Contrastive Pre-Training for Multimodal Medical Time Series

Anonymous Author(s)
Affiliation
Address
email

Abstract

Clinical time series data are highly rich and provide significant information about a patient's physiological state. However, these time series can be complex to model, particularly when they consist of multimodal data measured at different resolutions. Most existing methods to learn representations of these data consider only tabular time series (e.g., lab measurements and vitals signs), and do not naturally extend to modelling a full, multimodal time series. In this work, we propose a contrastive pre-training strategy to learn representations of multimodal time series. We consider a setting where the time series contains sequences of (1) high-frequency electrocardiograms and (2) structured data from labs and vitals. We outline a strategy to generate augmentations of these data for contrastive learning, building on recent work in representation learning for medical data. We evaluate our method on a real-world dataset, finding it obtains improved or competitive performance when compared to baselines on two downstream tasks.

1 Introduction

In various clinical settings, such as the intensive care unit (ICU), patients are closely monitored, resulting in time series data being generated for each patient. This highly rich data contains significant physiological information about a patient's state and their disease progression over time [7]. As a result, there have been efforts to conduct representation learning on these data, with the goal being to use these representations for various downstream predictive tasks [9, 12, 11, 13].

Although these works develop effective strategies to model clinical time-series data, they focus on learning representations of a time-series of unimodal, structured data alone (e.g. lab values). However, in reality, data originating from a patient's encounter is significantly more complex, containing several modalities of time-series measured at different resolutions. For example, a given patient may have two distinct, complementary types of time-series recorded at periodic intervals: (1) high-frequency physiological signals (e.g., a 10 second electrocardiogram recorded at 240 Hz); and (2) structured labs and vitals signs. Extending existing methods to learn representations of these complex, hierarchical time-series is challenging, since they neither model sequences of high-frequency waveforms nor contend with the multimodal nature of the data stream.

In this paper, we take steps towards addressing this gap and outline an approach for learning representations of multimodal clinical time series. We describe a contrastive pre-training strategy where augmentations of the multimodal time series are generated by combining methods from recent work on representation learning for medical data [13, 8]. We evaluate our pre-training strategy on a real-world dataset of heart failure patients from a large tertiary hospital. Our initial results suggest that our approach yields pretrained models that outperform or perform competitively with baselines on two downstream tasks of clinical interest.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

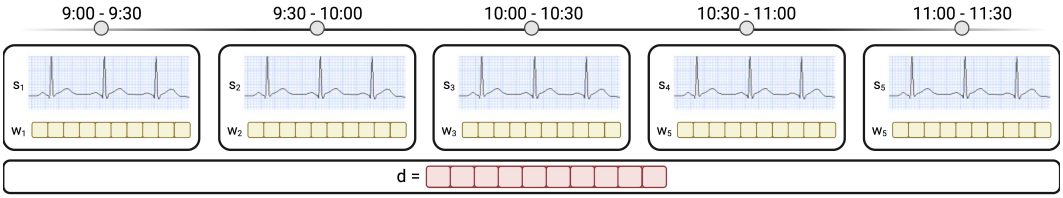

Figure 1: **An example of a multimodal clinical time-series or 'trajectory'.** The trajectory τ_n is characterized by an overall static vector d consisting of measurements that remain constant, and a time series of signals s_t and structured data w_t measured at each time step. Here, each time step is a 30 minute window.

Related work. Contrastive objectives encourage the learned representations of different views coming from the same sample to be similar, and from different samples to be dissimilar [4, 6, 2]. Multiview contrastive learning objectives can generalize to any modality of data given a procedure to generate different views of each sample. Here, we adopt the contrastive framework from SimCLR [3], and define procedures for generating augmentations of multimodal time-series.

Contrastive representation learning has previously been used to improve predictive performance on downstream clinical tasks [12, 11]. Prior work has focused on applying contrastive approaches to unimodal time-series, such as electrocardiograms [8, 5] and sequences of tabular data (labs and vitals) [13]. Here, we study contrastive learning on multimodal clinical data, building on existing approaches to enable their application to our time-series data with ECGs, labs, and vitals.

2 Methods

2.1 Problem Setup

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 our use-case in cardiovascular medicine, where patients may be monitored with telemetry devices that continuously record 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.

We frame the problem of representation learning as one of pre-training (PT). We assume access to a (large) PT dataset of patient trajectories, and a set of (likely smaller) fine-tuning (FT) datasets that have labels for downstream tasks. We first pre-train a model f_θ on the PT dataset, and then evaluate the model based on its performance after FT on the downstream tasks.

More formally, we denote the PT dataset as $\mathcal{D}_{\text{PT}} = \{\tau_n\}_{n=1}^{N_{\text{PT}}}$, which contains N_{PT} trajectories (denoted τ). A trajectory τ_n is of length T_n, and has the following structure: $\tau_n = (d, \{(w, s)_t\}_{t=1}^{T_n})$. 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 $\{(w, s)_t\}_{t=1}^{T_n}$ contains a vector of structured data $w \in \mathbb{R}^M$, and a high-dimensional signal $s \in \mathbb{R}^{C \times Z}$, where C is the number of signal channels and Z is the number of samples in the signal (typically on the order of a few thousand). A visualization is shown in Figure 1.

During FT we have a collection of K tasks indexed by k, each with an accompanying FT dataset of trajectories $\mathcal{D}_{\text{FT}}^{(k)} = \{(\tau_n, y_n\}_{n=1}^{N_{\text{FT}}^{(k)}}$. Each FT trajectory has an accompanying label y_n that the model is fine-tuned to predict. '24 hour mortality' is an example of a fine-tuning task, where the model predicts if a patient will die in the next day.

2.2 Contrastive Learning for Trajectories

We use a contrastive learning approach to pre-train the model f_θ. This is because contrastive approaches have previously been demonstrated to be (independently) effective on clinical signals data [8, 5] and structured data [13]. In particular, we adopt a multiview contrastive approach inspired by SimCLR [3]:

1. Sample a batch of trajectories from the PT dataset.

Task	# Patients	# Trajectories	Prevalence
Elevated mPAP	1,484	5,225	77.8%
24 hour mortality	10,990	15,835	1.3%

Table 1: **Statistics for fine-tuning tasks.**

2. For each trajectory τ_n, generate two augmented views of it.
3. Pass the augmented views through the representation model f_θ and a projection head g_ϕ.
4. Minimize the normalized temperature-scaled cross-entropy loss (NT-Xent) on these projections. The loss is averaged over all positive pairs that arise from the same underlying trajectory, as in Chen et al. [3]. The form of the loss function is detailed in the appendix.

Trajectory-level augmentation. This learning framework requires generating augmented views of each trajectory. We use the following approach for each type of data (further details in the appendix):

- **High-frequency signal s:** For each signal in the trajectory of length T_n, 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 Kiyasseh et al. [8]. The intuition is that two segments of a signal that are close in time should encode similar physiological state, and can therefore be considered paired views.
- **Structured-time series data w:** The tabular data sequence forms a $T_n \times M$ matrix over all timesteps of the trajectory. Following prior work [13], we apply two data augmentation strategies to this matrix: history cutout and channel dropout.
- **Static features d:** Following Bahri et al. [1], we obtain two views of by (1) randomly selecting features to corrupt; (2) corrupting them by replacing them values drawn from a uniform distribution over the values that feature takes in the training set.

We note that the individual aspects of our approach (contrastive loss function and augmentations for signals/tabular data) have been used in prior work; however, combining these together in a single pipeline for pre-training jointly on a time-series of signals and structured data is a new direction.

3 Experiments

We evaluate our method on a dataset from a large tertiary hospital. We assess whether our pre-training scheme on multimodal trajectories can improve predictive performance on downstream tasks over (1) no pre-training and (2) pre-training on structured data alone.

Dataset. Our cohort consists of a total of 51,921 trajectories from 10,990 unique patients with a diagnosis of heart failure. Each element in a trajectory corresponds to data from 30 minutes of a patient's stay, and for simplicity, we fix each trajectory to be of length 10. Within each trajectory, $d \in \mathbb{R}^9$ contains information from a basic metabolic panel and blood pressures. These values are measured relatively infrequently in our dataset (every 6-12 hrs) so we treat them as constant within a trajectory. $w_t \in \mathbb{R}^9$ represents summary statistics related to heart rate and SpO2 within each 30 minute window. $s_t \in \mathbb{R}^{4\times 7200}$ is a 4-channel ECG signal (leads I, II, III, and V1) measured for 30 seconds at 240 Hz within that 30 minute window. Further dataset details are in the appendix.

Downstream tasks. We evaluate models on two downstream tasks (Table 1):

- **Elevated mPAP**: Predict whether a patient's mean Pulmonary Arterial Pressure (mPAP) is abnormally high (greater than 20 mmHg) at a time marker, given the trajectory data until that point. This task is of clinical interest since the mPAP is typically only 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 [10] has studied a similar task from the 12-lead ECG, but not in the context of trajectory data, as we do here.
- **24 hour mortality:** Predict whether the patient is likely to die in the next day, given a trajectory from the start of a patient's admission. This task is commonly used to evaluate pre-training strategies on clinical time series data from the ICU [9, 13].

Methods evaluated. Motivated by our experimental goals, we evaluate the following methods:

Task (Metric)	RandInit	RandInit (No ECG)	Contrastive	Contrastive (No ECG)
Elevated mPAP (AUROC)	0.70 ± 0.01	0.67 ± 0.01	$\mathbf{0.72 \pm 0.01}$	0.66 ± 0.01
24 hour mortality (AUPRC)	0.07 ± 0.03	$\mathbf{0.12 \pm 0.01}$	$\mathbf{0.12 \pm 0.01}$	$\mathbf{0.12 \pm 0.02}$

Table 2: **Multimodal contrastive pre-training obtains improved or competitive performance when compared to baselines.** Using our multimodal pre-training approach improves performance over (1) not pre-training and (2) training with only the structured data on the elevated mPAP detection task. On mortality prediction, our method performs comparably with baselines that do not use ECGs, and improves over the RandInit baseline that uses the multimodal trajectory as input, suggesting the pre-trained model better utilizes the rich feature space.

- **RandInit**: We train a model from random initialization on each FT task on the full trajectory (statics, structured data timeseries, and signals timeseries).
- **RandInit (No ECG)**: We train a model from random initialization on only the tabular data (statics and structured data timeseries).
- **Contrastive**: we follow the pre-training strategy outlined in Section 2.2 on the PT dataset, before fine-tuning this model on the FT tasks.
- **Contrastive (No ECG)**: we pre-train on the tabular data alone, following the same augmentation process as described above, before fine-tuning the model on the tabular data alone for each FT task.

Experimental setup. We split our dataset on a per-patient level into an 80/20 development/test set split and use 20% of the development set as a validation set. We standardize the encoder f_θ to be a CNN-MLP-GRU. We use a 2-layer MLP for the projection head g_ϕ. We conduct pre-training for 10 epochs, using a batch size of 128. We set augmentation strength hyperparameters to values used in prior work [8, 13, 1], and set the temperature of the NT-Xent loss to 0.1, following [13]. Fine-tuning is for 10 epochs, using early stopping based on validation loss. All models are trained with the Adam optimizer with a learning rate of 1e-3. We run 5 random seeds at fine-tuning time, and report the median/half IQR of test AUROC for elevated mPAP and test AUPRC for 24 hour mortality (since the label has very low prevalence). Further experimental details are in the appendix.

Results. Results are shown in Table 2. We highlight two key findings:

- **Multimodal contrastive pre-training improves or obtains competitive performance.** The full contrastive strategy improves performance over all baselines on Elevated mPAP, and performs comparably to the best baselines on the 24 hour mortality task. This result suggests that the benefit of multimodal data in pre-training may be task-dependent. For the mortality task, with the amount of data available, the structured data alone are likely highly predictive, and so also including the ECGs does not improve performance. Improved performance may be observed at lower data regimes, as explored in prior work [9].
- **A more complex feature space can worsen performance without pre-training.** On the 24 hour mortality task, including the ECGs in a model without pre-training worsens performance noticeably, which is likely a result of overfitting with a more complex feature space. The pre-trained model does not show this behaviour, suggesting that the pre-training process helps mitigate this issue.

4 Conclusion

In this work, we outlined a contrastive pre-training strategy for multimodal clinical time series. Our method models trajectories of electrocardiogram signals and structured measurements from lab tests and vitals signs. In an experimental evaluation on a dataset of patients with heart failure, we find that our strategy can result in performance improvements on two downstream tasks.

Scope and Limitations. Given that this is an early-stage investigation, we only evaluate methods on two downstream tasks, and did not conduct extensive search over various model and training hyperparameters. Addressing these, and comparing to other pre-training schemes, is an important direction of future work.

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

References

[1] D. Bahri, H. Jiang, Y. Tay, and D. Metzler. Scarf: Self-supervised contrastive learning using random feature corruption. In *International Conference on Learning Representations*, 2021.

[2] A. Bardes, J. Ponce, and Y. LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *ICLR*, 2022.

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

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

[5] B. Gopal, R. W. Han, G. Raghupathi, A. Y. Ng, G. H. Tison, and P. Rajpurkar. 3kg: Contrastive learning of 12-lead electrocardiograms using physiologically-inspired augmentations. 2021.

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

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

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

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

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

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

[12] A. Weatherhead, R. Greer, M.-A. Moga, M. Mazwi, D. Eytan, A. Goldenberg, and S. Tonekaboni. Learning unsupervised representations for icu timeseries. In *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pages 152–168. PMLR, 07–08 Apr 2022.

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

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 4.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 4.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The dataset is private so we cannot include it here. We describe key aspects of our experimental setup in the main text and appendix.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We describe these details in the appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We describe these details in the appendix.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] We describe these details in the appendix.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [Yes] We have IRB approval for this study – details are provided in the appendix.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Further Methods Details

Loss function. Given a positive pair of projections z_i and z_j that originate from different augmentations of the same underlying sample, the NT-Xent loss function is computed as follows:

$$\mathcal{L}(z_i, z_j) = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2B} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}, \tag{1}$$

where $\text{sim}(a, b) = a^{\mathsf{T}} b/(\|a\|\|b\|)$ is cosine similarity and τ is the temperature hyperparameter. This is the exact form used in prior work [3].

B Further Experimental Details

Dataset. Our dataset is derived from the electronic medical record at a large tertiary hospital and was obtained with IRB approval. We will provide the IRB approval number if the paper is accepted. Since the dataset has some identifiable information, all computations are performed on a server that sits behind the hospital firewall.

Architectural details. We use the following architectures for the encoder and projection head:

- **Encoder** (f_θ): Each signal 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 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 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 head** (g_ϕ): This is a 2-layer fully connected network with batch normalization and ReLU activation with 128 hidden units. It takes the last hidden state of the GRU as input. The resulting projection is normalized (following Chen et al. [3]) before computing the NT-Xent loss over the batch.

Augmentations and hyperparameters. We use the following augmentations in the contrastive learning pipeline. Unless otherwise mentioned, the strength of the transformations were chosen based on the values reported in the original works.

- Random signal masking [8]: Randomly set 50% of the signal to zero.
- Random signal noise: Add Gaussian noise with variance 0.1 to the signal.
- History cutout [13]: With probability 0.8, randomly set 50% of the timesteps in each feature dimension to zero (i.e., impute with the mean). The masking fraction of 50% is different to the value of 17% used in the original paper, and was determined based on validation loss on the downstream tasks.
- Channel dropout [13]: For each feature dimension, with probability 0.2, set all the values over time to zero (i.e., impute with the mean).
- Feature corruption [1]: For each feature dimension, with probability 0.6, replace the value that feature takes with a random value drawn from the uniform distribution over the values that feature takes in the training data.

Other hyperparameters. We did not search over architectural choices, such as embedding dimensions for the different modalities, and instead opted to standardize them to be the same (128). The learning rate of 1e-3 was deduced based on analyzing learning curves and seeing that this value resulted in stable training across all models.

Compute details. All models were trained on a single NVIDIA Quadro RTX 8000 GPU. Pre-training takes about 10 hours, and fine-tuning on an individual task takes about 1 hour. Pre-training uses approximately 20 GB of GPU memory, and fine-tuning uses approximately 10 GB of GPU memory.