Self-supervised Representation Learning from Random Data Projectors

Yi Sui
Layer 6 AI
amy@layer6.ai

Tongzi Wu
Layer 6 AI
tongzi@layer6.ai

Jesse C. Cresswell
Layer 6 AI
jesse@layer6.ai

Ga Wu
Dalhousie University
ga.wu@dal.ca

George Stein
Layer 6 AI
george@layer6.ai

Xiaoshi Huang
Layer 6 AI
gary@layer6.ai

Xiaochen Zhang
Layer 6 AI
lisa@layer6.ai

Maksims Volkovs
Layer 6 AI
maks@layer6.ai

Abstract

Self-supervised representation learning (SSRL) has advanced considerably by exploiting the transformation invariance assumption under artificially designed data augmentations. While augmentation-based SSRL algorithms push the boundaries of performance in computer vision and natural language processing, they are often not directly applicable to other data modalities such as tabular and time-series data. This paper presents an SSRL approach that can be applied to these data modalities because it does not rely on augmentations or masking. Specifically, we show that high-quality data representations can be learned by reconstructing random data projections. We evaluate the proposed approach on real-world applications with tabular and time-series data. We show that it outperforms multiple state-of-the-art SSRL baselines and is competitive with methods built on domain-specific knowledge. Due to its wide applicability and strong empirical results, we argue that learning from randomness is a fruitful research direction worthy of attention and further study.

1 Introduction

In recent times, self-supervised representation learning (SSRL) has witnessed remarkable success, particularly in the fields of computer vision [19] and natural language processing [18]. Even with an abundance of raw data structured in tabular and time-series formats throughout the domains of healthcare, finance, the natural sciences, and more, extending the success of SSRL to these data modalities remains challenging [6].

The success of SSRL in computer vision and natural language processing primarily stems from well-designed pretext tasks that create heuristics from unlabelled data allowing models to identify and encode useful information. Pretext tasks are often highly customized to specific applications based on a handful of underlying assumptions. Specifically, pretext tasks utilizing the transformation invariance assumption across data augmentation views show leading performance in multiple research domains. For computer vision, image representations are commonly guided to remain identical after cropping, rotating, flipping, or corrupting, among others [11, 19]. Similarly, in natural language processing, sentences with similar words and semantic meaning are expected to have the same representation [45, 46]. In these and other domains transformation invariance is encouraged explicitly through contrastive or momentum objectives that aim to bring together representations before and after transformation.

Table Representation Learning Workshop at NeurIPS 2023.

Despite their strong performance in many domains, SSRL algorithms that enforce transformation invariance are limited in that they do not support generic data types. Many well-known data augmentation methods are tailored to specific modalities, and are restricted in their generality across different domains. For instance, image rotation is not applicable in the tabular domain.

Beyond the issue of cross-modal applicability, a more subtle challenge emerges where even standard augmentations clash with application-specific constraints. In tabular settings where relatively few options for augmentations are available, random noise addition or random swapping of features between training examples can easily produce unrealistic examples. As a concrete example from particle physics, consider energies and momenta of interacting particles collected experimentally [8]. Since the incoming and outgoing energies are constrained by the laws of physics, only certain combinations are possible to observe. Tabular augmentations would produce unphysical combinations, harming the consistency of learned representations and leading to unpredictable failures on downstream tasks [35]. For time series with high periodicity, such as online monitoring data, random shifting augmentations can create identical augmented views that are not useful for pretext tasks [17, 51]

Instead of relying on transformation invariance, many self-supervised learning techniques involve masking and reconstructing input data including masked image modelling [23, 47] for computer vision and masked language modelling [16, 32] for natural language processing. However, these methods often require specific backbone architectures like transformers [41] to achieve good performance [6].

In this paper, we introduce a SSRL training scheme that neither requires domain-specific data augmentations nor particular architectures as in masking approaches. Instead, the proposed learning method is based on a surprising hypothesis that good data representations can be obtained by learning to simultaneously reconstruct multiple randomly generated data projection functions. The hypothesis comes from the conventional motivation of representation learning – capturing and extracting abstract and valuable concepts that can support a range of downstream predictive tasks [7, 27]. In particular, the downstream tasks could include arbitrary data projections. Formally, given random projection functions $G = \{\cdots g^{(k)}(\mathbf{x}) \cdots\}$ whose input domains are raw data features $\mathbf{x} \in \mathcal{X}$, the motivation above suggests that, for a good representation $\mathbf{z}$ of data $\mathbf{x}$, there is another group of simple prediction functions $H = \{\cdots h^{(k)}(\mathbf{z}) \cdots\}$ that can correctly predict the random function's outputs. With this insight, the representation learning task can be construed as a search for a combination of representation $\mathbf{z}$ and prediction functions H that can reproduce random data projections. In short, we conduct SSRL by *learning from randomness* (LFR).

The primary advantage of LFR is that random projection functions G can easily be created for *any* data modality. One straightforward instantiation is by taking a neural network with suitable architecture for consuming the data, and randomly initializing its parameters. Hence, LFR applies to all subfields of SSRL. Also, data augmentations are not used, so LFR avoids any concerns of identical, unrealistic, or unsafe augmentations as discussed above.

We empirically evaluate the effectiveness of LFR on a wide range of representation learning tasks within the tabular and time-series modalities where no standard pretext pipelines exist. The results show that LFR outperforms both commonly used domain-agnostic SSRL algorithms as well as domain-specific approaches that rely heavily on expert knowledge for their data augmentation designs. The remarkable performance demonstrates that learning high-quality data representations from randomness is a feasible and plausible alternative when the transformation invariance assumption is hard to establish or enforce in a given application domain.

2 Background and Related Work

Self-supervised representation learning (SSRL) methods enable the extraction of informative and compact representations from raw data without manual annotation or labelling. These methods rely on large amounts of unlabeled data and pretext tasks to implicitly model the observed distribution and optimize deep neural networks. Contrastive learning methods use heavy augmentation to generate positive views – semantically similar examples which are optimized to have the same representation as the original datapoint [15, 20, 40, 11]. To ensure the quality of learned representations, augmentations should be semantic-preserving [37, 6]. However, finding suitable augmentations for different application domains can be a challenging task, and researchers have invested considerable effort into this area to enhance downstream performance. Augmentation strategies that work well for one

modality may not directly translate to others due to inherent differences, and the choice of suitable augmentations can also be influenced by the specific application domain.

While masking approaches offer general applicability to all data modalities, the most effective frameworks often rely on transformer-based backbones for optimal performance [30, 23, 14]. In this work, we focus on a model-agnostic SSRL approach. Classic autoencoder-based methods provide an alternative to SSRL without relying explicitly on transformation invariance [24, 43, 52]. However, these methods tend to prioritize low-level reconstruction over capturing high-level abstractions required for downstream tasks, resulting in suboptimal performance in practical applications [28].

Tabular SSRL methods are understudied in the tabular domain as designing effective semantic-preserving augmentations is particularly challenging for structured data [49]. Unlike computer vision tasks on photographic images, with tabular data small changes to individual features can drastically change the content, and it is often difficult for a human to determine if two views should be considered semantically equivalent. To generate positive views SubTab [38] uses different feature subsets. More recently, SCARF [3] proposed to augment each record by corrupting a random subset of features. Finally, STab [21] creates the contrastive views by imposing different regularization on the encoder for the same input.

Time series Time series data often contains underlying patterns that are not easily identifiable by humans, unlike images with recognizable features [29]. Consequently, designing effective data augmentation methods for time series data poses significant challenges and often requires domain knowledge. For example, augmentations for wearable sensor signals include rotation to simulate different sensor placements and jittering to simulate sensor noise [39]. Other researchers have focused on bio-signals and introduced channel augmentations that preserve the semantic information in the context [31, 13]. Neighbourhood contrastive learning [48] proposed leveraging patient identity information in online patient monitoring and using near-in-time data sequences of the same patient as semantically equivalent pairs. However, these augmentations are often specifically designed for the dataset and downstream task [51], and their performance may deteriorate when applied to other time series data [17]. Therefore, identifying the optimal augmentation pipeline for each dataset and task requires extensive analysis [25].

The current landscape of SSRL research highlights the need for a more versatile and effective approach capable of addressing a wider range of modalities, applications, and architectures, especially the understudied tabular and time-series modalities.

3 Representation Learning from Random Data Projectors

In this section, we present *learning from randomness* (LFR), an efficient and general SSRL algorithm. We recap the representation learning problem setting as the following: given observed raw data $X = \{\cdots \mathbf{x}_i \cdots\}$, where all data points share the same feature domain $\mathcal{X}$, the representation learning task is to learn a function $f_\theta(\mathcal{X})$ that produces a low-dimensional representation $\mathbf{z}_i \in \mathcal{Z}$ for each raw data input $\mathbf{x}_i$. The representation $\mathbf{z}_i$ should carry useful information about $\mathbf{x}_i$ such that for an arbitrary downstream task $g(\mathcal{X})$ it is possible to learn a simple prediction function $h_\phi(\mathcal{Z})$ that replicates $g(\mathbf{x}_i)$ as $h_\phi\left(f_\theta(\mathbf{x}_i)\right)$ for all $\mathbf{x}_i \in \mathcal{X}$.

3.1 Pretext Task: Multi-objective Learning from Randomness

As mentioned in the problem statement above, the ultimate purpose of representation learning is to support arbitrary downstream predictive tasks. In reality, there is usually a small subset of downstream tasks which are considered important. It is not *a priori* clear that directly learning to predict purely random tasks would lead to good representations for important tasks.

To demonstrate the possibility of learning from randomness, we propose the surprising pretext task shown in Figure 1. The pretext task contains three components, namely a representation model $f_\theta(\mathcal{X})$, a set of randomly generated data projection functions $G = \{\cdots g^{(k)}(\mathcal{X}) \cdots\}$, and a set of simple predictors $H_\Phi = \{\cdots h_\phi^{(k)}(\mathcal{Z}) \cdots\}$ that aim to predict the outcome of each random projection

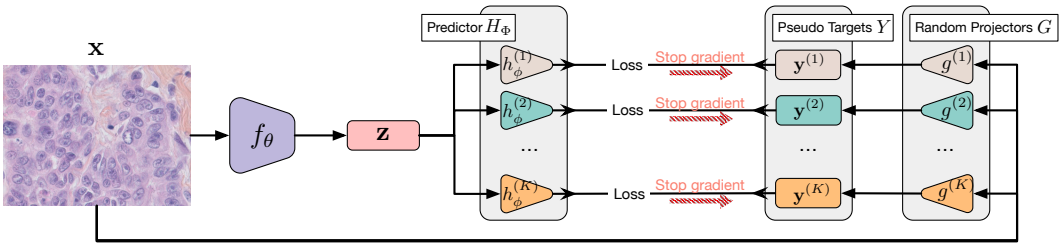

Figure 1: Our proposed architecture for learning from randomness. An input $\mathbf{x}$ is encoded by f_θ into a useful representation $\mathbf{z}$, while also being fed to random projection functions $g^{(k)}$. Simple, learnable predictor functions $h_\phi^{(k)}$ try to match the outputs $\mathbf{y}^{(k)}$ from the projectors $g^{(k)}$, which is only possible when $\mathbf{z}$ contains rich information about the input.

function respectively. Formally, we propose optimizing the high-level objective:

$$\underset{\theta,\Phi}{\operatorname{argmin}} \sum_{\mathbf{x}_i \in \mathcal{X}} \sum_k \mathcal{D}\left[g^{(k)}(\mathbf{x}_i), h_\phi^{(k)}\left(f_\theta(\mathbf{x}_i)\right)\right] + \lambda_1 \Omega_1(\theta) + \lambda_2 \Omega_2(\Phi), \tag{1}$$

where $\mathcal{D}[\cdot,\cdot]$ is a divergence metric measuring the similarity between its inputs, the $\Omega(\cdot)$'s denote regularization terms, and the λ's are the corresponding weights. To make this a non-trivial task, the predictors $h_\phi^{(k)}$ should have limited capacity, such as being linear functions, or simple neural networks with a few layers. This objective aligns with existing SSRL methods that use predictors [19, 12].

Objective 1 is essentially a lower-bound of the maximum likelihood estimation (MLE) objective; we aim to maximize the probability of observing data projections $\mathbf{y}_i^{(k)} = g^{(k)}(\mathbf{x}_i)$ given all datapoints,

$$\begin{aligned}\sum_i \log p\left(Y_i \mid \mathbf{x}_i\right) &= \sum_i \sum_k \log \int_{\mathbf{z}_i} p\left(\mathbf{y}_i^{(k)} \;\middle|\; \mathbf{z}_i\right) p\left(\mathbf{z}_i \mid \mathbf{x}_i\right) d\mathbf{z}_i \\ &\geq \sum_i \sum_k \int_{\mathbf{z}_i} p\left(\mathbf{z}_i \mid \mathbf{x}_i\right) \log p\left(\mathbf{y}_i^{(k)} \;\middle|\; \mathbf{z}_i\right) d\mathbf{z}_i = \sum_i \sum_k \log p\left(\mathbf{y}_i^{(k)} \;\middle|\; \mathbf{z}_i = f_\theta(\mathbf{x}_i)\right),\end{aligned} \tag{2}$$

where $p\left(\mathbf{z}_i \mid \mathbf{x}_i\right)$ is a Dirac delta distribution since the representation model is a deterministic function. As an example of the connection between Objectives (1) and (2), when the $p\big(\mathbf{y}_i^{(k)} \mid \mathbf{z}_i\big)$ are assumed to be Gaussian distributions, the corresponding $\mathcal{D}$ in Objective 1 is the Euclidean distance. We discuss other options for $\mathcal{D}$ below.

A straightforward training strategy for Objective 1 is joint training where we treat the representation model f_θ and predictors H_Φ as a multi-objective autoencoder, updating all parameters in the same backpropagation pass. However, in preliminary experiments this naïve training strategy showed fluctuating progress which prevented the model from converging to satisfactory solutions.

We tackle the training instability issue by adopting the classic Expectation-Maximization (EM) method. Considering the MLE lower-bound (Objective 2), optimizing the representation model f_θ is an E-step that repositions the posterior distribution of data $p(\mathcal{Z}|\mathcal{X})$ given fixed log-likelihood estimation modules $\log p\big(\mathbf{y}_i^{(k)} \mid \mathbf{z}_i\big)$. For the M-step, the representation distribution is fixed and we optimize the predictor heads H_Φ. Hence, in this work we train the proposed SSRL model by alternating steps:

E-step: Optimize the representation model parameters θ for one iteration

$$\underset{\theta}{\operatorname{argmin}} \sum_i \sum_k \mathcal{D}\left[g^{(k)}(\mathbf{x}_i), h_\phi^{(k)}(f_\theta(\mathbf{x}_i))\right] + \lambda_1 \Omega_1(\theta), \tag{3}$$

M-step: Optimize the predictor model parameters Φ for M iterations

$$\underset{\Phi}{\operatorname{argmin}} \sum_i \sum_k \mathcal{D}\left[g^{(k)}(\mathbf{x}_i), h_\phi^{(k)}(f_\theta(\mathbf{x}_i))\right] + \lambda_2 \Omega_2(\Phi). \tag{4}$$

Optimizing the predictor model for more iterations than the representation model brings the predictor closer to its optimal performance with the latest representation model f_θ. Previous studies have shown that optimizing the predictor to achieve optimality leads to improved performance [12, 19, 36].

3.2 Divergence Measure: Batch-wise Barlow Twins

Now we return to options for the divergence $\mathcal{D}$ in Objective 1. While there are several common choices of divergence used in machine learning such as Mean Squared Error (MSE), Cross Entropy (CE), or the Contrastive [15] and Triplet [34] losses, they are often inadequate for enforcing identifications between subtly different points as is crucial for representation learning tasks; MSE downweights the importance of small errors, CE is ill suited for regression tasks, while the Contrastive and Triplet losses introduce significant stochasticity.

The Barlow Twins loss [50] has garnered much interest in the SSRL literature as it disentangles learned representations through redundancy reduction. We also note its ability to scale to very high-dimensional vectors [50]. Thus, we introduce Batch-wise Barlow Twins (BBT), a variant that measures representation differences between data instances from two sources, the random projector $g^{(k)}$ and the predictor $h_\phi^{(k)}$, rather than disentangling the representation encoding. We define the BBT loss as

$$L_{\text{BBT}} = \sum_k \sum_i \left[\left(1 - c_{ii}^{(k)}\right)^2 + \lambda \sum_{j \neq i} {c_{ij}^{(k)}}^2 \right], \tag{5}$$

where the c_{ij} are the entries of a cosine similarity matrix,

$$c_{ij}^{(k)} = \frac{{\mathbf{y}_i^{(k)}}^\top \hat{\mathbf{y}}_j^{(k)}}{\left\|\mathbf{y}_i^{(k)}\right\|_2 \left\|\hat{\mathbf{y}}_j^{(k)}\right\|_2}, \quad \mathbf{y}_i^{(k)} = g^{(k)}(\mathbf{x}_i), \quad \hat{\mathbf{y}}_i^{(k)} = h_\phi^{(k)}(f_\theta(\mathbf{x}_i)), \tag{6}$$

and $\mathbf{y}_i^{(k)}, \hat{\mathbf{y}}_i^{(k)} \in \mathbb{R}^{d^{(k)}}$. Compared to the loss in [50], Eq. 5 has an extra summation over the ensemble k. The main difference comes from the definition of the cosine similarity matrix; our cosine similarity is an $m \times m$ matrix with m the batch size, whereas in Barlow Twins it is a $d^{(k)} \times d^{(k)}$ matrix.

3.3 Diversity Encouragement on Random Data Projectors

Learning from randomness aims to extract useful representations from random projection functions $g^{(k)}(\mathcal{X}) \in G$ which mimic arbitrary downstream tasks. In practice we create multiple data projections by randomly initializing neural networks of various architectures. Functions generated this way can often capture similar information to each other when diversity is not specifically encouraged, which limits the generalization capabilities of the representations learned by f_θ. While increasing the number of random projection functions could mitigate the diversity problem by brute force, such an approach is computationally wasteful because it would maintain many similar projectors.

We propose a solution that picks K diverse projectors from $N \gg K$ randomly generated candidates. The underlying hypothesis is that one sufficiently large batch of data can reveal the behavioral differences between candidate random projectors. Presuming there is a batch of data $X \in \mathbb{R}^{m \times d}$, for each of the N randomly generated projectors $g^{(k)}(\mathcal{X}) \in G$ we produce the normalized outputs

$$Y^{(k)} = g^{(k)}(X) / \|g^{(k)}(X)\|_2, \quad Y^{(k)} \in \mathbb{R}^{m \times d^{(k)}}. \tag{7}$$

We then compute the cosine similarity over the batch of outputs for each projector as

$$A^{(k)} = Y^{(k)} (Y^{(k)})^\top, \quad A^{(k)} \in \mathbb{R}^{m \times m}. \tag{8}$$

By flattening the matrix $A^{(k)}$ and again normalizing, we obtain a vector $\mathbf{a}^{(k)} \in \mathbb{R}^{m^2 \times 1}$, which acts as the signature of the k'th projector with respect to the batch. Finally, to select K target models from the N candidates, we search for a subset that maximizes the following binary constraint optimization problem involving matrices $\tilde{A}$ made from K stacked vectors $\mathbf{a}^{(k)}$,

$$\underset{\mathbf{s}}{\text{argmax}} \, |\det(B)| \quad \text{s.t.} \quad B = \tilde{A}\tilde{A}^\top, \quad \tilde{A} = \left[\mathbf{a}^{(k)} \;\middle|\; k \in [0, N], s_k = 1, \sum_{k'} s_{k'} = K \right], \tag{9}$$

where the 1's in the binary vector $\mathbf{s} \in \{0, 1\}^N$ indicate the chosen projectors. While this problem is known to be NP-hard, approximate methods such as the Fast Determinantal Point Process [10] can find good solutions in reasonable time. It is worth noting that our diversity encouragement solution does not involve gradient computations, and can be run once as a pre-processing step without occupying computation resources during the SSRL training phase.

We summarize the full LFR algorithm in Appendix A.

4 Experiments and Evaluation

4.1 Datasets

We consider both time series and tabular data types in various domains to show the wide applicability of *learning from randomness*. Datasets are further detailed in Appendix B.1.

Time series We utilized two standard time-series datasets, Human Activity Recognition (HAR) [2] and Epileptic Seizure Recognition [1]. Both datasets were pre-processed using the same methods as in TS-TCC [17]. As a larger scale test we also include the MIMIC-III dataset, a standard in the medical domain for tasks involving electronic health record data. We utilized the pre-processed version of the MIMIC-III Benchmark dataset [22], and focused on the length-of-stay task [48] which is framed as a 10-class classification problem, where each class represents a different duration of stay.

Tabular We used three tabular UCI datasets in our experiments: Adult Income (Income) [26], First Order Theorem Proving (Theorem) [9], and HEPMASS [4]. For Income, a binary classification problem, we followed the data preprocessing steps in [38]. The Theorem dataset is framed a a 6-class classification problem. The much larger HEPMASS dataset is another binary classification task which includes 7 million training and 3.5 million testing events, each with 27 features.

4.2 Implementations

Evaluation All the downstream tasks in our study are treated as classification problems. To evaluate the quality of the pre-trained representations, we employed supervised classifiers that are specific to each dataset. For the MIMIC-III dataset we utilized a MLP classifier [48]. For tabular datasets, we used logistic regression, similar to the approach in STab [21]. For the remaining datasets, a linear classifier was employed. We experimented with both downstream evaluation where the classifiers were trained on frozen representations, and finetuning where the classifier and representation model are jointly trained. For finetuning on the larger datasets MIMIC-III and HEPMASS, we chose a semi-supervised approach where we randomly selected 10% of labeled data from these datasets, and we used a subset of the baseline methods. These decisions were driven by the computational resources required when dealing with extensive data and finetuning large representation models. Conversely, for smaller datasets, we conducted finetuning using the complete set of available labeled data, enabling us to evaluate the model's performance across the entirety of the datasets.

Metrics were then computed on the test set. Accuracy is our primary metric, except for MIMIC-III where we adopted linearly weighted Cohen's Kappa as in [48], with higher values indicating better agreement. To ensure the robustness of our results, we conducted multiple random runs and report the mean and standard deviation, using 5 runs for tabular datasets and 3 runs for time-series.

Model architectures Regarding the model architectures, we adopted similar backbone encoders as previous works. For the HAR and Epilepsy datasets, we utilized the same 3-block convolutional layers as TS-TCC [44]. For the MIMIC-III dataset, we employed the Temporal Convolutional Network used by NCL [48]. For the Tabular datasets, we used 4-layer MLPs, following the approach in SCARF [3]. To avoid domain-specific projector design, in each case the random projectors reuse the architecture from the encoder, but are scaled down. Complete details are in Appendix B.2.

Table 1: Baseline methods

Category	Method	Description
Domain-agnostic	Autoencoder [33]	Encoder/decoder with low dimensional latents trained via the reconstruction loss.
	DIET [5]	Self-supervised learning method that predicts the datum index as a pretext task.
	DACL [42]	Self-supervised learning method that uses mix-up as data augmentation across modalities.
Domain-specific augmentations	SimCLR [11]	Contrastive learning method with both positive and negative pairs.
	SimSiam [12]	Self-supervised learning with Siamese networks and only positive pairs.
Time series	TS-TCC [17]	Contrastive learning method that uses a correlation-based similarity to capture temporal relationships, and time-series augmentations to generate positive and negative views.
Tabular	SCARF [3]	Adaptation of SimCLR to tabular domains, using random corruption for dual views.
	STab [21]	An augmentation-free framework for tabular self-supervised learning akin to SimSiam. Positive pairs are created by different regularizations in the forward pass.
Supervised	LogReg	Supervised training with logistic regression.
	Supervised	Supervised training with a classification layer added to the encoder used in other methods.
Ablation	Random Init	As an ablation baseline we report the accuracy using a randomly initialized encoder [17].

Baseline methods Table 1 summarizes all baselines used in our experiments. It is worth noting that while our proposed framework LFR is domain-agnostic, popular SSRL methods such as SimCLR and SimSiam require domain-specific augmentations to achieve optimal performance. Specifically, the default augmentations used for view creation in SimCLR and SimSiam are designed for natural image classification, and may not be suitable for other modalities. In our experiments with tabular datasets, we compare our approach to SCARF [3], which is a version of SimCLR adapted to tabular data that uses random corruptions as augmentations, as well as STab [21] which is similar to SimSiam. For more detailed information on the implementations and augmentations, please refer to Appendix B.3, and B.4. Information on the computing resources used is in Appendix B.5.

4.3 Performance Comparison

The performance of LFR and baselines across multiple modalities and domains using downstream evaluation is shown in Table 2. Our experiments show that for time series and tabular data where there are no standardized augmentation pipelines, LFR had the strongest performance among the SSRL methods, outperforming other self-supervised learning methods in most cases including the domain-agnostic ones such as DACL. For instance, on the HAR and Epilepsy datasets, LFR was the best performing method, beating the time-series specific self-supervised learning method TS-TCC. Similarly, for the Income and Theorem datasets, LFR outperformed the tabular data specific self-supervised learning baselines SCARF and STab. Although on the HEPMASS dataset LFR was not the best, it still performed well, comparable to the autoencoder and SCARF. Interestingly, for the Income dataset, LFR even outperformed supervised training. For time series and tabular data, augmentation-based methods like SimSiam tend to underperform. For example, SimSiam was worse than a randomly initialized encoder in HAR and Income.

Table 2: Performance comparison across various application domains with downstream evaluation. Results of the best self-supervised learning methods are in bold.

		Time series			Tabular		
		HAR	Epilepsy	MIMIC-III	Income	Theorem	HEPMASS
	Log Reg	57.5±N/A	80.9±N/A	47.8±N/A	84.8±N/A	45.3±N/A	90.7±N/A
	Supervised	96.0±0.6	98.3±0.1	48.8±0.0	81.5±0.2	53.8±0.5	91.5±0.0
self-supervised	Random Init	80.7±2.3	89.1±0.1	42.4±1.1	83.1±0.2	44.9±0.8	84.3±1.3
	Autoencoder	77.2±0.7	90.8±1.3	44.9±0.5	85.0±0.1	50.0±0.4	**90.7±0.0**
	DIET	88.6±1.3	96.8 ± 0.3	33.8±5.2	82.2 ±0.4	47.1 ±0.5	-
	SimSiam	65.1±0.8	97.4±0.0	41.0±1.9	79.2±1.9	40.9±0.9	85.3±3.1
	SimCLR	87.8±0.4	97.4±0.2	44.1±0.1	-	-	-
	SCARF	-	-	-	84.2±0.1	48.5 ±1.0	90.1±0.1
	STab	-	-	-	84.2±0.3	50.7±0.7	83.6±1.7
	TS-TCC	91.2±0.8	97.6±0.2	38.5±1.3	-	-	-
	DACL	90.7±0.4	97.5±1.5	40.9±0.6	79.8±0.7	47.6±1.0	88.7±0.8
	LFR (Ours)	**93.1±0.5**	**97.9±0.2**	**46.6±0.3**	**85.2±0.1**	**51.6±0.7**	90.1±0.2

Table 3: Performance comparison across various application domains with finetuning. Results of the best self-supervised learning methods are in bold.

		Time series			Tabular		
		HAR	Epilepsy	MIMIC-III	Income	Theorem	HEPMASS
	Supervised	96.0 ± 0.6	98.3 ± 0.1	48.8 ± 0.0	81.5 ± 0.2	53.8 ± 0.5	91.5 ± 0.0
self-supervised	Autoencoder	93.9 ± 1.3	95.1 ± 2.0	49.2 ± 0.6	85.2 ± 0.1	53.9 ± 0.5	90.8 ± 0.0
	DIET	**95.6 ± 0.5**	97.8 ± 0.1	48.4 ± 0.1	85.2 ± 0.1	52.4 ± 0.9	-
	SimSiam	93.4 ± 0.6	97.9 ± 0.2	49.4 ± 0.3	85.2 ± 0.1	52.5 ± 0.8	90.7 ± 0.0
	SimCLR	93.7 ± 1.1	97.8 ± 0.2	48.6 ± 0.8	-	-	-
	SCARF	-	-	-	85.1 ± 0.2	53.8 ± 0.8	90.9 ± 0.0
	STab	-	-	-	**85.3 ± 0.2**	53.0 ± 0.7	**91.1** ± 0.0
	LFR	94.7 ± 1.4	**98.2 ± 0.2**	**49.6 ± 0.1**	**85.3 ± 0.1**	**54.3 ± 0.4**	90.8 ± 0.0

Finetuning results are shown in Table 3. Through the finetuning process, all methods exhibit more comparable performance across the datasets. LFR still achieved the best performance on a majority of the datasets we used, although with overlapping error bars to other methods in those cases.

Overall, these experimental results reflect our hypothesis – it is feasible to learn high-quality data representations across all domains tested by predicting random data projections. LFR shows comparatively good performance on domains where semantic-preserving augmentations are difficult to create.

4.4 Ablation Study

Embedding dimensions The dimensionality of embeddings may have a strong effect on the richness of learned representations. On the Theorem dataset we evaluated the performance of LFR and baseline SSRL approaches across latent dimension sizes. Figure 2 shows that increasing the latent dimension improved the accuracy of each approach up to about 256. LFR consistently outperformed all the other baselines across all the latent dimension settings.

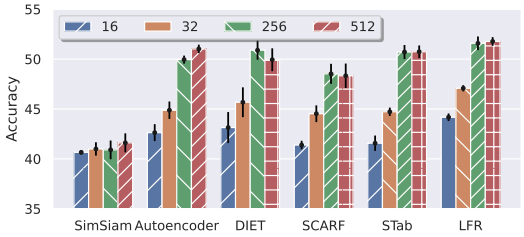

Figure 2: Effect of embedding dimension on LFR and self-supervised learning baselines.

5 Conclusion and Future Work

This paper presents a novel self-supervised representation learning framework that is modality-agnostic, application-agnostic, and architecture-agnostic. Our proposed framework utilizes random projectors to learn representations from unlabelled data, demonstrating excellent performance across various modalities and applications, particularly in situations where robust augmentation pipelines are not yet established.

The LFR technique is best suited to situations where the data cannot be reasonably augmented (due to the lack of domain knowledge). Although surprising, this situation occurs frequently in critical application domains, such as healthcare as we have highlighted with the MIMIC-III dataset. For general applications, however, if one knows the application domain well with adequate intuition around sensible data augmentations, using contrastive-learning-based SSRL is still likely to outperform random projectors. Overall, we treat LFR as a great complement to SSRL literature to fill the gap of data augmentation-free SSRL that satisfies the needs of many crucial applications.

While LFR encourages the use of SSRL across modalities and domains without the need for expert knowledge to craft augmentations, the limited human input to the learning process may increase the risk of sensitive features being misused, leading to privacy or fairness concerns. We recommend human oversight of self-supervised methods to monitor for appropriate use of data.

References

[1] R. G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger. Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. *Physical Review E*, 64(6):061907, 2001.

[2] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *The European Symposium on Artificial Neural Networks*, 2013.

[3] D. Bahri, H. Jiang, Y. Tay, and D. Metzler. Scarf: Self-supervised contrastive learning using random feature corruption. In *International Conference on Learning Representations*, 2022.

[4] P. Baldi, K. Cranmer, T. Faucett, P. Sadowski, and D. Whiteson. Parameterized neural networks for high-energy physics. *The European Physical Journal C*, 76(5):235, Apr 2016.

[5] R. Balestriero. Unsupervised Learning on a DIET: Datum IndEx as Target Free of Self-Supervision, Reconstruction, Projector Head. *arXiv:2302.10260*, 2023.

[6] R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian, et al. A Cookbook of Self-Supervised Learning. *arXiv:2304.12210*, 2023.

[7] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[8] J. Brehmer and K. Cranmer. Flows for simultaneous manifold learning and density estimation. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[9] J. P. Bridge, S. Holden, and L. C. Paulson. Machine Learning for First-Order Theorem Proving - Learning to Select a Good Heuristic. *J. Autom. Reason.*, 53:141–172, 2014.

[10] L. Chen, G. Zhang, and E. Zhou. Fast greedy map inference for determinantal point process to improve recommendation diversity. *Advances in Neural Information Processing Systems*, 31, 2018.

[11] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

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

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

[14] M. Cheng, Q. Liu, Z. Liu, H. Zhang, R. Zhang, and E. Chen. Timemae: Self-supervised representations of time series with decoupled masked autoencoders. *arXiv preprint arXiv:2303.00320*, 2023.

[15] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 539–546, 2005.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[17] E. Eldele, M. Ragab, Z. Chen, M. Wu, C. K. Kwoh, X. Li, and C. Guan. Time-series representation learning via temporal and contextual contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 2352–2359, 2021.

[18] T. Gao, X. Yao, and D. Chen. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, 2021.

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

[20] M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9, pages 297–304, 2010.

[21] E. Hajiramezanali, N. L. Diamant, G. Scalia, and M. W. Shen. STab: Self-supervised Learning for Tabular Data. In *NeurIPS 2022 First Table Representation Workshop*, 2022.

[22] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific data*, 6(1):96, 2019.

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

[24] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[25] B. K. Iwana and S. Uchida. An empirical survey of data augmentation for time series classification with neural networks. *PLOS ONE*, 16(7):1–32, 2021.

[26] R. Kohavi et al. Scaling up the accuracy of Naive-Bayes classifiers: A decision-tree hybrid. In *KDD*, volume 96, pages 202–207, 1996.

[27] P. H. Le-Khac, G. Healy, and A. F. Smeaton. Contrastive representation learning: A framework and review. *IEEE Access*, 8:193907–193934, 2020.

[28] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):857–876, 2021.

[29] D. Luo, W. Cheng, Y. Wang, D. Xu, J. Ni, W. Yu, X. Zhang, Y. Liu, Y. Chen, H. Chen, et al. Time series contrastive learning with information-aware augmentations. *arXiv:2303.11911*, 2023.

[30] K. Majmundar, S. Goyal, P. Netrapalli, and P. Jain. Met: Masked encoding for tabular data, 2022.

[31] M. N. Mohsenvand, M. R. Izadi, and P. Maes. Contrastive representation learning for electroencephalogram classification. In *Machine Learning for Health*, pages 238–253. PMLR, 2020.

[32] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning Internal Representations by Error Propagation*, page 318–362. MIT Press, Cambridge, MA, USA, 1986.

[34] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[35] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.

[36] Y. Tian, X. Chen, and S. Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *International Conference on Machine Learning*, pages 10268–10278. PMLR, 2021.

[37] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola. What makes for good views for contrastive learning? *Advances in neural information processing systems*, 33:6827–6839, 2020.

[38] T. Ucar, E. Hajiramezanali, and L. Edwards. SubTab: Subsetting features of tabular data for self-supervised representation learning. *Advances in Neural Information Processing Systems*, 34:18853–18865, 2021.

[39] T. T. Um, F. M. J. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulić. Data Augmentation of Wearable Sensor Data for Parkinson's Disease Monitoring Using Convolutional Neural Networks. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, page 216–220, 2017.

[40] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv:1807.03748*, 2018.

[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[42] V. Verma, T. Luong, K. Kawaguchi, H. Pham, and Q. Le. Towards domain-agnostic contrastive learning. In *International Conference on Machine Learning*, pages 10530–10541. PMLR, 2021.

[43] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[44] Z. Wang, W. Yan, and T. Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[45] J. Wei and K. Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

[46] X. Wu, S. Lv, L. Zang, J. Han, and S. Hu. Conditional bert contextual augmentation. In *Computational Science–ICCS 2019: 19th International Conference, Faro, Portugal, June 12–14, 2019, Proceedings, Part IV 19*, pages 84–95. Springer, 2019.

[47] Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu. Simmim: A simple framework for masked image modeling, 2022.

[48] H. Yèche, G. Dresdner, F. Locatello, M. Hüser, and G. Rätsch. Neighborhood contrastive learning applied to online patient monitoring. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 11964–11974, 2021.

[49] J. Yoon, Y. Zhang, J. Jordon, and M. van der Schaar. Vime: Extending the success of self- and semi-supervised learning to tabular domain. *Advances in Neural Information Processing Systems*, 33:11033–11043, 2020.

[50] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

[51] J. Zhang and K. Ma. Rethinking the augmentation module in contrastive learning: Learning hierarchical augmentation invariance with expanded views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16650–16659, 2022.

[52] R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1058–1067, 2017.

A Algorithm

We summarize the LFR algorithm in Algorithm 1 which uses the subroutine in Algorithm 2.

Algorithm 1 LFR: Learning From Randomness

Require: Dataset $\mathcal{D} = (x_i)_{i=1}^n$, number of random projectors K
Ensure: Encoder f_θ
Initialize encoder f_θ
Initialize $10 \cdot K$ different random projectors and select K diverse projectors $g^1, ..., g^K$ using DPP as introduced in Section 3.3
Initialize K predictors $h_\phi^1, ..., h_\phi^K$, one for each projector.
for epoch$_{\text{all}}$ in training epochs **do**
    for each mini-batch B **do** ▷ Train encoder
        Train-Network(B, f_θ, h_ϕ^k, g^k)
        Update parameters of f_θ with gradient descent using L_{BBT} as introduced in Section 3.2
    end for
    for epoch$_p$ in predictor epochs **do**
        for each mini-batch B **do** ▷ Train predictor
            Train-Network(B, f_θ, h_ϕ^k, g^k)
            Update parameters of all h_ϕ^k with gradient descent using L_{BBT}
        end for
    end for
end for

Algorithm 2 Train-Network subroutine

procedure TRAIN-NETWORK(B, f_θ, h_ϕ^k, g^k)
    Compute representations: $Z = f_\theta(B)$
    for $k = 1$ to K **do**
        Compute output representations: $h_\phi^k(Z)$
        Compute representation from projector k: $g^k(Z)$
        Compute loss L_{BBT}
    end for
end procedure

B Implementation Details

B.1 Dataset Summary

Table 4 provides a summary of all datasets used in our experiments, along with the corresponding downstream tasks and evaluation metrics.

Table 4: Dataset description.

Dataset	Modality	Data Domain	Train Size	Test Size	Downstream Task	Metric
HAR	Time series	Mobile sensors	7352	2947	Multi-class classification (6)	Accuracy
Epilepsy	Time series	Brain EEG	9200	2300	Binary classification	Accuracy
MIMIC-III	Time series	Online Monitoring	2,568,619	563,742	Multi-class classification(10)	Cohen's Kappa
Income	Tabular	Census	30162	15060	Binary classification	Accuracy
Theorem	Tabular	Logic Reasoning	3059	1530	Multi-class classification (6)	Accuracy
HEPMASS	Tabular	Particle Physics	7,000,000	3,500,000	Multi-class classification (2)	Accuracy

Table 5: Details on LFR architectural parameters

Dataset	Projectors	Projector Initialization	Encoder Architecture	Projector Architecture
HAR/Epilepsy	6	Pytorch Default	Three-block CNN	Two-block CNN
Income/Theorem	6	Pytorch Default	Four-layer MLP	Two-layer MLP
HEPMASS	6	Pytorch Default	Four-layer MLP	Two-layer MLP
MIMIC-III	10	Pytorch Default	Five-block temporal CNN	Three-block temporal CNN

B.2 Neural Network Architectures

HAR/Epilepsy: For LFR, we used a three-block convolutional network from [44, 17] as the representation model f_θ. For the predictors $h_\phi^{(k)}$, we used a single linear layer. For the random projectors $g^{(k)}$, we adopted a similar architecture to the representation model but with slightly decreased complexity, a two-block convolutional network with 16 and 32 channels, followed by two sequential linear layers with a hidden dimension of 256. For all other self-supervised methods, we used the same representation model for a fair comparison. For SimCLR [11] and SimSiam [11], we used the same predictors as LFR, and a 3-layer ReLU network of hidden dimension 512 as a projector. The first two linear layers are followed by a batchnorm layer. To create the contrastive view for SimCLR [11] and SimSiam [11], we adopted the same augmentations as designed in TS-TCC [17].

MIMIC-III: For all methods, we followed the encoder structure from [48] as the representaion model/encoder, with the exception that we used flattened temporal convolutional network (TCN) features followed by a linear layer, which produced the embedding size of 64. We also disabled the L2 normalization in the encoder. For the random projectors in LFR, we adopted a three-block TCN with kernel size of 2, followed by a linear layer with output channel size of 64 for each layer. The two-layer ReLU predictor is shared in LFR, SimCLR and SimSiam with a hidden dimension of 256. We used the same projector and augmentation as in HAR/Epilepsy for SimCLR and SimSiam.

Income/Theorem: For LFR, we followed the setup in [21, 3] and used a 4-layer ReLU network with a hidden dimension of 256 as the representation model, with a single linear layer predictor. The random projector networks had a similar architecture but were less complex, using a 2-layer ReLU network with a hidden dimension of 256. For the contrastive baselines, we employed the same encoder and predictor for a fair comparison, and followed [3] by using a 2-layer ReLU network with a hidden dimension of 256 as projectors. To generate the contrastive views, we used the SCARF [3] augmentation technique to randomly corrupt features with values sampled from their empirical distribution, ensuring that our SimCLR baseline was identical to SCARF.

HEPMASS: For the HEPMASS dataset, we used the same network architecture as for the Income/Theorem datasets but with the output latent dimension of the encoder set to 16.

All supervised baselines use the same representation model as the SSL methods, with the final layer being a linear classification layer. We summarize all architectural related settings of LFR in Table 5

B.3 Details of Training Settings

Table 6 summarizes all the training settings used for LFR, while Table 7 outlines the evaluation settings used for downstream tasks. We used a Logistic Regression classifier for all tabular datasets including Income, Theorem, and HEPMASS, while for MIMIC-III we used a MLP network to predict the length of stay following [48]. For the remaining datasets, we followed prior works such as TS-TCC [17], SimSiam [11], and BYOL [27] by using a linear classifier for classification tasks.

Table 6: Details on LFR Training Settings

Dataset	Optimizer	Batch Size	Learning Rate	Optimizer Parameters	Epochs
HAR/Epilepsy	Adam	128	3e-4	β=(0.9,0.999), wd=3e-4	Train epochs = 200, Predictor epochs = 5
Income/Theorem	Adam	128	1e-3	β=(0.9,0.999), wd=0	Train epochs = 100, Predictor epochs = 1
HEPMASS	Adam	512	1e-6	β=(0.9,0.999), wd=0	Train epochs = 20, Predictor epochs = 1
MIMIC-III	Adam	4096	1e-3	β=(0.9,0.999), wd=5e-4	Train steps = 600, Predictor steps = 5

Table 7: Details on Linear Evaluation Settings of SSL methods

Dataset	Optimizer	Batch size	Learning Rate	Optimizer Parameters	Epochs
HAR/Eplipsey	Adam	128	3e-4	β=(0.9,0.999), wd=3e-4	100
MIMIC-III	Adam	4096	1e-4	β=(0.9,0.999), wd=5e-4	300

Baseline training settings: All self-supervised baselines adopt the same training setting as LFR unless stated otherwise. For DIET with MIMIC-III, we used batch size 512 and trained for 2000 steps with 10 warmup epochs. We reserved 5000 epochs for training the autoencoder on MIMIC-III with 500 warmup epochs. For other self-supervised methods with MIMIC-III, we also added 60 warmup epochs. We summarize the training settings of supervised baselines in Table 8.

Table 8: Details on Supervised Training Settings

Dataset	Optimizer	Batch size	Learning Rate	Optimizer Parameters	Epochs	Augmentations
HAR/Eplipsey	Adam	128	3e-4	β=(0.9,0.999), wd=3e-4	500	None
Income/Theorem	Adam	128	1e-3	β=(0.9,0.999), wd=0	100	None
MIMIC-III	Adam	4096	5e-6	β=(0.9,0.999), wd=5e-4	10	Same as SimCLR and SimSiam

B.4 Reproducibility Notes

TS-TCC: Our results for TS-TCC on the HAR and Epilepsy datasets had several discrepancies with the values reported in the original TS-TCC paper [17]. We discovered that in the official implementation of TS-TCC, the input data was augmented once and then kept the same throughout training, rather than being randomly augmented in each forward pass. We fixed this bug and were able to achieve *better* results. Additionally, we increased the number of training epochs for our supervised baseline, which also led to improved performance. Lastly, we noticed that in the original TS-TCC implementation, the random initialization ablation was evaluated using a randomly initialized linear classification head that was not trained, whereas we evaluated with a trained linear classification layer and saw a significant increase in accuracy for this ablation.

STab: The original STab paper [21] did not provide information about the random DropConnect ratio or training hyperparameters used in their experiments. In our implementation, we used the same training hyperparameters as other SSL methods and tested DropConnect ratios of 0.1, 0.2, 0.4, 0.6, and 0.8, with the results shown in Figure 3. We selected the best-performing ratio for each experiment and reported the corresponding results. We ended up selecting 0.1 for Income and 0.8 for Theorem.

B.5 Computational Resources and Time Spent

The time series experiments with HAR and Epilepsy were conducted on a Tesla V100 GPU with 32 GB of memory, except for TS-TCC which was conducted on a TITAN V with 12 GB of memory.The experiments took a total of 102 GPU hours, including all baseline experiments. The MIMIC-III experiments were conducted with a NVIDIA A100 GPU with 40GB of memory, except for TS-TCC

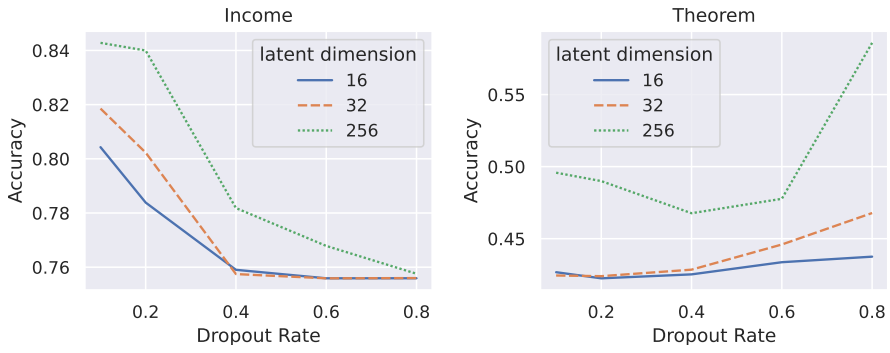

Figure 3: STab accuracy as a function of dropout rate

which was again conducted on a TITAN V with 12 GB of memory, and cost 608 GPU hours, including all baseline experiments. The tabular dataset experiments with Income, Theorem, and HEPMASS were conducted on a NVIDIA TITAN V GPU with 12 GB of memory. The experiments took a total of 70 GPU hours, including all baseline experiments.