigated the deployment of offline methods in the ICU.

Second, zero-shot clustering is an unsupervised method that involves classifying the time series the first time it is seen without information corresponding to a label. 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.

Subsequently, we demonstrate this technique qualitatively on intracranial pressure (ICP) waveforms because identifying patterns in ICP waveforms in real-time can be utilized in a clinical setting to mitigate cerebral edema or ischemia. We perform all experiments in PyTorch on an A100 NVIDIA GPU, and bootstrapped 10 times. The repository can be accessed here (link removed for double blind peer review; code provided as supplementary material).

4.1 Online Forecasting with SpaRTeN

In the architecture we chose, we used sub-networks consisting of a latent space with 9 single-layer LSTMs organized in a 3×3 fashion. These dimensions were chosen to be computationally tractable and yet complex enough to be interesting. The forecasting block of SpaRTeN has approximately 500 parameters, and so we compared it to a TFT with a similar number of parameters. However, networks such as N-beats and TFT can replace the current LSTM sub-network due to the modular nature of the R matrix. Ultimately, the goal of the experiment is to show that discretizing the latent space does not significantly harm the forecasting ability of the network.

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), where SpaRTEn outperforms on three out of the four datasets (Appendix A.5). Moreover, we run ablation experiments across these four datasets including ablation of the S-Block, increasing the size of the latent space, and eliminating the distance-weighted contrastive loss (Appendix A.5). These ablation studies demonstrate that the S-block is necessary for adequate performance in online forecasting, which may be due to oversquashing in naive LSTM Alon & Yahav (2020) implementations, wherwas the other aspects of model are less important.

To highlight the application of SpaRTeN to medical data, we use the MIMIC waveform database (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 poly-plethysmography Moody et al.. This dataset is released under the Open Data Commons Open Database License v1.0, is publicly available and does not contain personally identifiable information. Waveforms in healthcare applications from ICU sources typically require online forecasting to provide real-time feedback on future patient states.

As a result, we train and evaluate on a simple online-forecasting task to impute waveform. The dataset consists of 60,000-time steps of lead II ECG data for an ICU patient from the MIMIC waveform database. We use a window length of 80 timesteps and a forecasting length of 20 time steps, typical of online forecasting tasks. The R-Block and S-Block weights were all initialized via the technique described in He et al. (2015). We used a grid search during the first 100 time steps to determine an ideal learning rate with an Adam optimizer for both the R and S-Blocks, the depth and width of each LSTM, and the hyper-parameter α to relate the primary and secondary losses for the R-block. We select a cross-entropy loss function for the S-Block to encourage smoothness over time, detailed in Appendix A.3. For the R-Block, we minimize sMAPE (Symmetric Mean Absolute Percentage Error) as the primary objective in a forecasting task:

$$\text{sMAPE} = \frac{1}{N} \sum_{i=1}^{N} 2 \times \frac{|y_i - \hat{y}_i|}{|y_i| + |\hat{y}_i|} \tag{7}$$

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). We report 10x bootstrapped experiments and their corresponding 95% confidence intervals (Table 1).

4.2 Representation Learning of High-Frequency Time Series

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 (Appendix A.5), and then demonstrating its practical application on intracranial pressure waveforms from the MIMIC-IV data-set. Intracranial pressure waveforms are a good example for this assessment because identifying underlying states may provide insight into an appropriate treatment regimen to avoid an acute neurological injury. A

Table 1: sMAPE of SpaRTEn on an online forecasting task involving Lead II ECGs

	sMAPE		
Model	mean	lower 95% CI	upper 95% CI)
LSTM	1.84	1.79	1.89
N-Beats	1.53	1.51	1.55
TFT	1.63	1.62	1.64
SpaRTeN	**1.48**	**1.47**	**1.49**

good interpretable representation of states should be a) able to demonstrate distinct properties within each state, and b) cluster waveforms 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 waveforms within a given state, we train SpaRTeN 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. Rousseeuw (1987). 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 compare the SpaRTeN to other modes of unsupervised clustering, including k-means applied random sampling Pedregosa et al. (2011) as a baseline; spectral clusteringHochreiter et al. (2010), which imposed 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 (Appendix A.5).

We run additional ablation experiments across these four datasets including increasing the size of the latent space, and eliminating the distance-weighted contrastive loss, which lead to deterioration on the clustering metric (Appendix A.5). For example, eliminating the distance-weighted contrastive loss leads to a 15% decrease in silhouette score for UCI electricity, a 29% decrease for UCI traffic, and a 53% decrease for kaggle retail. Although the online forecasting aspect of SpaRTen performed poorly in the Oxford stocks dataset, the clustering task beat the benchmarks and eliminating the distance-weighted contrastive loss led 51% decrease in silhouette score. We hypothesize that adding a spatial loss improves RNN diversity by increasing state separation-which both aids interpretability and downstream clustering tasks. Second, we find that an over-parameterized the state-space can actually lead to self-regularization, where the network learns to under-utilize the coordinate system given the complexity of the time series.

Subsequently, we highlight the use of SpaRTen on real-time representation learning of intracranial pressure waveforms. We train SparTEn on ICP waveforms across 400 time steps to generate 9 distinct classes (Figure 2). First, we show that SpaRTEn is robust to different sample sizes (Table 2). Second, we visualize the aggregate of these in Figure 2b.

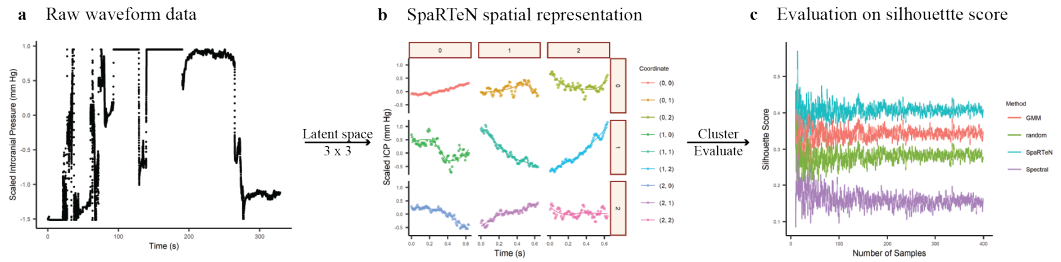

Figure 2: **Qualitatively and quantitatively evaluating model representations.** a) Raw waveforms of Intra-cranial Pressure with high intra-sample variability. b) Clusters generated by SpaRTeN represent distinct trends within the time series. c) Bootstrapped (10x) results of silhouette score across different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

Table 2: 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	$\mathbf{0.415} \pm 0.020$	$\mathbf{0.422} \pm 0.006$	$\mathbf{0.385} \pm 0.005$	$\mathbf{0.405} \pm 0.002$

Upon visual inspection of the waveform patterns recognized by SpaRTeN, we can see distinct trends emerge. For example $(2, 0)$ is parabola shaped, whereas $(1, 0)$ recognizes dips in intra-cranial pressure. These trends are interpretable in the ICU as a change from cerebral ischemia to edema, and a cerebral vessel rupture, respectively. The trend in $(2, 1)$ shows an acute cerebral edematous inflow, whereas $(2, 2)$ shows a sinusoidal hyper-acute vasoconstriction and dilation in the area measured.

To our knowledge, this is the first algorithm to use an online contrastive learning approach for time series classification. It is known that discriminative region-based zero-shot learning in images can preserve context information Narayan et al. (2021). SpaRTeN representations quantitatively capture variations within the data, and qualitatively provide key clinical insights into waveform patterns.

5 Potential 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 A.3). Subsequently, a collapse in the gradients on one of the blocks can be highly detrimental to the other block.

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 for adequate interventions if necessary.

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

6 Conclusions

We introduced 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. This work is not expected to lead to negative societal implications.

7 Reproducibility Statement

All experiments were seeded with PyTorch. The code for the algorithm is attached in the supplements.

8 ETHICS STATEMENT

All experiments utilize publicly available data de-identified waveform data from the MIMIC-III waveform database, and were conducted with IRB approval.

REFERENCES

Jerome Aboab, Andrea Polito, David Orlikowski, Tarek Sharshar, Muriel Castel, and Djillali Annane. Hydrocortisone effects on cardiovascular variability in septic shock: a spectral analysis approach. *Crit. Care Med.*, 36(5):1481–1486, May 2008.

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.

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 (eds.), *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, pp. 1735–1742, June 2006.

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*, pp. 4834–4843, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

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.

T Kohonen. The self-organizing map. *Proc. IEEE*, 78(9):1464–1480, September 1990.

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.

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.

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

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.

Sanath Narayan, Akshita Gupta, Salman Khan, Fahad Shahbaz Khan, Ling Shao, and Mubarak Shah. Discriminative region-based Multi-Label Zero-Shot learning. August 2021.

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.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, David Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need. July 2020.

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.

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.

Moonis Shakeel and Bhavana Srivastava. Stylized facts of high-frequency financial time series data. *Global Bus. Rev.*, 22(2):550–564, April 2021.

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

Jacob R Sutton, Ruhi Mahajan, Oguz Akbilgic, and Rishikesan Kamaleswaran. PhysOnline: An open source machine learning pipeline for Real-Time analysis of streaming physiological waveform. *IEEE J Biomed Health Inform*, 23(1):59–65, January 2019.

Hans-Christian Thorsen-Meyer, Annelaura B Nielsen, Anna P Nielsen, Benjamin Skov Kaas-Hansen, Palle Toft, Jens Schierbeck, Thomas Strøm, Piotr J Chmura, Marc Heimann, Lars Dybdahl, Lasse Spangsege, Patrick Hulsen, Kirstine Belling, Søren Brunak, and Anders Perner. Dynamic and explainable machine learning prediction of mortality in patients in the intensive care unit: a retrospective study of high-frequency data in electronic patient records, 2020.

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.

Jennifer Williams, Yi Zhao, Erica Cooper, and Junichi Yamagishi. Learning disentangled phone and speaker representations in a semi-supervised vq-vae paradigm. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7053–7057, 2021. doi: 10.1109/ICASSP39728.2021.9413543.

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.

Fangfang Yao, Jida Wang, Chao Wang, and Jean-François Crétaux. Constructing long-term high-frequency time series of global lake and reservoir areas using landsat imagery. *Remote Sens. Environ.*, 232:111210, October 2019.

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.

A Appendix

A.1 Mathematical Notation

A.2 Distance-Weighted Contrastive Loss

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:

$$L_{\text{contrastive}}(x_i, x_j; \theta) = \frac{1[y_i = y_j]}{2} d(f_\theta(x_i), f_\theta(x_j)) + \frac{1[y_i \neq y_j]}{2} \max(0, \epsilon - d(f_\theta(x_i), f_\theta(x_j)) \tag{8}$$

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

$$L_q = -\log \frac{e^{\text{sim}(f_\theta(x_t), f_\theta(x_j))/\tau}}{\sum_{k=1}^{N} e^{\text{sim}(f_\theta(x_k), f_\theta(x_j))/\tau}} \tag{9}$$

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.

Table 3: Mathematical Notation

Symbol	Meaning
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.
y_t	A time series from $t+1$ to $t+w$. $\{x_{t+1}, \ldots x_{t+w}\}$. w is the forecast window.
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
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 $X_t \rightarrow \hat{y}_{t_s}$ given s
$\hat{y}_{t_s}$	Prediction of y_t given state s
θ_f	Parameter of function f.
L	Loss function.
sim	A metric of similarity. Normalized dot produce or cosine similarity, for example.

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 3.

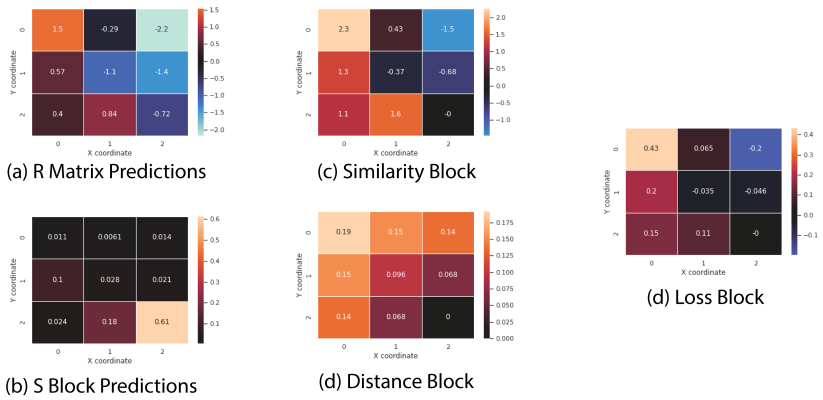

Figure 3: Loss calculations for the R-matrix. (a) Is the calculation of the R-matrix predictions for the next time step. (b) S-Block predicts the appropriate state for the next time step. (c) The similarity block calculates similarity between the chosen state prediction and the other states. (d) The 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

A.3 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.:

$$L_S = -\sum_{i,j}^{\mathbb{Z}^{2+}:[a,b]} \arg\min_{i,j}(R_{i,j}(x_t) - x_{t+1})^2 \times \log \sigma(S(x_t)) \tag{10}$$

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, $R_{i,j}(x_t)$ 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 $S(x_t)$ 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 4). In turn, this causes the S-block to be increasingly volatile, which can in turn further destabilize the R-block.

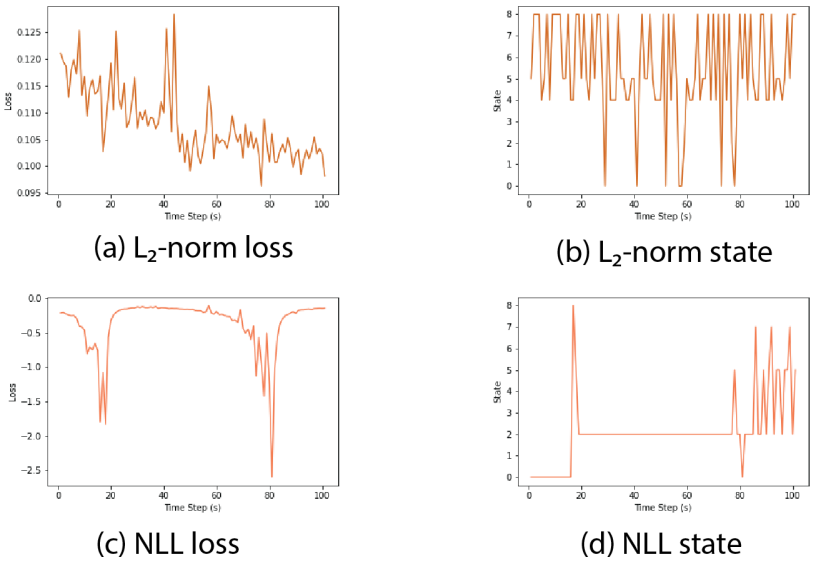

Figure 4: Losses and associated states for L_2-norm and negative log-likelihood. (a) L_2-norm loss is significantly smaller and signal-to-noise ratio is smaller than (c) negative log likelihood (NLL) loss. The corresponding states calculated by the (b) L_2 loss are far more unstable than the states calculated by the (d) negative log likelihood S-block.

A.4 S-BLOCK ARCHITECTURE

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 5).

A discrete state space was chosen to improve the interpretability of the model sub-networks 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.

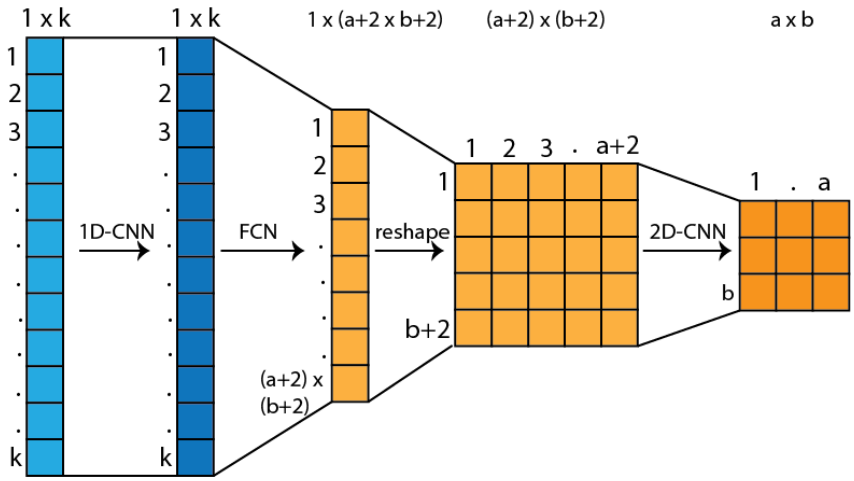

Figure 5: Network architecture of the S-Block. The S-block consists of four key layers, a 1D-CNN, a fully connected network layer with $(a + 2) \times (b + 2)$ number of units, a layer that reshapes the fully connected network block into an $(a + 2) \times (b + 2)$ rectangle, followed by a 3×3 convolution with a stride length of 2, to produce an ultimate output layer of dimension width length.

A.5 BENCHMARKS

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 & Newman (2007). We benchmark on long short-term memory networks (LSTMs), N-Beats, and temporal fusion transformers on an online forecasting task. We report root mean squared error (RMSE) Oreshkin et al. (2019a); Lim et al. (2021).

Table 4: 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 benchmark the representations learned via SpaRTEn via a clustering metric, silhouette score. We benchmark on the same datasets above, and compare against standard models such as k-means applied to random sampling as a baseline; spectral clustering, which imposed 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 exceeds other modes of clustering.

Table 5: 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**

A.6 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 & 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 6: 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**