Sequential Multi-Dimensional Self-Supervised Learning for Clinical Time Series

Aniruddh Raghu [1] **Payal Chandak** [2] **Ridwan Alam** [1] **John Guttag** [1] **Collin M. Stultz** [1]

Abstract

Self-supervised learning (SSL) for clinical time series data has received significant attention in recent literature, since these data are highly rich and provide important information about a patient's physiological state. However, most existing SSL methods for clinical time series are limited in that they are designed for unimodal time series, such as a sequence of structured features (e.g., lab values and vitals signs) or an individual high-dimensional physiological signal (e.g., an electrocardiogram). These existing methods cannot be readily extended to model time series that exhibit multimodality, with structured features *and* high-dimensional data being recorded at each timestep in the sequence. In this work, we address this gap and propose a new SSL method — *Sequential Multi-Dimensional SSL* — where a SSL loss is applied both at the level of the entire sequence and at the level of the individual high-dimensional data points in the sequence in order to better capture information at both scales. Our strategy is agnostic to the specific form of loss function used at each level – it can be contrastive, as in SimCLR, or non-contrastive, as in VICReg. We evaluate our method on two real-world clinical datasets, where the time series contains sequences of (1) high-frequency electrocardiograms and (2) structured data from lab values and vitals signs. Our experimental results indicate that pre-training with our method and then fine-tuning on downstream tasks improves performance over baselines on both datasets, and in several settings, can lead to improvements across different self-supervised loss functions.

[1]Massachusetts Institute of Technology, Cambridge, MA, USA [2]Harvard-MIT Program in Health Sciences and Technology, Cambridge, MA, USA. Correspondence to: Aniruddh Raghu <araghu@mit.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

1. Introduction

In clinical settings such as the intensive care unit (ICU), patients are closely monitored and consequently generate a profusion of time series data. This rich data contains significant physiological information about a patient's state and progression over time (Johnson et al., 2016). As a result, there have been many efforts to study representation learning and pre-training on these data, particularly using self-supervised learning (SSL) strategies, with the goal of using the pre-trained models for various downstream predictive tasks (McDermott et al., 2021; Weatherhead et al., 2022; Tonekaboni et al., 2021; Yèche et al., 2021; Tipirneni & Reddy, 2022).

Although these works develop effective strategies to model clinical time series data, they focus only on unimodal time series, such as a sequence of structured features alone, or an individual high dimensional physiological signal. In reality however, data originating from a patient's encounter is significantly more complex, containing multimodal data recorded at regular intervals. As an example, a given patient may have two very different types of data recorded hourly: (1) high-frequency physiological signals (e.g., an electrocardiogram recorded at 240 Hz); and (2) structured data from labs and vitals signs. These modalities provide complementary information about a patient's physiological state. Extending existing SSL methods to operate on these time series is challenging, since they do not deal with sequences of high-dimensional data, and do not contend with the multimodal data stream.

In this paper, we take steps towards addressing this gap and outline an approach for self-supervised pre-training on these complex clinical time series. We propose a SSL strategy where we jointly optimize two SSL losses to better capture structure in the data. Our contributions are as follows:

1. We formalize the problem of self-supervised learning (SSL) on *trajectories*, our abstraction of a multimodal time series that contains complex, high-dimensional data recorded at each timestep in the sequence.
2. We outline a new SSL method, *Sequential Multi-Dimensional Self-Supervised Learning* (SMD SSL), for trajectories. Motivated by the structure of trajectories, SMD SSL incorporates two losses: (1) a component SSL loss on the level of individual high dimensional

data points in the sequence; and (2) a global SSL loss on the level of the overall sequence. SMD SSL can be instantiated with contrastive losses, as in SimCLR (Chen et al., 2020a) or non-contrastive losses, as in VICReg (Bardes et al., 2022a). This is beneficial since different loss functions may be effective in different applications.

3. We evaluate SMD SSL on two real-world clinical datasets where the time series contains sequences of (1) high-frequency electrocardiograms and (2) structured data from labs and vitals signs. On both datasets and on two downstream tasks — (1) detecting elevated pulmonary pressures and (2) predicting 24 hour mortality — we find SMD SSL improves performance over baselines. In several settings, we observe performance boosts using both SimCLR and VICReg objective functions.

2. Related work

Self-supervised learning (SSL). SSL methods are used to pre-train models and/or learn generalizable representations using unlabeled data. Many existing methods take either a multiview perspective (Chen et al., 2020a;c; Bardes et al., 2022a; He et al., 2020; Chen et al., 2020c; Grill et al., 2020) or an autoregressive denoising approach (Vincent et al., 2008; He et al., 2022). Here, we focus on multiview approaches since they have been effective in improving predictive performance on clinical tasks (Weatherhead et al., 2022; Tonekaboni et al., 2021).

SSL for clinical data. Existing applications of multiview SSL to medical data have been focused either on physiological signals, such as electrocardiograms (ECGs) (Cheng et al., 2020; Kiyasseh et al., 2021; Gopal et al., 2021; Diamant et al., 2021; Oh et al., 2022), on sequences of tabular data, such as laboratory tests (Yèche et al., 2021; Li et al., 2021), or on medical imaging (Ren et al., 2022). In contrast, we consider SSL on time series where individual timesteps contain both high-dimensional data (such as ECGs) and structured features. Prior studies exploring SSL on multimodal medical data typically infer one modality of data from the other at test time, such as predicting radiologist comments from chest X-ray images (Tiu et al., 2022), rather than modeling sequences of multimodal data where the modalities present non-overlapping sources of information, as we do here.

Multilevel SSL loss functions. Our method uses a two-level loss function that is motivated by the complex structure of the data stream we consider: sequences in which individual elements are themselves high-dimensional. A related approach in computer vision, VICRegL (Bardes et al., 2022b), applies multilevel self-supervision to images where patch-level similarity is defined using spatial transformations. In contrast to VICRegL, which formulates a component level loss using patches, our method formulates a component level loss on entire signals, and so operates on a different level of abstraction. Another recent method decouples local and global representation learning for a single time series (Tonekaboni et al., 2022). This work also operates on a different level of abstraction, since it does not consider sequences of time series. Finally, Ren et al. (2022) demonstrate that multiscale SSL for neuroimaging offers improvement on downstream tasks. However, their techniques are tailored for neuroimaging and do not generalize readily to the data we consider.

3. Methods

In this section, we describe our approach for self-supervised learning (SSL) on multimodal clinical time series: *Sequential Multi-Dimensional SSL.* We first outline our problem setup, describing the multimodal data stream that we consider. We then detail our SSL scheme, specifying the loss functions used to learn representations on both an individual timestep (component) level and on a overall sequence (global) level. We conclude with a discussion of other applications of the method.

3.1. Problem Setup

Defining *trajectories*. We use the term *trajectory* to refer to a sequence of physiological signals and structured data collected over time for a patient. This definition is motivated by an important use-case in cardiovascular medicine, where patients may be monitored with telemetry devices that regularly record physiological waveforms in addition to having lab tests and vitals signs periodically measured. The concept of a trajectory could be readily expanded to include other information, such as imaging or medications, depending on the context.

Formally, a trajectory τ of length T has the structure:

$$\tau = (d, \{(w_t, s_t)\}_{t=1}^{T}).$$

Here, $d \in \mathbb{R}^L$ represents a set of static features that do not change over the trajectory (such as demographic information or infrequently measured lab values). The sequence $\{(w_t, s_t)\}_{t=1}^{T}$ contains a vector of structured data $w \in \mathbb{R}^M$, and a high-dimensional signal $s \in \mathbb{R}^{C \times P}$, where C is the number of signal channels and P is the number of samples in the signal (typically on the order of a few thousand). A visualization of a trajectory is shown in Figure 1.

Trajectory neural network. Trajectories are mapped into vector representations using a neural network f_θ with three components (Figure 5): (1) a static and structured features encoder $f_\theta^{w,d}$; (2) a signals encoder f_θ^s; (3) a sequence module f_θ^τ. At each timestep t, the modalities are embedded and concatenated into timestep representations: $z_t = \text{concat}\left(f_\theta^{w,d}(w_t, d), f_\theta^s(s_t)\right)$. The se-

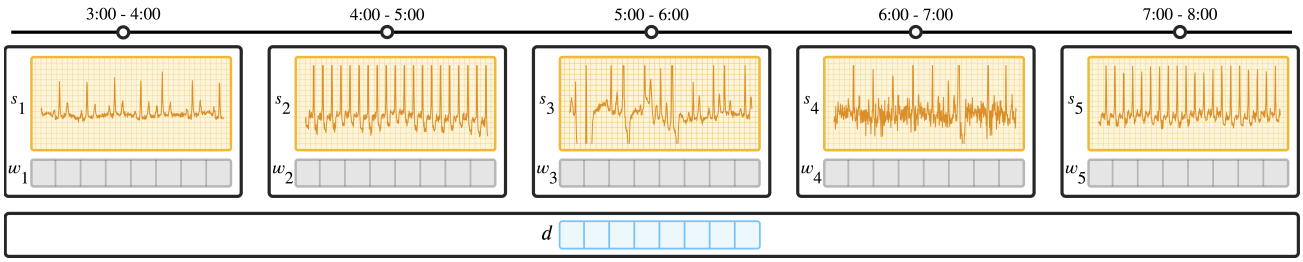

Figure 1: **An example of a multimodal clinical time-series or 'trajectory'.** The trajectory τ contains an static vector d consisting of measurements that remain constant over the time period, and a time series of high-dimensional physiological signals s_t and structured data w_t measured at each time step. Here, each time step is a 1 hour window.

quence module maps these representations into a vector: $z = f_\theta^\tau(z_1, z_2, \ldots, z_T)$. In a supervised setting, a classifier c_ψ maps z to a predicted label $\hat{y}$.

Trajectory self-supervised learning (SSL). Inspired by recent work in SSL (Chen et al., 2020a;b; Bardes et al., 2022a; Zhang et al., 2022), we consider a two-stage learning problem: pre-training (PT) followed by fine-tuning (FT). We first pre-train the model f_θ on an unlabelled dataset of trajectories using some SSL algorithm, and then evaluate the SSL method by FT this pre-trained model on a set of downstream tasks and measuring performance on these tasks. At FT time, different paradigms could be used – we could initialize a classification head and then fine-tune the whole model, or train a linear classifier on the frozen model.

To pre-train the model, we assume access to an unlabelled PT dataset of N_{PT} patient trajectories, $\mathcal{D}_{\text{PT}} = \{\tau^{(n)}\}_{n=1}^{N_{\text{PT}}}$. To fine-tune the model, we assume access to a set of labelled FT datasets – given K FT tasks indexed by k, we denote each FT dataset as $\mathcal{D}_{\text{FT}}^{(k)} = \{(\tau^{(n)}, y^{(n)})\}_{n=1}^{N_{\text{FT}}^{(k)}}$, where $y^{(n)}$ denotes the label for a trajectory $\tau^{(n)}$.

3.2. Sequential Multi-Dimensional SSL

We propose a new method for SSL on trajectories – Sequential Multi-Dimensional SSL (SMD SSL), depicted in Figure 2. Our approach builds on multi-view SSL like SimCLR (Chen et al., 2020a) and VICReg (Bardes et al., 2022a), since prior work has successfully used these strategies on clinical data (Diamant et al., 2021; Kiyasseh et al., 2021; Gopal et al., 2021; Vu et al., 2021; Oh et al., 2022).

SMD SSL uses a loss function with two terms – a global loss, computed at the trajectory level, and a component loss, computed at the individual signal level. We now describe these two losses, and then present the overall objective.

3.2.1. Global Loss

The global loss $\mathcal{L}_{\text{G}}$ encourages the encoding model f_θ to embed similar trajectories to similar points in the representation space. We follow related work (Chen et al., 2020a) and define a similar (or positive) pair of trajectories to be those that are augmentations of the same base trajectory.

Given a trajectory-level augmentation function, the computation of the global loss proceeds as follows:

1. Sample a batch of trajectories from the PT dataset: $\{\tau^{(n)}\}_{n=1}^{B}$, where B is the batch size.
2. For each trajectory $\tau^{(n)}$, generate two augmented views of it: $\tilde{\tau}^{(n)}$ and $\hat{\tau}^{(n)}$.
3. Pass the augmented views through the representation model f_θ and a projection head g_ϕ generating two sets of projections: $\tilde{h}^{(n)}$ and $\hat{h}^{(n)}$.
4. Assemble projected pairs into two sets of matrices: $\hat{H} = [\hat{h}^{(1)}, \ldots, \hat{h}^{(B)}]$, $\tilde{H} = [\tilde{h}^{(1)}, \ldots, \tilde{h}^{(B)}]$.
5. The global loss is equal to the trajectory self-supervised loss $\mathcal{L}_{\text{SSL}}^\tau$ computed on these projections:

$$\mathcal{L}_{\text{G}} = \mathcal{L}_{\text{SSL}}^\tau(\hat{H}, \tilde{H}). \tag{1}$$

The choice of the trajectory self-supervised loss $\mathcal{L}_{\text{SSL}}^\tau$ is a design decision; one choice is the normalized temperature-scaled cross-entropy loss (NT-Xent) as in SimCLR (Chen et al., 2020a;b). Given all $2B$ positive pairs (h_i, h_j) as the rows of the two matrices $[\hat{H}, \tilde{H}]$ and $[\tilde{H}, \hat{H}]$, we compute:

$$\mathcal{L}_{\text{NT-Xent}}^\tau = \frac{1}{2B}\sum_{i=1}^{2B} -\log \frac{\exp(\text{sim}(h_i, h_j)/\gamma)}{\sum_{k=1}^{2B} \mathbb{1}_{[k\neq i]} \exp(\text{sim}(h_i, h_k)/\gamma)}, \tag{2}$$

where $\text{sim}(a, b)$ is cosine similarity and γ is the temperature hyperparameter. Another choice is the VICReg loss, minimizing mean squared error between positive pairs, with variance and covariance regularizers (Bardes et al., 2022a):

$$\mathcal{L}_{\text{VICReg}}^\tau = \lambda I(\hat{H}, \tilde{H}) + \mu\text{Var}(\hat{H}, \tilde{H}) + \nu\text{Cov}(\hat{H}, \tilde{H}), \tag{3}$$

where $I()$ is the mean squared error, Var() is the variance regularizer, Cov() is the covariance regularizer, and λ, μ, and ν are hyperparameters. Flexibility in the form of the loss function is beneficial since different applications might benefit from different losses. In our experiments, we focus on the NT-Xent and VICReg loss functions.

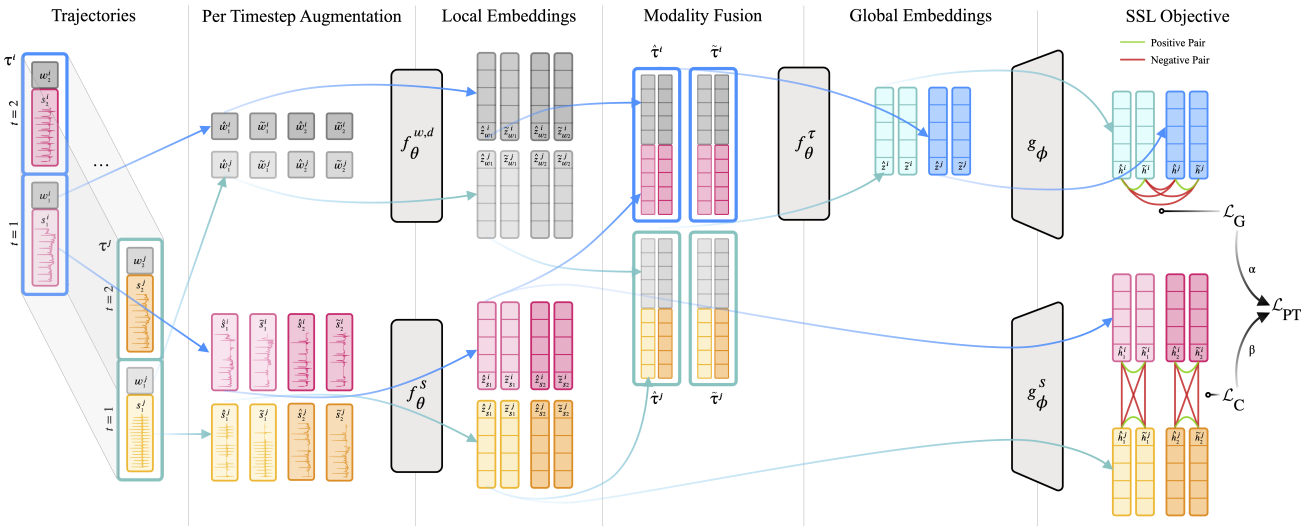

Figure 2: **Overview of Sequential Multi-Dimensional Self-Supervised Learning (SMD SSL),** which uses losses at two levels to encourage effective pre-training on complex time series. We start with a batch of trajectories, each denoted τ, consisting of a static vector d (not shown for clarity) and a sequence of signals s_t and structured data w_t (sequence of length 2 here). These data are augmented on a per-modality and per-timestep basis (arrows show flow for the data at a single timestep) and passed through encoders f_θ^s and $f_\theta^{w,d}$ to generate local embeddings of the signals and structured data at each timestep. The signal embeddings pass through a projection head g_ϕ^s, after which we compute a component SSL loss $\mathcal{L}_{\text{C}}$. Separately, the embedding of the entire trajectory (obtained by concatenating the per-modality embeddings) is passed through a sequence model f_θ^τ and a global projection head g_ϕ, on which we compute the global SSL loss $\mathcal{L}_{\text{G}}$. The total loss $\mathcal{L}_{\text{PT}}$ is a weighted sum of the component and global losses. SMD SSL can be instantiated with both contrastive and non-contrastive losses – shown here is a contrastive framing (as in SimCLR) with explicit negative pairs.

3.2.2. Component Loss

Pre-training with the global loss is a straightforward application of SSL to trajectories. However, each trajectory contains complex substructures (the high-frequency signals s_t) and the global loss alone may not be sufficient to guide the model to learn useful representations of these substructures. We hypothesize that incorporating a second loss term on the individual signal level, the *component loss* $\mathcal{L}_{\text{C}}$, would lead to learning richer representations of the signals.

Given a signal augmentation function, we compute the component loss as follows:

1. Sample a batch of trajectories from the PT dataset: $\{\tau^{(n)}\}_{n=1}^B$, where B is the batch size.
2. Generate two augmented views of each signal in each trajectory. For a given trajectory $\tau^{(n)}$, let the two augmented sets of signals be: $\{\tilde{s}_t^{(n)}\}_{t=1}^T$ and $\{\hat{s}_t^{(n)}\}_{t=1}^T$.
3. Pass the augmented views through the signal encoder model f_θ^s and a signal projection head g_ϕ^s generating two sets of projections: $\{\tilde{h}_t^{(n)}\}_{t=1}^T$ and $\{\hat{h}_t^{(n)}\}_{t=1}^T$.
4. Assemble these pairs of projections into pairs on a per-timestep basis: $\hat{S}_t = [\hat{h}_t^{(1)}, \ldots, \hat{h}_t^{(B)}]$, $\tilde{S}_t = [\tilde{h}_t^{(1)}, \ldots, \tilde{h}_t^{(B)}], \forall t = 1, \ldots, T$.
5. The component loss is equal to the signal self-supervised loss $\mathcal{L}_{\text{SSL}}^s$ averaged over timesteps:

$$\mathcal{L}_{\text{C}} = \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{\text{SSL}}^s(\hat{S}_t, \tilde{S}_t). \tag{4}$$

As with the global loss, the form of the signal SSL loss used is a design decision. The intuition for computing the signal SSL loss separately at each timestep is that nearby timesteps in a trajectory can be very similar. Particularly if we use a contrastive loss such as NT-Xent, we do not want these nearby timesteps to serve as negative examples in the contrastive loss. Separating out the computation over timesteps addresses this issue.

3.2.3. Overall Objective

The overall objective used at PT is:

$$\mathcal{L}_{\text{PT}} = \alpha\mathcal{L}_{\text{G}} + \beta\mathcal{L}_{\text{C}}. \tag{5}$$

The hyperparameters α and β control the contributions of the global and component losses. Fixing $\alpha = 0$ is SSL on a signal-level alone (only PT the signal encoder) and fixing $\beta = 0$ is SSL on the overall trajectory level alone; we evaluate both in our experiments, finding that combining the two losses is beneficial to performance. In Appendix B.3, we study the evolution of the two losses over SMD SSL training, which provides intuition as to the effect of each term during pre-training.

3.3. Augmentation Functions

SMD SSL requires augmentations for trajectories and signals in order to compute the global and component losses respectively. We now describe these.

Trajectory augmentation. We form an augmented trajectory by separately augmenting each of the data modalities within the trajectory, using the following approach for each data type (further details in Appendix A):

- **High-frequency signal s:** For each signal in the trajectory of length T, we form a pair of augmented views by first splitting the signal into two disjoint segments (e.g., taking the first 10 seconds as one view, and the second 10 seconds as the second view) and then applying random masking and noise addition as augmentations to each view independently, similar to the approach used in CLOCS (Kiyasseh et al., 2021). The intuition is that two segments of a signal that are close in time should encode similar physiology, and can therefore be considered paired views. Random masking and noise addition are commonly used as time-series augmentations (Gopal et al., 2021; Zhang et al., 2022; Raghu et al., 2022; Iwana & Uchida, 2021).
- **Structured-time series data w:** The tabular data sequence forms a $T \times M$ matrix over all timesteps of the trajectory. Following prior work (Yèche et al., 2021), we apply two data augmentation strategies to this matrix: Gaussian noise addition and history cutout.
- **Static features d:** Following Yèche et al. (2021), we use random dropout and noise addition. Other corruption strategies (e.g., Bahri et al. (2021)), were found to be less effective, potentially due being too strong (also seen in Levin et al. (2022)).

We note that our approach of forming augmented trajectories by independently transforming each individual data type is straightforward, but not necessarily optimal. Exploring other strategies for generating multiple views of trajectory data is an important direction of future work.

Signal augmentation. For computational efficiency, we re-use the augmentated signals already generated during the trajectory-level augmentation when computing the component loss. However, additional/different augmentations could be applied on the signal level.

3.4. Broader Applications of SMD SSL

We have instantiated SMD SSL for a setting in which multimodal trajectories consists of a sequence of structured data and high-dimensional signals. More generally, our approach is valuable in any setting where we have sequences of high-dimensional data – the two-level loss function encourages representation learning on both an individual signal level and an overall sequence level. For example, SMD SSL could be useful in modeling sequence of medical images for a patient taken over time. The component loss encourages learning rich embeddings of individual images, and the global loss encourages learning temporal trends.

4. Experiments

In this section, we evaluate Sequential Multi-Dimensional Self-Supervised Learning (SMD SSL) on two clinical datasets [1]. We begin by describing the datasets, tasks, and experimental setup. We then evaluate SMD SSL and baselines in two settings: *unimodal*, with trajectories that contain only a sequence of physiological signals; and *multimodal*, with trajectories containing both signals and structured data. We find that SMD SSL performs strongly in both settings. We also analyze SMD SSL's sensitivity to the component loss weight and its learned representations.

4.1. Datasets and Tasks

We consider two clinical datasets (Table 1):

- **Dataset 1** is a private dataset from Massachusetts General Hospital (MGH), consisting of 9605 patients with a prior diagnosis of heart failure.
- **Dataset 2** is a public dataset derived from the commonly used MIMIC-III clinical database (Johnson et al., 2016; Goldberger et al., 2000) and its associated database of physiological signals (Moody et al., 2020). We use the preprocessing pipeline introduced in Harutyunyan et al. (2019) to form our cohort of 5689 patients and extract the structured data features.

Constructing PT and FT sets. Each dataset consists of a number of patient hospital visits. We resample each patient's hospital visit at hourly resolution, and each hour of a patient's stay represents a single timestep in our trajectory abstraction. For simplicity, we fix the length of trajectories to be 8 elements (letting a trajectory correspond to a common shift length of 8 hours).

To generate PT trajectories from these resampled visits, we first split each visit into non-overlapping contiguous 12 hour blocks. A PT trajectory is formed by first sampling a 12 hour block, and then selecting 8 contiguous timesteps from that block (with the starting timestep selected randomly). We discuss the implications of this (particularly relating to negative samples in contrastive losses) in Appendix B.1.

To generate FT trajectories, we use a sliding window to select contiguous 8 hour blocks at 1 hour increments from each visit. Each of these contiguous 8 hour blocks is a trajectory in the FT dataset. The trajectory labels are formed based on the specific task, as described below.

[1] Code at `https://github.com/aniruddhraghu/smd-ssl`.

Table 1: Dataset Statistics.

	Dataset 1			Dataset 2		
Task	# Patients	# Trajectories	Prevalence	# Patients	# Trajectories	Prevalence
Pre-training	8888	43858	N/A	5022	26615	N/A
Elevated mPAP	2025	48511	77.5%	500	14957	87.9%
24hr Mortality	9605	57758	1.4%	5689	318306	2.3%

Fine-tuning tasks. We consider two predictive tasks:

- **Elevated mPAP**: Each hour, detect whether a patient's mean Pulmonary Arterial Pressure (mPAP) is abnormally high. This task is of clinical interest since the mPAP is typically measured via an invasive study, and so inferring whether it is abnormal using minimally invasive signals (i.e., the ECG, labs, and vitals signs) is valuable. Prior work (Schlesinger et al., 2022; Raghu et al., 2023) studied similar tasks of predicting hemodynamic variables from the 12-lead ECG, but not in the context of online trajectory data, as we do here.
- **24hr Mortality:** Each hour, predict whether the patient is going to die in the next 24 hours. This task is commonly used to evaluate predictive models for ICU time series data (McDermott et al., 2021; Yèche et al., 2021) – our goal with studying this task is to understand how our approach performs when compared to other established methods. In Dataset 2, this task is named 'Decompensation' in the preprocessing pipeline from Harutyunyan et al. (2019); we refer to it at 24hr mortality here.

Trajectory features and preprocessing. The trajectories in PT and FT sets consist of static features d and a time-series of structured data and physiological signals $\{(w_t, s_t)\}_{t=1}^{T}$, as presented in Section 3.1. The static features d contain information on infrequently measured vitals signs and lab values; $d \in \mathbb{R}^9$ in Dataset 1 and $d \in \mathbb{R}^{38}$ in Dataset 2. At each timestep, w_t captures summary statistics related to regularly measured vitals signs within a 1 hour window; $w_t \in \mathbb{R}^{30}$ in Dataset 1 and $w_t \in \mathbb{R}^{13}$ in Dataset 2. s_t is a 10 second electrocardiogram (ECG) signal extracted from a longer signal measured within each 1 hour window; in Dataset 1, $s_t \in \mathbb{R}^{4\times 2400}$ is a 240 Hz 4-channel ECG, and in Dataset 2, $s_t \in \mathbb{R}^{1\times 1250}$ is a 125 Hz 1-channel ECG.

Missing structured data are forward-fill imputed if part of a time series and otherwise imputed with the mean over the training dataset. Missing signals are represented with zeros. Any trajectories that have more than 1 timestep with a missing signal are excluded. Further dataset and preprocessing details are in the appendix.

4.2. Experimental Setup

Dataset splits. We split Dataset 1 on a per-patient level into 80/20 development/test sets and use 20% of the development set as a validation set. For Dataset 2, we use the predefined development/test split defined in the preprocessing pipeline (Harutyunyan et al., 2019), and use 20% of the development set (splitting on a per-patient basis) as a validation set.

Model architecture. Recall that the encoder f_θ has three components: we implement the structured features encoder as a 2-layer MLP, the signals encoder as a 1-D ResNet18 CNN (He et al., 2016), and the sequence model as a 4-layer GRU. We use a 2-layer MLP for the projection head g_ϕ. The model architecture is described more fully in Appendix B.2.

Training setup. We conduct pre-training for 15 epochs, using a batch size of 128, with the Adam optimizer (Kingma & Ba, 2014). We found that model performance did not improve with longer pre-training times (Appendix B.3).

At fine-tuning time, we consider two evaluation strategies:

1. Linear evaluation: train a linear classifier on the frozen representations from f_θ (Chen et al., 2020a).
2. Full FT: initialize a new linear layer after f_θ and fine-tune the entire model for a maximum of 10 epochs with Adam (performance did not improve after this point), with early stopping based on validation AUROC.

For each method and task, we report the test set AUROC from the evaluation strategy that obtains the best validation set AUROC. We adopt this approach since our goal is determine to which self-supervised pre-training approach is best – in order to do so fairly, we compare results under the evaluation strategy that obtains the highest performance. To obtain error bars, we use bootstrapping: we sample with replacement 100 bootstraps from the testing dataset, and report the 95% confidence interval in AUROC over these bootstraps. Since the mortality task has low prevalence, we additionally check trends in AUPRC in Appendix B.3.

Unimodal and Multimodal evaluation. SMD SSL is generally applicable when we have sequence-structured data where elements of the sequence are themselves high-dimensional. We consider two instantiations of such sequences here: (1) the *unimodal setting*, where the input trajectory only contains the signals sequence of the input,

Table 2: **Pre-training with the SMD SSL objective improves performance on both datasets in the unimodal setting.** Mean and 95% confidence interval of AUROC on FT tasks. We observe that PT methods outperform not doing PT, and training with the SMD SSL (component and global) losses boosts performance the most.

(a) Results on Dataset 1.

	Elevated mPAP	24hr Mortality
RandInit	65.0 ± 0.1	56.1 ± 0.6
SimCLR (global)	68.1 ± 0.1	66.7 ± 0.5
VICReg (global)	66.6 ± 0.1	64.9 ± 0.5
SimSiam (global)	63.8 ± 0.1	50.6 ± 0.6
SimCLR (component)	67.5 ± 0.1	71.7 ± 0.5
VICReg (component)	68.7 ± 0.1	63.5 ± 0.4
SimSiam (component)	64.2 ± 0.1	54.3 ± 0.5
SMD SSL (SimCLR)	$\mathbf{69.9 \pm 0.1}$	72.3 ± 0.4
SMD SSL (VICReg)	67.6 ± 0.1	$\mathbf{74.6 \pm 0.5}$

(b) Results on Dataset 2.

	Elevated mPAP	24hr Mortality
RandInit	63.4 ± 0.4	54.6 ± 0.2
SimCLR (global)	65.9 ± 0.4	56.6 ± 0.2
VICReg (global)	66.7 ± 0.4	53.6 ± 0.2
SimSiam (global)	61.9 ± 0.4	61.1 ± 0.2
SimCLR (component)	65.7 ± 0.4	61.1 ± 0.2
VICReg (component)	66.7 ± 0.4	63.1 ± 0.2
SimSiam (component)	65.9 ± 0.4	50.3 ± 0.2
SMD SSL (SimCLR)	$\mathbf{67.0 \pm 0.4}$	$\mathbf{65.9 \pm 0.2}$
SMD SSL (VICReg)	66.6 ± 0.4	58.5 ± 0.2

$\tau = \{s_t\}_{t=1}^T$; (2) the full *multimodal setting*, where the input trajectory contains the full input sequence of structured data and signals, $\tau = (d, \{(w_t, s_t)\}_{t=1}^T)$.

Baselines and SMD SSL variations. Existing methods for SSL on clinical data are not exactly applicable, since they do not study pre-training pipelines for multimodal and multi-dimensional time series; e.g., Neighbourhood Contrastive Learning (NCL) (Yèche et al., 2021) is primarily designed for structured data time series alone, and CLOCS (Kiyasseh et al., 2021) and SACL (Cheng et al., 2020) operate on single physiological waveforms (rather than sequences of waveforms).

As a result, we focus in the main paper on evaluating general SSL methods as baselines (with further baselines in Appendix B.3), varying whether we use the component and/or global loss, in both unimodal and multimodal settings. Our goal is to understand whether the two-level loss formulation boosts performance. The full set of baselines is as follows (further details in Appendix B.2):

- **RandInit**: A standard baseline: train a model from random initialization on each FT task.
- **SimCLR (global) (Chen et al., 2020a):** Pre-train using the NT-Xent global loss alone. This is SimCLR PT on the trajectory level, setting $\alpha = 1, \beta = 0$ in Eqn. 5.
- **VICReg (global) (Bardes et al., 2022a):** Pre-train using the VICReg global loss alone. This is VICReg PT on the trajectory level, setting $\alpha = 1, \beta = 0$ in Eqn. 5.
- **SimSiam (global) (Chen & He, 2021):** Pre-train using SimSiam at a global level. This is SimSiam PT on the trajectory level.
- **SimCLR (component):** Pre-train using the NT-Xent component loss alone ($\alpha = 0, \beta = 1$ in Eqn. 5).
- **VICReg (component):** Pre-train using the VICReg component loss alone($\alpha = 0, \beta = 1$ in Eqn. 5).
- **SimSiam (component):** Pre-train using SimSiam at component level.

We consider two variations of SMD SSL:

- **SMD SSL (SimCLR):** Pre-train using SMD SSL with the NT-Xent loss (Eqns. 2 and 5), fixing the global loss weight $\alpha = 1$ and tuning the component loss weight β.
- **SMD SSL (VICReg):** Pre-train using SMD SSL with the VICReg loss (Eqns. 3 and 5), fixing the global loss weight $\alpha = 1$ and tuning the component loss weight β.

Hyperparameters. There are various hyperparameters to tune, such as learning rates and loss weighting for VICReg and SMD SSL. Evaluating many hyperparameters is computationally expensive (involves doing both PT and FT runs), so we conduct a reduced search on a subset of the hyperparameters. We include full details in Appendix B.2.

4.3. Results

4.3.1. Unimodal Evaluation

Table 2 shows results in the unimodal setting, where the input trajectories consist only of the signals. We highlight three key takeaways from these results:

1. **Pre-training (PT) helps performance.** In the unimodal setting, PT (particularly SimCLR or VICReg variants in Tables 2a and 2b) almost always improve performance over not doing any PT (RandInit in Tables 2a and 2b). This result is expected, since we would expect that given the complex input space, a PT phase for the highly-parameterized CNN encoder and GRU should condition the model better for the downstream tasks, given the limited amount of labelled data on these tasks.
2. **SMD SSL obtains the best performance.** On both datasets, we observe that a SMD SSL method does best, suggesting the utility of a two-level loss, which encourages the learning of informative representations on both

Table 3: **Pre-training with the SMD SSL objective improves performance in three settings in a multimodal evaluation.** Mean and 95% confidence interval of AUROC on FT tasks. In all settings except 24hr Mortality on Dataset 1, we observe that SMD SSL obtains the best performance. The 24 hour mortality task on Dataset 1 appears to benefit little from incorporating the high-dimensional signals, which could explain why SMD SSL does not improve performance here.

(a) Results on Dataset 1.

	Elevated mPAP	24hr Mortality
RandInit (signals)	65.0 ± 0.1	56.1 ± 0.6
SSL (signals)	69.9 ± 0.1	74.6 ± 0.5
RandInit (structured)	65.3 ± 0.1	$\mathbf{79.1 \pm 0.4}$
SSL (structured)	66.7 ± 0.1	79.0 ± 0.4
RandInit	69.1 ± 0.1	79.0 ± 0.4
SimCLR (global)	69.8 ± 0.1	76.4 ± 0.5
VICReg (global)	69.4 ± 0.1	78.0 ± 0.4
SimSiam (global)	69.6 ± 0.1	78.4 ± 0.4
SimCLR (component)	71.4 ± 0.1	78.6 ± 0.4
VICReg (component)	64.0 ± 0.1	74.0 ± 0.5
SimSiam (component)	68.4 ± 0.1	79.0 ± 0.4
SMD SSL (SimCLR)	$\mathbf{72.3 \pm 0.1}$	77.4 ± 0.4
SMD SSL (VICReg)	70.3 ± 0.1	77.0 ± 0.4

(b) Results on Dataset 2.

	Elevated mPAP	24hr Mortality
RandInit (signals)	63.4 ± 0.4	54.6 ± 0.2
SSL (signals)	67.0 ± 0.4	65.9 ± 0.2
RandInit (structured)	65.3 ± 0.3	90.0 ± 0.1
SSL (structured)	68.3 ± 0.3	89.3 ± 0.1
RandInit	65.3 ± 0.3	87.8 ± 0.1
SimCLR (global)	63.7 ± 0.4	86.6 ± 0.1
VICReg (global)	70.4 ± 0.4	87.8 ± 0.1
SimSiam (global)	60.6 ± 0.3	90.4 ± 0.1
SimCLR (component)	59.7 ± 0.4	89.8 ± 0.1
VICReg (component)	67.1 ± 0.4	84.4 ± 0.1
SimSiam (component)	67.4 ± 0.4	90.6 ± 0.1
SMD SSL (SimCLR)	69.9 ± 0.3	88.1 ± 0.1
SMD SSL (VICReg)	$\mathbf{71.6 \pm 0.3}$	$\mathbf{90.7 \pm 0.1}$

a signal-level and a sequence-level.

3. **SMD SSL vs single-level SSL.** When using SimCLR, we find that SMD SSL consistently improves on a component-only or global-only SimCLR model. With VICReg, improvements are less consistent, and the component-only VICReg variation often performs the best. This may be because VICReg has many loss weighting terms, and these were not jointly tuned with the component loss weight in SMD SSL. A more thorough hyperparameter search, perhaps using efficient gradient-based methods (Raghu et al., 2021), might improve performance of SMD SSL (VICReg).

4.3.2. Multimodal Evaluation

Table 3 shows results in the multimodal setting, where the input trajectories consist of both signals and structured data. In addition to the aforementioned baselines, we also include results from RandInit and SSL methods trained on only signals and on only structured data. We highlight some key takeaways from these results:

1. **The effect of multimodal data is task-specific.** Incorporating both the signals and structured data leads to improvements in the Elevated mPAP task on both datasets (particularly with SMD SSL), but has less significant effects in the 24hr Mortality task. This is likely because the structured data in their raw (relatively low-dimensional) form are highly predictive of mortality, and so there is little benefit to be gained from PT. This is seen clearly when comparing the performance of RandInit (structured) and SSL (structured). This phenomenon has been observed in prior work (McDermott et al., 2021).
2. **SMD SSL performs effectively.** On the Elevated mPAP task (both datasets) and 24hr mortality task (Dataset 2), SMD SSL, either with the SimCLR or VICReg loss function, obtains the best performance among all methods.
3. **SMD SSL vs single-level SSL.** When compared to single-level SSL, we observe improvements when using SMD SSL on Elevated mPAP for both SimCLR and VICReg. However, this is not the case for 24hr Mortality – for example, component-only SSL with SimCLR on this task performs better than SMD SSL. This may arise because structured data PT does not improve performance significantly, and so it is preferable to only pre-train the signals encoder rather than the entire model.

4.3.3. Further Results

Additional evaluation. In Appendix B.3, we present three additional experiments: (1) comparing SMD SSL to other global-only baselines using different loss functions (NCL, CLOCS, and SACL), finding that SMD SSL improves on these methods; (2) comparing linear evaluation to full FT for different methods, finding that full FT improves on linear evaluation; and (3) studying the effect of longer PT times, finding that performance does not improve.

Component loss weight sensitivity. Considering SMD SSL (SimCLR) in the unimodal setting, we fix the global loss weight $\alpha = 1.0$ and vary the component loss weight β for the Elevated mPAP task. The validation set AUROC results are shown in Figure 3. The optimal value of the component loss weight is higher for Dataset 1, perhaps because the

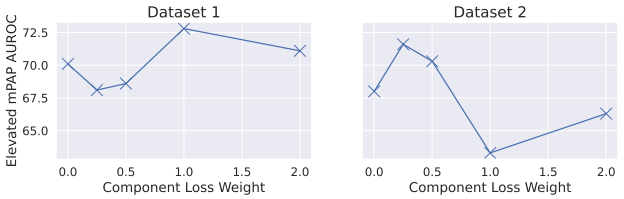

Figure 3: **Studying sensitivity to the component loss weight.** We find a higher optimal loss weight on Dataset 1 compared to Dataset 2, possibly due to Dataset 1 having more complex signals.

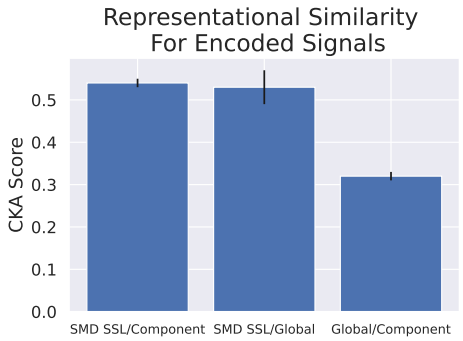

Figure 4: **Learned representations from SMD SSL are similar to both component-only and global-only SSL.** Comparing learned representations of the signals using different SimCLR-based SSL strategies – SMD SSL, Global, and Component, using Centered Kernel Alignment (CKA). Global and Component SSL show low representational similarity (right-most bar) but SMD SSL shows higher similarity with both individually, suggesting that learned representations in SMD SSL encode aspects of both component and global SSL.

signals in Dataset 1 are more complex than the signals in Dataset 2 (higher sampling rate, multiple channels).

Representational similarity analysis. Using Centered Kernel Alignment (CKA) (Kornblith et al., 2019), we study the representations learned by the signal encoder on Dataset 2 under different SSL methods. Our findings are: (1) representations from SMD SSL encode aspects of both component-only SSL and global-only SSL (illustrated in Figure 4); (2) the component loss appears to have more effect in the earlier layers of the signal encoder, whereas the global loss has more effect in later layers (details in Appendix B.3).

5. Scope and Limitations

Intended use-case. Our method is most appropriate in settings where we have multimodal trajectory data for patients, i.e., both structured data and ECGs are available. This use case is driven by our target application of monitoring patients with cardiovascular disease (Dataset 1), where the majority of patients have structured data and ECGs. In datasets where a small proportion of patients have multimodal data (such as in Dataset 2), it may be preferable to use other unimodal SSL approaches so that patients without multimodal data can still be included in model development. Although Dataset 2 does not exactly match our intended use case, we still considered it because it is publicly available and well-studied in related work.

Choice of data augmentations. SMD SSL uses data augmentations to generate different views. Our main contribution is in our two-level loss function, so we did not investigate novel augmentation strategies, instead leveraging existing effective data augmentations for clinical data. Our framework could equally apply with different augmentations or multiview generation approaches.

Additional data modalities. Our experiments focused on clinical time series consisting of a sequence of structured data and high-dimensional physiological signals; we did not include predictive information that may be available from other modalities, such as medical imaging. SMD SSL could be readily extended to this scenario, and it could be a valuable direction to explore.

6. Conclusion

In this work, we outlined a self-supervised learning (SSL) strategy for complex clinical time series where individual timesteps in the sequence also contain high-dimensional information, such as physiological signals. Our method, Sequential Multi-Dimensional SSL (SMD SSL), encourages effective pre-training on both a component level (level of individual signals) and a global level (level of the entire sequence). In experiments on two clinical datasets, pre-training with SMD SSL and then fine-tuning improves performance on downstream tasks compared to baselines. Future work could extend SMD SSL's component level loss into multiple levels by adopting frameworks from recent work (Tonekaboni et al., 2022; Bardes et al., 2022b). This could induce more structure in the pre-training phase, potentially improving performance.

Social impact. Our contribution in this work is mostly methodological. However, given that our application domain is in medicine, a high-risk setting, our method must be thoroughly validated in larger retrospective and prospective studies before any real-world use. This is to understand any potential risks from its use in practice.

Acknowledgements

This work was supported in part by funds from Quanta Computer, Inc. The authors thank the members of the Clinical and Applied Machine Learning group at MIT, the members of the Computational Cardiovascular Research Group at MIT, Paige Stockwell, Neel Dey, and the reviewers for helpful feedback and assistance with the work.

References

Bahri, D., Jiang, H., Tay, Y., and Metzler, D. SCARF: Self-Supervised Contrastive Learning using Random Feature Corruption. In *International Conference on Learning Representations*, 2021.

Bardes, A., Ponce, J., and LeCun, Y. VICReg: Variance-Invariance-Covariance Regularization For Self-Supervised Learning. In *ICLR*, 2022a.

Bardes, A., Ponce, J., and LeCun, Y. VICRegL: Self-Supervised Learning of Local Visual Features. In *NeurIPS*, 2022b.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020a.

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., and Hinton, G. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020b.

Chen, X. and He, K. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15750–15758, 2021.

Chen, X., Fan, H., Girshick, R., and He, K. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020c.

Cheng, J. Y., Goh, H., Dogrusoz, K., Tuzel, O., and Azemi, E. Subject-aware contrastive learning for biosignals. *arXiv preprint arXiv:2007.04871*, 2020.

Diamant, N., Reinertsen, E., Song, S., Aguirre, A., Stultz, C., and Batra, P. Patient contrastive learning: a performant, expressive, and practical approach to ecg modeling. 2021.

Goldberger, A., Amaral, L. A., Glass, L., Hausdorff, J. M., Ivanov, P., Mark, R., Mietus, J., Moody, G., Peng, C., and Stanley, H. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *Circulation*, 101 23:E215–20, 2000.

Gopal, B., Han, R. W., Raghupathi, G., Ng, A. Y., Tison, G. H., and Rajpurkar, P. 3KG: Contrastive Learning of 12-Lead Electrocardiograms using Physiologically-Inspired Augmentations. 2021.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Harutyunyan, H., Khachatrian, H., Kale, D. C., Ver Steeg, G., and Galstyan, A. Multitask learning and benchmarking with clinical time series data. *Scientific data*, 6(1): 1–18, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

Iwana, B. K. and Uchida, S. An empirical survey of data augmentation for time series classification with neural networks. *Plos one*, 16(7):e0254841, 2021.

Johnson, A. E., Pollard, T. J., Shen, L., Lehman, L.-w. H., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Anthony Celi, L., and Mark, R. G. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2014.

Kiyasseh, D., Zhu, T., and Clifton, D. A. CLOCS: Contrastive learning of cardiac signals across space, time, and patients. In *International Conference on Machine Learning*, pp. 5606–5615. PMLR, 2021.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. Similarity of neural network representations revisited. *arXiv preprint arXiv:1905.00414*, 2019.

Levin, R., Cherepanova, V., Schwarzschild, A., Bansal, A., Bruss, C. B., Goldstein, T., Wilson, A. G., and Goldblum, M. Transfer learning with deep tabular models. *arXiv preprint arXiv:2206.15306*, 2022.

Li, Y., Mamouei, M., Salimi-Khorshidi, G., Rao, S., Hassaine, A., Canoy, D., Lukasiewicz, T., and Rahimi, K. Hi-behrt: Hierarchical transformer-based model for accurate prediction of clinical events using multimodal longitudinal electronic health records. *arXiv preprint arXiv:2106.11360*, 2021.

McDermott, M., Nestor, B., Kim, E., Zhang, W., Goldenberg, A., Szolovits, P., and Ghassemi, M. A comprehensive ehr timeseries pre-training benchmark. In *Proceedings of the Conference on Health, Inference, and Learning*, pp. 257–278, 2021.

Moody, B., Moody, G., Villarroel, M., Clifford, G., and Silva, I. MIMIC-III waveform database, 2020.

Oh, J., Chung, H., Kwon, J.-m., Hong, D.-g., and Choi, E. Lead-agnostic self-supervised learning for local and global representations of electrocardiogram. In *Conference on Health, Inference, and Learning*, pp. 338–353. PMLR, 2022.

Raghu, A., Lorraine, J., Kornblith, S., McDermott, M., and Duvenaud, D. K. Meta-learning to improve pre-training. *Advances in Neural Information Processing Systems*, 34, 2021.

Raghu, A., Shanmugam, D., Pomerantsev, E., Guttag, J., and Stultz, C. M. Data augmentation for electrocardiograms. In *Conference on Health, Inference, and Learning*, pp. 282–310. PMLR, 2022.

Raghu, A., Schlesinger, D., Pomerantsev, E., Devireddy, S., Shah, P., Garasic, J., Guttag, J., and Stultz, C. M. Ecg-guided non-invasive estimation of pulmonary congestion in patients with heart failure. *Scientific Reports*, 13(1): 3923, 2023.

Ren, M., Dey, N., Styner, M. A., Botteron, K., and Gerig, G. Local spatiotemporal representation learning for longitudinally-consistent neuroimage analysis, 2022. URL https://arxiv.org/abs/2206.04281.

Schlesinger, D. E., Diamant, N., Raghu, A., Reinertsen, E., Young, K., Batra, P., Pomerantsev, E., and Stultz, C. M. A deep learning model for inferring elevated pulmonary capillary wedge pressures from the 12-lead electrocardiogram. *JACC: Advances*, 1(1):100003, 2022.

Tipirneni, S. and Reddy, C. K. Self-supervised transformer for sparse and irregularly sampled multivariate clinical time-series. *ACM Trans. Knowl. Discov. Data*, 1(1), 2022.

Tiu, E., Talius, E., Patel, P., Langlotz, C. P., Ng, A. Y., and Rajpurkar, P. Expert-level detection of pathologies from unannotated chest x-ray images via self-supervised learning. *Nature Biomedical Engineering*, 6(12):1399–1406, September 2022. doi: 10.1038/s41551-022-00936-9. URL https://doi.org/10.1038/s41551-022-00936-9.

Tonekaboni, S., Eytan, D., and Goldenberg, A. Unsupervised representation learning for time series with temporal neighborhood coding. *arXiv preprint arXiv:2106.00750*, 2021.

Tonekaboni, S., Li, C.-L., Arik, S. O., Goldenberg, A., and Pfister, T. Decoupling local and global representations of time series. In Camps-Valls, G., Ruiz, F. J. R., and Valera, I. (eds.), *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 8700–8714. PMLR, 28–30 Mar 2022. URL https://proceedings.mlr.press/v151/tonekaboni22a.html.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. Extracting and composing robust features with denoising autoencoders. *Proceedings of the 25th international conference on Machine learning*, 2008. doi: 10.1145/1390156.1390294. URL https://doi.org/10.1145/1390156.1390294.

Vu, Y. N. T., Wang, R., Balachandar, N., Liu, C., Ng, A. Y., and Rajpurkar, P. MedAug: Contrastive learning leveraging patient metadata improves representations for chest X-ray interpretation. 149:755–769, 06–07 Aug 2021. URL https://proceedings.mlr.press/v149/vu21a.html.

Weatherhead, A., Greer, R., Moga, M.-A., Mazwi, M., Eytan, D., Goldenberg, A., and Tonekaboni, S. Learning Unsupervised Representations for ICU Timeseries. In *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pp. 152–168. PMLR, 07–08 Apr 2022.

Yèche, H., Dresdner, G., Locatello, F., Hüser, M., and Rätsch, G. Neighborhood contrastive learning applied to online patient monitoring. In *International Conference on Machine Learning*, pp. 11964–11974. PMLR, 2021.

Zhang, X., Zhao, Z., Tsiligkaridis, T., and Zitnik, M. Self-supervised contrastive pre-training for time series via time-frequency consistency. In *Proceedings of Neural Information Processing Systems, NeurIPS*, 2022.

A. Augmentation Functions for Sequential Multi-Dimensional Self-Supervised Learning

In this section, we specify more details about augmentation functions used in our method.

A.1. Augmentation Details

We form an augmented trajectory by separately augmenting each of the data modalities within the trajectory, using the following approach for each data type.

High-frequency signal s: For each signal in the trajectory of length T, we form a pair of augmented views by first splitting the signal into two disjoint segments and then applying random masking and noise addition as augmentations to each view independently, similar to the approach used in CLOCS (Kiyasseh et al., 2021). The intuition is that two segments of a signal that are close in time should encode similar physiology, and can therefore be considered paired views. Random masking and noise addition are commonly used as time-series augmentations (Gopal et al., 2021; Zhang et al., 2022; Raghu et al., 2022; Iwana & Uchida, 2021). In more detail:

- Signal splitting: We split a raw signal of 30 seconds into two disjoint 10 second segments for the first phase of the augmentation process.
- Random signal masking: we choose a random 25% of the signal to set to zero – this was found to overall be more effective than masking proportions of 10% and 50%.
- Noise addition: we add Gaussian noise with standard deviation 0.25 to the signal.

Structured-time series data w: The tabular data sequence forms a $T \times M$ matrix over all timesteps of the trajectory. Following prior work (Yèche et al., 2021), we apply two data augmentation strategies to this matrix: history cutout and noise addition. In more detail:

- History cutout: For each feature, with probability 0.25, randomly set 25% of the timesteps in that timeseries to be missing. Forward fill impute this value. This mirrors the imputation strategy used in our raw data. Unlike in Yèche et al. (2021), we use forward filling rather than replacing with zeros, because our time series are much shorter (8 timesteps rather than 48) and therefore replacing with zeros destroyed too much information. Using more aggressive cutout augmentations was found to worsen performance, likely because they destroyed too much information in the data. This is in general a challenge when using relatively short time series.
- Noise addition: For each feature, add Gaussian noise with standard deviation equal to 10% of the standard deviation of that feature's values in the training set.

Static features d: Following Yèche et al. (2021), we use random dropout and noise addition. Other corruption strategies (e.g., Bahri et al. (2021)), were found to be less effective, potentially due being too strong (also seen in Levin et al. (2022)). We randomly drop out 25% of the features (impute with the mean value) and add add Gaussian noise with standard deviation equal to 10% of the standard deviation of that feature's values in the training set.

A.2. Other Augmentations

Early on in our experiments, we investigated other augmentation functions such as channel dropout for the structured time-series data (Yèche et al., 2021) and more complex signal augmentations, such as random lead masking (Oh et al., 2022). However, we found the improvements from these to be inconsistent and the hyperparameters to be difficult to tune, so we opted for this more focused set of augmentations.

B. Further Experimental Details

In this section, we provide further details about our experiments. We first describe more about the datasets and data preprocessing. We then provide further information on the model architecture for our method and baselines, and discuss our hyperparameter search and settings. We then provide additional quantitative results and representational similarity analysis.

B.1. Dataset Details

As discussed in the main text, we consider two clinical datasets in our experiments:

- **Dataset 1** is a private dataset derived from the electronic health record (EHR) of the Massachusetts General Hospital

Table 4: Dataset statistics.

	Dataset 1		Dataset 2	
Task	# Patients	# Trajectories	# Patients	# Trajectories
Pre-Training	8888	43858	5022	26615
Elevated mPAP	2025	48511	500	14957
24hr mortality	9605	57758	5689	318306

(MGH), consisting of a cohort of patients with a prior diagnosis of heart failure. For each patient, we have structured data from the EHR and physiological signals measured by a bedside telemetry monitor. These signals include vitals signs such as heart rate (HR) and oxygen saturation (SpO2), measured at a low frequency (0.5 Hz), and waveforms such as the electrocardiogram (ECG), measured at a high frequency (240 Hz).
This dataset was obtained with IRB approval (protocol number 2020P003053). Since the dataset has some identifiable information, all computations are performed on a server that sits behind the hospital firewall. Due to restrictions surrounding its use, this dataset cannot be released at this stage.

- **Dataset 2** is a public dataset derived from the commonly used MIMIC-III clinical database (Johnson et al., 2016; Goldberger et al., 2000) and its associated database of physiological signals (Moody et al., 2020). The clinical database contains structured data over a patient's stay, and the physiological signals database contains vitals signs (HR, SpO2) and waveforms (ECG) measured by a bedside telemetry monitor.
  We use the widely adopted preprocessing pipeline introduced in Harutyunyan et al. (2019) to form the specific cohort and extract the structured data features used in modeling. This pipeline also provides the functionality to create development and testing sets for the different downstream tasks we consider.
  The clinical database is available on PhysioNet (Goldberger et al., 2000) to credentialed users. The database of physiological signals is open-access on PhysioNet.

Constructing PT and FT sets. As outlined in Section 4.1, both datasets consist of a number of hospital visits, which we resample at hourly resolution. We extract 30 seconds of the high-dimensional physiological signals at each hour marker from the raw data store.

To generate PT trajectories from these resampled visits, we first split each visit into non-overlapping contiguous 12 hour blocks. A PT trajectory is formed by first sampling a 12 hour block from all the extracted blocks, and then selecting 8 contiguous timesteps from the sampled block (with the starting timestep selected randomly). This trajectory construction strategy has implications in terms of the negative samples in both losses:

- **Global loss.** Consider a sampled anchor trajectory from a given patient i, timesteps $1-8$. When computing the global loss, the negative pairs for that anchor trajectory are either: (1) a trajectory from a different patient $j \neq i$; or (2) or a trajectory from that same patient i starting after timestep 8.
- **Component loss.** When computing the component loss, recall that for an anchor signal, other signals from the same trajectory are *not* used as negatives, in order to minimize correlation between the anchor and negatives. Therefore, negative pairs for an anchor signal are either signals from a different patient, or signals from that same patient from further off in time.

This strategy of sampling trajectories that do not overlap was applied to ensure that we do not use highly correlated trajectories/signals as negatives. We note that this strategy is not necessarily optimal, and that different approaches could be used for both the component and global loss terms. Our framework could easily be used with these other sampling strategies and loss formulations.

To generate FT sets, we first use a sliding window to select contiguous 8 hour blocks at 1 hour increments from each visit. Each of these contiguous 8 hour blocks becomes a trajectory in the FT dataset. The trajectory labels are formed based on the nature of the specific task. For example, for the 24 hour mortality task, the label is based on whether the patient dies within 24 hours of the ending time of that trajectory.

Trajectory Features and Preprocessing. The trajectories in PT and FT sets consist of static features d and a time-series of structured data and physiological signals $\{(w_t, s_t)\}_{t=1}^{T}$, as presented in Section 3.1.

In Dataset 1, $d \in \mathbb{R}^9$ contains the following features from the EHR: BUN, Chloride, CO2, Creatinine, Glucose, Potassium,

Sodium, Systolic Blood Pressure, Diastolic Blood Pressure. We take the average if multiple values are recorded in each time window. $w_t \in \mathbb{R}^{30}$ has mean, standard deviation, maximum, and minimum of heart rate and SpO2 recorded within each hour window (this is sourced from the telemetry monitor), and also 22 heart rate variability features from the ECG recorded by the telemetry monitor during each time window. $s_t \in \mathbb{R}^{4\times 2400}$ is a 10 second 4-channel ECG measured at 240 Hz, containing leads I, II, III, and V1, extracted from a longer ECG measured by the telemetry monitor during that hour window.

In Dataset 2, $d \in \mathbb{R}^{38}$ contains the following features from the EHR: FiO2, Glucose, Temperature, pH, and one-hot encoded Glasgow Coma Scale measures, following Harutyunyan et al. (2019). $w_t \in \mathbb{R}^{13}$ contains the following information from the physiological signals database: mean, standard deviation, maximum, and minimum of heart rate and SpO2 recorded within each hour window from the telemetry monitor, diastolic blood pressure, systolic blood pressure, mean blood pressure, heart rate, and SpO2 from the EHR. $s_t \in \mathbb{R}^{1\times 1250}$ is a 10 second 1-channel ECG measured at 125 Hz, containing lead II, extracted from a longer ECG measured by the telemetry monitor during that hour window.

Missing structured data are forward-fill imputed where possible (for example, if part of a time series) and otherwise imputed with the mean over the training dataset. Missing signals are represented with zeros. We drop any trajectories that have more than 1 timestep with a missing signal. For Dataset 2, we note that by dropping any trajectory with more than 1 timestep with a missing signal, we have a smaller dataset than in Harutyunyan et al. (2019).

Forming labels. The FT labels for the Elevated mPAP task are formed based on the PA pressure waveform recorded for patients (when available) – if the mean pressure is over 20 mmHg over a 1 minute period at the final timestep of the trajectory, we assign a binary label of 1, and else 0. For the 24 hour mortality task, we use the recorded time of death recorded in the EHR and if it is less than 24 hours from the end time of the trajectory, we assign a label of 1, and otherwise 0. This is as was done in Harutyunyan et al. (2019).

B.2. Experimental Setup

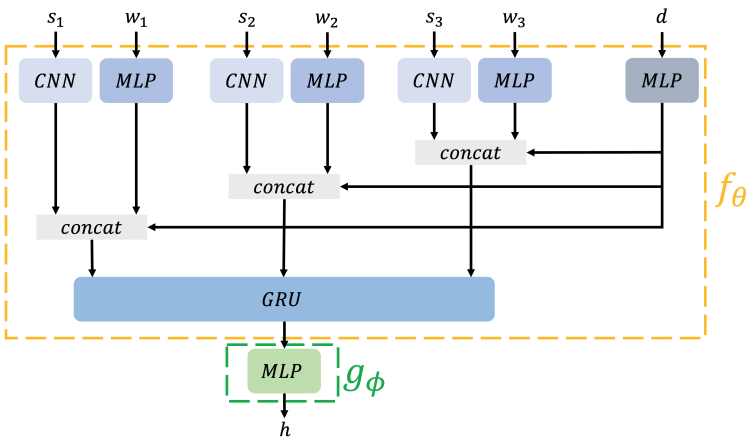

Figure 5: **Model architecture used in our experiments.** We show the architecture used to model trajectories in a scenario where the input trajectory has 3 timesteps.

B.2.1. Model Architecture

We use the following architecture for the encoder and projection head for all methods:

- **Encoder**: Each signal s_t in the trajectory is passed through a ResNet-styled 1-D CNN encoder with global average pooling. We base our CNN encoder model off a ResNet-18 architecture with kernel size of 15. Following global average pooling over the temporal dimension, each signal is projected into 128 dimensions with a linear layer.
  The structured data w_t at each timestep is embedded with a 2-layer fully-connected network with 128 hidden units and ReLU activation at each layer, and this embedding is then concatenated with the signal embedding. The static features d are passed through a different 2-layer fully-connected network with 128 hidden units and ReLU activation at each layer, and then concatenated with the embeddings of the signal and structured data timeseries at each timestep. The resulting sequence of vectors is passed into a 4-hidden layer GRU with hidden size of 384, with the last hidden state of the GRU being used as the overall trajectory embedding vector.

- **Projection heads**: The two projection heads for the signal and the trajectory are both 2-layer fully connected networks with batch normalization and ReLU activation with 2048 hidden units. The trajectory projection head takes the last hidden state of the GRU as input, and the signal projection head takes the output of the signals encoder as the input. When using the NT-Xent loss, the resulting projection is normalized (Chen et al., 2020a) before computing the NT-Xent loss over the batch.

Figure 5 shows the model architecture in a scenario in which the input trajectory has 3 timesteps.

B.2.2. SSL METHODS: IMPLEMENTATION

We describe the implementation of the SimCLR and VICReg methods in the main paper, Section 3. For SimSiam, we follow the setup in Chen & He (2021) and use a predictor network in the trainable branch, and minimize cosine distance between the output of the predictor network in the trainable branch and the output of the projection head in the stop-gradient branch.

For simplicity, we let this predictor network have the same architecture as the projection head – a two layer fully connected network with batch normalization and ReLU activation, with 2048 hidden units. We did not find a bottleneck structure to improve performance in initial investigations, but further experiments may be warranted here.

B.2.3. LOSS, ARCHITECTURE, AND OPTIMIZATION HYPERPARAMETERS

There are various hyperparameters to tune, such as learning rates, loss weighting for VICReg loss terms, and loss weighting for SMD SSL. Evaluating many hyperparameter settings is very computationally expensive (since it entails doing both a PT and FT run), so we conduct a reduced search on a subset of the hyperparameters focusing only on the Elevated mPAP task in the unimodal setting, optimizing validation AUROC.

In our hyperparameter search, we use the following setup:

- Learning rate: tune on a randomly initialized model for the elevated mPAP task on each dataset, and then use this learning rate for all other experiments. We compared Adam with a learning rate of 1e-4, 3e-4, 1e-3, and 3e-3. We found 1e-3 to be the most stable and best performing.
- VICReg loss weights: Tune these for the VICReg (global) model only, and use the best hyperparameters for all other uses of the VICReg loss, including SMD SSL (VICReg). Following the original paper, we set the covariance weight $\nu = 1$ and then tune the invariance weight λ and variance weight μ. We found in early experiments that the variance weight did not have much impact on performance, and so focused on the invariance weight, studying $\lambda = 1, 2, 5$. We found $\lambda = 1$ to perform the best on both datasets.
- SMD SSL (SimCLR) component loss weight: Set the global weight $\alpha = 1$ in Eqn. 5, and tune the component weight β, on both datasets separately, comparing $\beta = 0.25, 0.5, 1.0, 2.0$. We found $\beta = 1.0$ to perform the best on Dataset 1, and $\beta = 0.25$ to perform the best on Dataset 2.
- SMD SSL (VICReg) component loss weight: Use the best VICReg loss weights found above, set the global weight $\alpha = 1$ in Eqn. 5, and tune the component loss weight β, comparing $\beta = 0.1, 0.25, 0.5, 1.0, 2.0$. We found $\beta = 1.0$ to perform the best on Dataset 1, and $\beta = 0.1$ to perform the best on Dataset 2.

We fixed the temperature of the NT-Xent loss to 0.1, following Yèche et al. (2021).

We did not conduct tuning of the architecture hyperparameters, and instead opted to use architectural choices that were found to be effective in previous works, such as a ResNet signal encoder (Raghu et al., 2021; 2022), a wide projection head (Chen et al., 2020b; Bardes et al., 2022a), and a GRU sequence model (McDermott et al., 2021). Similar to McDermott et al. (2021), we did not find a transformer model to be beneficial as the sequence model, though perhaps architectural tuning could improve its performance.

B.2.4. COMPUTE DETAILS

All models were trained on either a single NVIDIA Quadro RTX 8000 or a single NVIDIA RTX A6000 GPU. Pre-training takes about 8 hours on Dataset 1 and about 2 hours on Dataset 2. Fine-tuning on Dataset 1 tasks takes about 4 hours. Fine-tuning in Dataset 2 on Elevated mPAP takes about 30 minutes, and about 4 hours on 24hr Mortality. Pre-training uses approximately 20 GB of GPU memory, and fine-tuning uses approximately 10 GB of GPU memory.

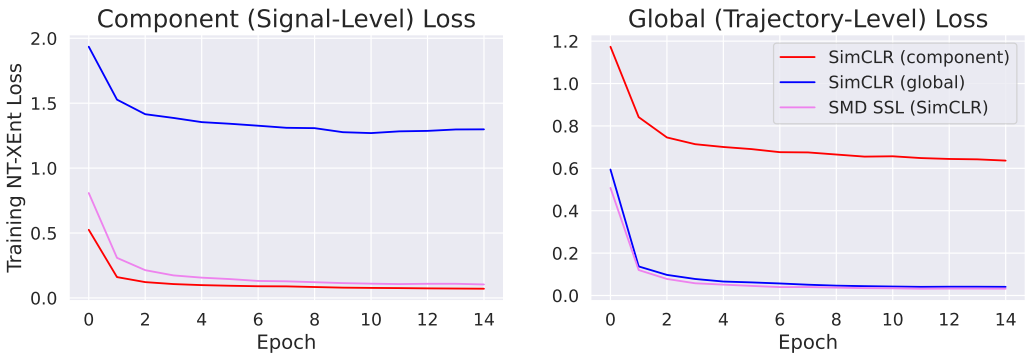

Figure 6: **Studying training loss curves for SMD SSL and variations.** We observe that SMD SSL effectively minimizes both the component and global NT-XEnt losses during training. Interestingly, we observe that the NT-Xent loss computed on the signal level and trajectory level reduces somewhat even when it is not explicitly minimized. To see this, consider the SimCLR (global) method and the component Loss – the component loss reduces over training even with a random projection head, without adding this term to the objective. This indicates that the global loss and component loss are not entirely independent (as is expected).

Table 5: Mean and 95% confidence interval of AUROC on fine-tuning tasks in the unimodal setting when only using structured data, comparing no PT (RandInit) to PT with the SimCLR and VICReg global losses. We find that when considering structured data alone, PT does not offer much benefit to performance; however, there are improvements seen in the Elevated mPAP task.

(a) Results on Dataset 1.

	Elevated mPAP	24hr Mortality
RandInit	65.3 $\pm$ 0.1	**79.0** $\pm$ **0.4**
SimCLR	**66.8** $\pm$ **0.1**	**79.0** $\pm$ **0.4**
VICReg	66.0 $\pm$ 0.0	77.9 $\pm$ 0.4

(b) Results on Dataset 2.

	Elevated mPAP	24hr Mortality
RandInit	65.0 $\pm$ 0.3	**90.1** $\pm$ **0.1**
SimCLR	66.8 $\pm$ 0.3	88.1 $\pm$ 0.1
VICReg	**68.1** $\pm$ **0.3**	89.3 $\pm$ 0.1

B.3. Additional Results

Studying loss curves. Figure 6 shows training loss curves for SimCLR-based models on Dataset 2. We observe that SMD SSL effectively minimizes both component and global losses over training. Considering the component loss alone (left plot), we see that this loss naturally reduces during training of the SimCLR (global) model even though this is not explicitly enforced during model training – we compute the component loss in this case with a randomly initialized signal level projection head that is not updated, and the network parameters are also not updated to minimize this component loss. An analogous situation with SimCLR (component) and the global loss is seen in the right plot. This suggests that training with one of the losses does encourage structure in both representation spaces, even with random projections, but this structure is more clearly defined when the loss is explicitly minimized (as in SMD SSL).

Trends in AUPRC. Since the mortality task has low prevalence, we study trends in AUPRC among methods as they compare to AUROC. We find that in the unimodal setting, on both datasets, our objective improves AUPRC by 1-2% over baselines, but AUPRCs are all relatively low (<5%) due to the limited predictive signal in the ECGs alone for the mortality prediction task. In the multimodal setting, on Dataset 1, SMD SSL worsens AUPRC over the single-level approach by 2.9% AUPRC (10.8% vs 7.9%) – this is consistent with what was seen with AUROC. On Dataset 2, the AUPRC with our approach is 28.4%, an improvement of about 4% over the best baseline.

Unimodal experiments with structured data. Table 5 shows results when training on only the structured data (structured time-series and static vector) in the trajectory. We find that the Elevated mPAP task can benefit from pre-training, but the 24hr Mortality task performance is not boosted by pre-training. This is likely because the structured data are relatively simple and low-dimensional, and there is enough data to learn useful predictive information from these data as-is, without pre-training.

Additional baselines. As discussed in the main text, related SSL strategies for time series data are not exactly applicable in our setting since they formulate pipelines for structured data-only time series (rather than multimodal time series), or are concerned with individual physiological waveforms (rather than sequences of waveforms).

Table 6: Test AUROC of different SSL algorithms on Dataset 2 (MIMIC) in the multimodal setting. We observe that SMD SSL (VICReg) improves on these additional SSL baselines.

	Elevated mPAP	24hr Mortality
SMD SSL (VICReg)	71.6	90.7
NCL (Structured data only)	67.6	87.7
NCL (Multimodal)	65.5	89.9
SACL	64.7	89.8
CLOCS	64.3	89.9

Table 7: Comparing test AUROC of the two evaluation paradigms — Linear Evaluation and Full Fine-tuning (FT) — with selected SSL algorithms on Dataset 2 (MIMIC) in the multimodal setting. We find that Full FT routinely performs better than linear evaluation.

	Elevated mPAP		24hr Mortality	
	Full FT	Linear Evaluation	Full FT	Linear Evaluation
RandInit	65.3	N/A	87.8	N/A
SMD SSL (VICReg)	71.6	65.0	90.7	72.2
VICReg (Global)	70.4	64.4	87.8	71.7
SimCLR (Global)	63.7	63.1	86.8	71.7
SimSiam (Component)	67.4	61.6	90.6	71.9
SimSiam (Global)	60.6	50.6	90.4	50.0

Despite these differences, for completeness, we study here the performance of adapted versions of three related methods from the literature for Dataset 2 (MIMIC-III), focusing on the multimodal setting. Specifically, we evaluate the SSL objective functions proposed in NCL (Yèche et al., 2021), SACL (Cheng et al., 2020), and CLOCS (specifically the CMSC formulation) (Kiyasseh et al., 2021). We use each of these losses in a global-only SSL setup in order to compare how they perform to our proposed two-level loss function. We additionally evaluate structured data-only NCL.

We evaluate these methods using the same experimental setup (augmentation pipeline, optimization hyperparameters, model architecture, etc) as what was used when evaluating our method. For NCL, we considered two values of α (0.3 and 0.5), and fixed $w = 16$ as in the original paper's configuration on MIMIC. We select the value of α that obtained the best validation AUROC on each downstream task.

Results are shown in Table 6. As seen, the best two-level approach, SMD SSL (VICReg), outperforms the different baselines. This indicates that a two-level loss is not easily outperformed by other global-only loss functions. An important investigation is to conduct a more thorough hyperparameter search for these alternative loss functions, and also evaluate whether two-level versions of these other objectives could improve on SMD SSL (VICReg).

Linear Evaluation vs Full Fine-tuning. As discussed in the main paper, our goal is to develop a self-supervised pre-training algorithm that finds an effective model initialization for adaptation to downstream tasks (i.e., a transfer learning setting). As a result, we evaluate both full FT and linear evaluation, reporting the evaluation strategy that obtains the best validation AUROC on a per-method and per-task basis. It is important to consider full FT since it almost always outperforms linear evaluation – we observed this in our results, and a similar finding was see in the evaluation from McDermott et al. (2021). Table 7 highlights this finding for a subset of the methods on Dataset 2 (MIMIC), in the multimodal setting.

Studying longer Pre-training. On Dataset 2 (MIMIC), we pre-trained methods for longer (50 epochs) and compared performance after 15 and 50 epochs, following the best of full FT and linear evaluation (following our standard experimental setup). Results are in Table 8, indicating that performance did not improve following longer PT.

Additional representational similarity experiments. In the main text, we presented a simple representational similarity study examining the how the learned representations by the CNN signals encoder compared in SMD SSL (SimCLR) vs. SimCLR (component only) and SimCLR (global only) pre-training. We found that SMD SSL representations had reasonable Centered Kernel Alignment (CKA) similarity (Kornblith et al., 2019) with both component-only and global-only PT. On the other hand, global-only and component-only PT were quite dissimilar.

Table 8: Comparing test AUROC after different amounts of PT with selected SSL algorithms on Dataset 2 (MIMIC) in the multimodal setting. Performance does not appear to improve after more PT.

	Elevated mPAP		24hr Mortality	
	15 epochs PT	50 epochs PT	15 epochs PT	50 epochs PT
SMD SSL (VICReg)	71.6	66.6	90.7	89.3
VICReg (Global)	70.4	63.2	87.8	78.8
SimSiam (Component)	67.4	59.6	90.6	78.8
SimSiam (Global)	60.6	51.1	90.4	88.2

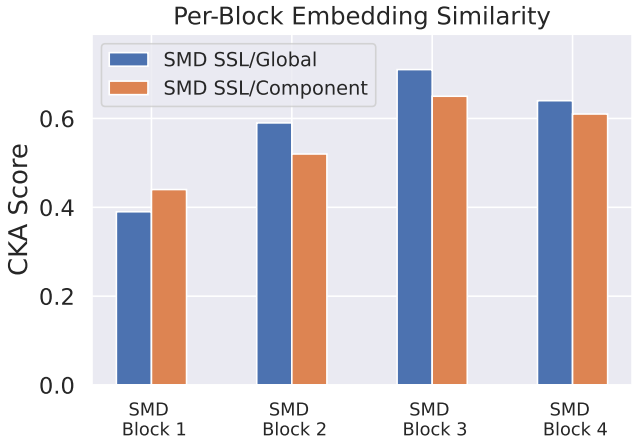

Figure 7: **Studying per-block representational similarity in the CNN encoder between SMD SSL pre-training and component-only and global-only pre-training.** SMD SSL representations are more similar to component-only PT early on in the CNN signals encoder, and more similar to global-only PT deeper in the network.

To further understand the effect of training with the component and global losses in SMD SSL, we conduct a finer-grained CKA study. We take the output of each residual block of the CNN signals encoder and compare the CKA similarity in these representations (following pooling over the sequence dimension) to the CKA similarity of component-only and global-only PT models, on a per-block basis. That is, we average the CKA similarity between SMD SSL (block i) and component-only PT (blocks $1, 2, 3, 4$), and similarly for global-only PT. The results are shown in Figure 7. We see that SMD SSL PT has more similarity with component-only PT in the first block, and greater similarity with global-PT in the remaining blocks. This suggests that the component loss is having the most impact on SMD SSL representations in the earlier CNN layers, indicating that these low-level features are particularly relevant for minimizing the component loss – this makes sense, since we would expect lower-level features to matter more in a per-signal embedding, and global-level features to matter more in a sequence-level embedding.