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

Augmentation-based SSRL algorithms have pushed the boundaries of performance in computer vision and natural language processing, but are often not directly applicable to other data modalities, and can conflict with application-specific data augmentation constraints. We present an SSRL approach that can be applied to *any* data modality because it does not rely on augmentations. We show that high-quality data representations can be learned by reconstructing random data projections, and evaluate the proposed approach on a range of representation learning tasks that span diverse modalities and real-world applications.

1 Introduction

Learning data representations in a self-supervised manner is commonly associated with well-designed pretext tasks that allow machine learning models to encode useful information from unlabelled data [7, 32, 6]. In several domains, transformation invariance is explicitly encouraged through contrastive or momentum objectives that bring together representations before and after transformation [10, 20, 51, 52]. Despite their strong performance, self-supervised representation learning (SSRL) algorithms that enforce transformation invariance through augmentation are limited in that they do not support generic data types. For instance, data augmentations like Gaussian noise corruption cannot be applied directly to textual data, while image rotation is not applicable in the tabular domain. Thus, augmentation based SSRL techniques are inherently constrained in their cross-domain applicability.

We introduce a SSRL training scheme that requires neither domain-specific data augmentations nor particular architectures as in masking approaches. The proposed 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. Formally, given random projection functions $G = \{\cdots g^{(k)}(\mathbf{x}) \cdots\}$ whose input domains are raw data features $\mathbf{x} \in \mathcal{X}$, 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 predict the random functions' outputs. The representation learning task is 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. We empirically evaluate the effectiveness of LFR on a range of representation learning tasks. The results show that LFR outperforms commonly used domain-agnostic SSRL algorithms [47, 5], and is competitive with many domain-specific approaches that rely heavily on domain knowledge [10, 11, 17, 3, 21]. The remarkable performance demonstrates that learning high-quality data representations from randomness is a feasible alternative when the transformation invariance assumption is hard to enforce in a given application domain.

NeurIPS 2023 Workshop: Self-Supervised Learning - Theory and Practice

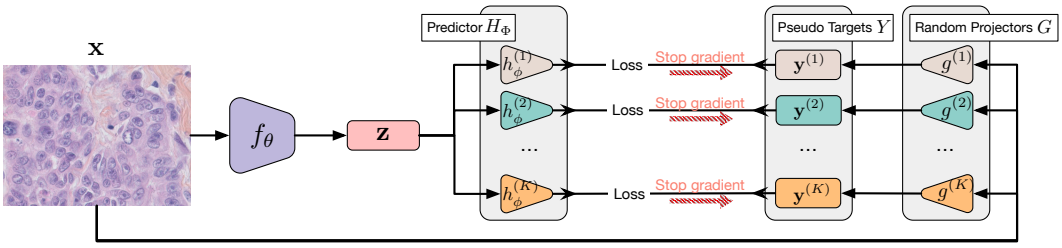

Figure 1: Our proposed architecture for learning from randomness. An input $\mathbf{x}$ is encoded by f_θ into a useful representation $\mathbf{z}$, while also being fed to random projection functions $g^{(k)}$. Simple, learnable predictor functions $h_\phi^{(k)}$ try to match the outputs $\mathbf{y}^{(k)}$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

2 Representation Learning from Random Data Projectors

We frame the representation learning task as the following: given observed data $X = \{\cdots \mathbf{x}_i \cdots\}$ from the feature domain $\mathcal{X}$, the 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 $\mathbf{x}_i \in \mathcal{X}$.

Pretext Task: Multi-objective Learning from Randomness We take the "arbitrary downstream tasks $g(\mathcal{X})$" literally and propose to model them as random data projections, creating the surprising pretext task shown in Figure 1. It contains three components: 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 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 [20, 11], and is essentially a lower-bound of the maximum likelihood estimation objective (see Appendix B.1).

We adopt the Expectation-Maximization method as a training strategy for Objective (1):

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

$$\underset{\theta}{\operatorname{argmin}} \sum_i \sum_k \mathcal{D}\left[g^{(k)}(\mathbf{x}_i), h_\phi^{(k)}(f_\theta(\mathbf{x}_i))\right] + \lambda_1 \Omega_1(\theta), \tag{2}$$

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

$$\underset{\Phi}{\operatorname{argmin}} \sum_i \sum_k \mathcal{D}\left[g^{(k)}(\mathbf{x}_i), h_\phi^{(k)}(f_\theta(\mathbf{x}_i))\right] + \lambda_2 \Omega_2(\Phi). \tag{3}$$

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 [11, 20, 44].

Divergence Measure: Batch-wise Barlow Twins While there are several common choices of divergence $\mathcal{D}$ which could be used in Objective (1), they are often inadequate for enforcing identifications between subtly different points as is crucial for representation learning tasks; Mean Squared Error downweights the importance of small errors, Cross Entropy is ill suited for regression tasks, while the Contrastive [14] and Triplet [41] losses introduce significant stochasticity. The Barlow Twins

loss [56] has garnered much interest in the SSRL literature as it disentangles learned representations through redundancy reduction, and can scale to very high-dimensional vectors. 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

$$\mathcal{L}_{\text{BBT}} = \sum_k \sum_i \Big[\Big(1 - c_{ii}^{(k)}\Big)^2 + \lambda \sum_{j \neq i} {c_{ij}^{(k)}}^2 \Big], \tag{4}$$

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

$$c_{ij}^{(k)} = \frac{{\mathbf{y}_i^{(k)}}^\top \hat{\mathbf{y}}_j^{(k)}}{\big\|\mathbf{y}_i^{(k)}\big\|_2 \big\|\hat{\mathbf{y}}_j^{(k)}\big\|_2}, \quad \mathbf{y}_i^{(k)} = g^{(k)}(\mathbf{x}_i), \quad \hat{\mathbf{y}}_i^{(k)} = h_\phi^{(k)}\left(f_\theta(\mathbf{x}_i)\right), \tag{5}$$

and $\mathbf{y}_i^{(k)}, \hat{\mathbf{y}}_i^{(k)} \in \mathbb{R}^{d^{(k)}}$. Compared to the loss in [56], Equation (4) has an extra summation over the ensemble k, and 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.

Diversity Encouragement on Random Data Projectors In practice we create multiple data projections $g^{(k)}(\mathcal{X}) \in G$ by randomly initializing neural networks that reuse the architecture design of f_θ, but scaled down, which avoids the need to make domain-specific choices about the projectors. To promote diversity amongst the tasks represented by $g^{(k)}$, we systematically pick K diverse projectors from $N \gg K$ randomly generated candidates. Given 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

$$Y^{(k)} = g^{(k)}(X)/\|g^{(k)}(X)\|_2, \quad Y^{(k)} \in \mathbb{R}^{m \times d^{(k)}}. \tag{6}$$

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

$$A^{(k)} = Y^{(k)}(Y^{(k)})^\top, \quad A^{(k)} \in \mathbb{R}^{m \times m}. \tag{7}$$

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 $\mathbf{a}^{(k)}$,

$$\underset{\mathbf{s}}{\text{argmax}} \, |\det(B)| \quad \text{s.t.} \quad B = \tilde{A}\tilde{A}^\top, \quad \tilde{A} = \Big[\mathbf{a}^{(k)} \;\Big|\; k \in [0, N], s_k = 1, \sum_{k'} s_{k'} = K \Big], \tag{8}$$

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 [9] can find good solutions in reasonable time. 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 B.2.

3 Experiments and Evaluation

Datasets We consider diverse data types to show the wide applicability of *learning from randomness*. See Appendix C.1 for details. **Time Series:** Human Activity Recognition (HAR) [2], Epileptic Seizure Recognition (Epilepsy) [1], and MIMIC-III [22]. **Tabular:** Adult Income (Income) [30], First Order Theorem Proving (Theorem) [8], and HEPMASS [4]. **Computer vision:** Kvasir [37].

Baseline methods We compared our LFR method to popular SSRL approaches, including domain-agnostic (e.g., DACL [47]) and modality-specific ones (e.g., SCARF [3] for tabular, and TS-TCC [17] for time-series). Appendix C.2 provides additional details and descriptions.

Evaluation All the downstream tasks in our study are treated as classification problems. To evaluate the quality of the pre-trained representations, we employed simple supervised classifiers that are specific to each dataset, see Appendix C.3. The classifiers were trained on the frozen representations of the training set and evaluated on the test set. We conducted multiple random runs and report the mean and standard deviation, using 5 runs for tabular datasets and 3 runs for others.

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 [50]. For the MIMIC-III dataset, we employed the Temporal Convolutional Network used by NCL [54]. For the Tabular datasets, we used 4-layer MLPs as in SCARF [3]. For Kvasir, we employed the ResNet18 architecture [25]. 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 C.4.

Performance Across Data Modalities Table 1 shows the performance of LFR and baselines across multiple modalities. For modalities without standardized augmentation pipelines such as medical images, time series, and tabular data, LFR had the strongest performance among SSRL methods. Hence, we have shown that learning useful representations from random projections is feasible.

Table 1: Performance comparison across various data modalities and application domains. Results of the best self-supervised learning methods are in bold. Shaded columns denote medical applications.

		Time series			Tabular			Image
		HAR	Epilepsy	MIMIC-III	Income	Theorem	HEPMASS	Kvasir
	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	83.2±0.2
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	28.9±5.7
	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**	72.4±0.6
	DIET	88.6±1.3	96.8±0.3	33.8±5.2	82.2±0.4	47.1±0.5	-	71.3±0.9
	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	72.6±1.4
	SimCLR	87.8±0.4	97.4±0.2	44.1±0.1	-	-	-	72.1±0.3
	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	72.0±0.1
	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	**74.9±0.6**

Medical Domains The standard image augmentation pipeline used in other self-supervised methods is heavily tailored towards natural images and does not lead to superior performance on medical images (Kvasir). Similarly, the standard augmentations proposed for time series do not capture the unique characteristics of online patient monitoring data (MIMIC-III). Overall, our results suggest that LFR is an effective method for learning meaningful representations, even for specific applications that would otherwise require domain knowledge.

Further investigations of our method are provided in the appendix. We explore the impact of projector diversity in Appendix D.1, and perform hyperparameter ablations in Appendix D.2. In Appendix D.3 we compare methods based on finetuning evaluation, rather than downstream evaluation, and in Appendix D.4 we use visualizations to explore the features that random projectors encode.

4 Conclusion

This paper presents a novel self-supervised representation learning framework that is both modality-agnostic and application-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). Surprisingly, such scenarios are quite common in crucial application domains, including healthcare, as highlighted in our work. For specific applications like natural images where domain knowledge allows for meaningful data augmentations, contrastive-learning-based SSRL methods can likely outperform random projectors. Overall, we treat LFR as a great complement to the SSRL literature that can fill the gap of data augmentation-free SSRL and thereby satisfy the needs of many crucial applications.

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. 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.

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

[10] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 1597–1607, 13–18 Jul 2020.

[11] 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.

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

[13] 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.

[14] 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.

[15] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015.

[16] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, 2021.

[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] M. Elgendi, M. U. Nasir, Q. Tang, D. Smith, J.-P. Grenier, C. Batte, B. Spieler, W. D. Leslie, C. Menon, R. R. Fletcher, N. Howard, R. Ward, W. Parker, and S. Nicolaou. The Effectiveness of Image Augmentation in Deep Learning Networks for Detecting COVID-19: A Geometric Transformation Perspective. *Frontiers in Medicine*, 8:629134, 2021.

[19] L. Ericsson, H. Gouk, and T. Hospedales. Why Do Self-Supervised Models Transfer? On the Impact of Invariance on Downstream Tasks. In *Proceedings of the 33rd British Machine Vision Conference 2022*, 2022.

[20] 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. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[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] 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.

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

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

[27] 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.

[28] S. Kalra, J. Wen, J. C. Cresswell, M. Volkovs, and H. R. Tizhoosh. Decentralized federated learning through proxy model sharing. *Nature Communications*, 14(1):2899, May 2023.

[29] M. Kang, H. Song, S. Park, D. Yoo, and S. Pereira. Benchmarking self-supervised learning on diverse pathology datasets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3344–3354, June 2023.

[30] R. Kohavi. Scaling up the accuracy of Naive-Bayes classifiers: A decision-tree hybrid. In *Second International Conference on Knowledge Discovery and Data Mining*, volume 96, pages 202–207, 1996.

[31] L. Kong, C. de Masson d'Autume, L. Yu, W. Ling, Z. Dai, and D. Yogatama. A mutual information maximization perspective of language representation learning. In *International Conference on Learning Representations*, 2020.

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

[33] 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.

[34] D. Luo, W. Cheng, Y. Wang, D. Xu, J. Ni, W. Yu, X. Zhang, Y. Liu, Y. Chen, H. Chen, and X. Zhang. Time series contrastive learning with information-aware augmentations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(4):4534–4542, 2023.

[35] K. Majmundar, S. Goyal, P. Netrapalli, and P. Jain. MET: Masked Encoding for Tabular Data. In *NeurIPS 2022 First Table Representation Workshop*, 2022.

[36] 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.

[37] K. Pogorelov, K. R. Randel, C. Griwodz, S. L. Eskeland, T. de Lange, D. Johansen, C. Spampinato, D.-T. Dang-Nguyen, M. Lux, P. T. Schmidt, et al. Kvasir: A multi-class image dataset for computer aided gastrointestinal disease detection. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, pages 164–169, 2017.

[38] J. M. S. Prewitt. Object enhancement and extraction. *Picture processing and Psychopictorics*, 10(1):15–19, 1970.

[39] S. Purushwalkam and A. Gupta. Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[40] 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.

[41] 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.

[42] Y. Shen, Y. Luo, D. Shen, and J. Ke. RandStainNA: Learning Stain-Agnostic Features from Histology Slides by Bridging Stain Augmentation and Normalization. In *Medical Image Computing and Computer Assisted Intervention*, pages 212–221. Springer, 2022.

[43] H. Sowrirajan, J. Yang, A. Y. Ng, and P. Rajpurkar. MoCo Pretraining Improves Representation and Transferability of Chest X-ray Models. In *Proceedings of the Fourth Conference on Medical Imaging with Deep Learning*, volume 143, pages 728–744, 2021.

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

[45] T. Ucar, E. Hajiramezanali, and L. Edwards. SubTab: Subsetting Features of Tabular Data for Self-Supervised Representation Learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 18853–18865, 2021.

[46] 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.

[47] V. Verma, T. Luong, K. Kawaguchi, H. Pham, and Q. Le. Towards domain-agnostic contrastive learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 10530–10541, 2021.

[48] 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.

[49] L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning*, 2013.

[50] 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*, pages 1578–1585. IEEE, 2017.

[51] J. Wei and K. Zou. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 2019.

[52] X. Wu, S. Lv, L. Zang, J. Han, and S. Hu. Conditional bert contextual augmentation. In *International Conference onComputational Science*, pages 84–95, 2019.

[53] T. Xiao, X. Wang, A. A. Efros, and T. Darrell. What Should Not Be Contrastive in Contrastive Learning. In *International Conference on Learning Representations*, 2021.

[54] 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.

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

[56] 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.

[57] 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.

[58] 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 Background and Related Works

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. In computer vision (CV) and natural language processing (NLP), SSRL has gained considerable attention primarily due to the availability of extensive amounts of unlabeled data.

Adopting the definitions outlined in [6], SSRL methods in the field of CV can be grouped into four main categories: deep metric learning [10], self-distillation [20, 10], canonical correlation analysis [56], and masked image modeling [23]. Among these, the first three rely on creating positive views of a given image to learn invariances. This is achieved by creating augmented versions of the same image and enforcing the latent embeddings of these versions to be identical, with the underlying assumption that the semantic meaning of the original image is invariant across different views. Recently, masked image modeling has emerged as a popular SSRL approach due to the success of Vision Transformers [16]. These methods predict masked vision tokens [23] or pixel patches [15], which have proven useful in learning representations.

In NLP, masked language modelling is very prominent. The dominant models such as BERTs and GPTs are trained to predict masked language tokens, which encourages the model to encode contextual information and reconstruct its inputs. Other efficient language-focused pretext tasks, such as maximizing the mutual information between a global sentence representation and n-grams in the sentence [31], occasionally emerge in the literature.

Despite the remarkable success in CV and NLP, the effectiveness of SSRL in other data modalities, such as tabular data, has been limited [3, 6]. One challenge for SSRL methods relying on transformation invariance assumptions lies in identifying and designing appropriate augmentations. 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. For example, augmentations designed for natural images may not be suitable for medical images with low color variation, leading to unrealistic results and unsatisfactory performance [29]. Extensive research efforts have been dedicated to designing effective modality- and application-specific augmentation techniques. We summarize the most common augmentation techniques used in contrastive learning and compare them across domains in the following paragraphs.

Computer vision augmentation techniques Image datasets benefit from a wide range of semantically similar augmentations, including random cropping and resizing, horizontal flipping, color jittering, converting to grayscale, and Gaussian blurring [24, 10, 11]. However, the best performing augmentations are often dataset-specific [19] and require domain knowledge to determine. For instance, enforcing color invariance by grayscale conversion may not be beneficial for flower datasets [57], but it can improve performance for ImageNet data [10]. Furthermore, augmentations designed for natural images may not be suitable for medical domains [18]. For instance, MoCo-CXR [43] focuses on chest X-ray images and uses random rotation and horizontal flipping instead of random crops, Gaussian blur, and random grayscale, as the latter may alter disease labels or are not meaningful for grayscale X-ray images. In another study, the authors proposed specific color space transforms for pathology images [29], as naïve color-jittering may produce unrealistic resulting images [42]. Finally, the choice of augmentations can even be task-specific [53]. As an example, aggressive cropping may be suitable for image classification, but may not be optimal for image recognition tasks [39].

Time series Time series data often contains underlying patterns that are not easily identifiable by humans, unlike images with recognizable features [34]. 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 [46]. Other researchers have focused on bio-signals and introduced channel augmentations that preserve the semantic information in the context [36, 12]. Neighbourhood contrastive learning [54] 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 [57], 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 [27].

Tabular SSRL methods are understudied in the tabular domain as designing effective semantic-preserving augmentations is particularly challenging for structured data [55]. Like for time-series, it is often difficult for a human to determine if two views should be considered semantically equivalent, and unlike computer vision small changes to individual features can drastically change the content. SubTab [45] proposed to generate positive views through 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.

While masking approaches offer general applicability to all data modalities, the most effective frameworks often rely on transformer-based backbones for optimal performance [35, 23, 13]. In this work, we focus on a model-agnostic SSRL approach. Alternatively, classic autoencoder-based methods provide an alternative to SSRL without relying explicitly on transformation invariance [26, 48, 58]. 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 [33].

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.

B Method

B.1 Analysis of Learning Objective

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,

$$\begin{aligned}\sum_i \log p\left(Y_i \mid \mathbf{x}_i\right) &= \sum_i \sum_k \log \int_{\mathbf{z}_i} p\left(\mathbf{y}_i^{(k)} \;\middle|\; \mathbf{z}_i\right) p\left(\mathbf{z}_i \mid \mathbf{x}_i\right) d\mathbf{z}_i \\ &\geq \sum_i \sum_k \int_{\mathbf{z}_i} p\left(\mathbf{z}_i \mid \mathbf{x}_i\right) \log p\left(\mathbf{y}_i^{(k)} \;\middle|\; \mathbf{z}_i\right) d\mathbf{z}_i = \sum_i \sum_k \log p\left(\mathbf{y}_i^{(k)} \;\middle|\; \mathbf{z}_i = f_\theta(\mathbf{x}_i)\right),\end{aligned} \tag{9}$$

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 (9), 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. Other options for $\mathcal{D}$ are discussed in the main text.

B.2 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 · K different random projectors and select K diverse projectors g^1, ..., g^K using DPP as introduced in Section 2
Initialize K predictors h_φ^1, ..., h_φ^K, one for each projector.
for epoch_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 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_θ(B)
    for k = 1 to K do
        Compute output representations: h_φ^k(Z)
        Compute representation from projector k: g^k(Z)
        Compute loss L_BBT
    end for
end procedure
```

C Implementation Details

C.1 Dataset Summary

Table 2 provides a summary of all datasets used in our experiments, along with the corresponding downstream tasks and evaluation metrics. Detailed descriptions are given for each modality below.

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 [54] 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) [30], First Order Theorem Proving (Theorem) [8], and HEPMASS [4]. For Income, a binary classification problem, we followed the data preprocessing steps in [45]. 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.

Computer vision: We tested on Kvasir [37], a medical image dataset consisting of 8,000 images of the gastrointestinal tract. There are eight balanced classes including seven disease types as well as healthy images. We followed [28] to resize all images to 100×80 pixels and split the data into 6,000 images for training and 2,000 for testing.

Table 2: Dataset descriptions.

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	Patient 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
Kvasir	Image	Medical Images	6000	2000	Multi-class classification	Accuracy

C.2 Baseline Algorithms

Table 3 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 (e.g. tabular data) or domains (e.g. medical images). Of particular note, for 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.

C.3 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 [54]. For tabular datasets, we used logistic regression, similar to the approach in STab [21]. For the remaining datasets, a linear classifier was employed. The classifiers were trained on the frozen representations of the training set and evaluated on the test set, which is the most commonly used protocol. Accuracy is our primary evaluation metric, except for MIMIC-III where we adopted linearly weighted Cohen's Kappa as in [54], 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 others.

C.4 Neural Network Architectures

HAR/Epilepsy: For LFR, we used a three-block convolutional network from [50, 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 [10] and SimSiam [10], we

Table 3: Baseline methods

Category	Method	Description
Domain-agnostic	Autoencoder [40]	Encoder/decoder with low dimensional latents trained via the reconstruction loss.
	DACL [47]	Self-supervised learning method that uses mix-up as data augmentation across modalities.
	DIET [5]	Self-supervised learning method that predicts the datum index as a pretext task.
Domain-specific augmentations	SimCLR [10]	Contrastive learning method with both positive and negative pairs.
	SimSiam [11]	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].

Table 4: 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 TCN	Three-block TCN
Kvasir	6	β Initializtion + Weight Dropout	ResNet18	Four-layer CNN

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 [10] and SimSiam [10], we adopted the same augmentations as designed in TS-TCC [17].

MIMIC-III: For all methods, we followed the encoder structure from [54] 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.

Kvasir: For LFR, we used ResNet18 [25] as the representation model, with an output dimension of 2048 for all datasets. The predictors are 4-layer ReLU networks with a hidden dimension of 256. For the random projector networks, we adopted a 4-layer CNN of channels $[3, 8, 16, 32]$, each layer is followed by a ReLU, and every two layers are followed by max-pooling. A linear layer is used to map the output to 2048 dimensions. For the contrastive baselines, we adopted the same representation model for a fair comparison and used 3-layer ReLU networks with hidden dimension 256 as projectors. To create the contrastive views for Kvasir, we employed the augmentation set adopted by SimSiam [10] for ImageNet (`RandomResizedCrop`, `RandomHorizontalFlip`, `ColorJitter`, `RandomGrayscale`, `GaussianBlur`). We applied the same set of augmentations to the implementation of DIET [5]. All other settings are the same as with the ResNet18 model.

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 4

C.5 Details of Training Settings

LFR training settings: Table 5 summarizes all the training settings used for LFR, while Table 6 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 [54]. For the remaining datasets, we followed prior works such as TS-TCC [17], SimSiam [10], and BYOL [32] by using a linear classifier for classification tasks.

Table 5: 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
Kvasir	SGD	256	1e-4, cosine decay	momentum=0.9, wd=5e-4	Train epochs = 400, Predictor epochs = 5

Table 6: Details on Linear Evaluation Settings of SSL methods

Dataset	Optimizer	Batch size	Learning Rate	Optimizer Parameters	Epochs
HAR/Epilepsy	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
Kvasir	SGD	256	1e-3, cosine decay	momentum=0.9, wd=0	100

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 7.

Table 7: Details on Supervised Training Settings

Dataset	Optimizer	Batch size	Learning Rate	Optimizer Parameters	Epochs	Augmentations
HAR/Epilepsy	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
Kvasir	SGD	256	1e-2, cosine decay	momentum=0.9, wd=0	600	Same as SimCLR and SimSiam

D Additional Experimental Results

D.1 Impact of Random Data Projector Diversity

In this section, we analyze the impact of the diversity of random data projectors on LFR using the Kvasir dataset. Projector diversity can be influenced at two stages: initialization and selection. For projector selection, we used the Determinantal Point Process illustrated in Section 2. Regarding projector initialization, we employ two techniques: Beta initialization and weight dropout.

Beta initialization: We drew inspiration from the Prewitt operator [38] and initialized the weights of 2D convolutional layers to create diverse targets that emphasize various edge features. To achieve this, we used a scaled version of the Beta distribution with parameters $\alpha = 0.5$ and $\beta = 0.5$ to initialize the convolutional layer weights to be close to -1 and 1.

Weight dropout: We utilized DropConnect [49] to initialize the target networks. This involved randomly setting a fraction of weights to zero with a corruption rate of 0.4. Unlike standard DropConnect used for network regularization, we froze the weights after initialization to enhance diversity in the target representations.

Results: We compared the effectiveness of diversity encouragement at both the initialization and selection stages through an ablation on Kvasir. Our results, shown in Figure 2, demonstrate that promoting diversity in the projector set at both initialization and selection helps with the downstream performance. These findings underscore the importance of diversity in the projector set and highlight the effectiveness of our proposed methods for promoting diversity in LFR.

D.2 Ablation Study

In this section, we investigate the impact of hyperparameters on the model's performance. To provide a more focused analysis of the trends observed in our ablation study, we present the results for Kvasir,

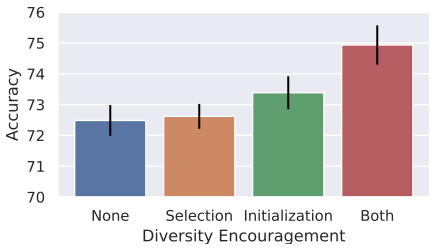

Figure 2: Effect of target diversity

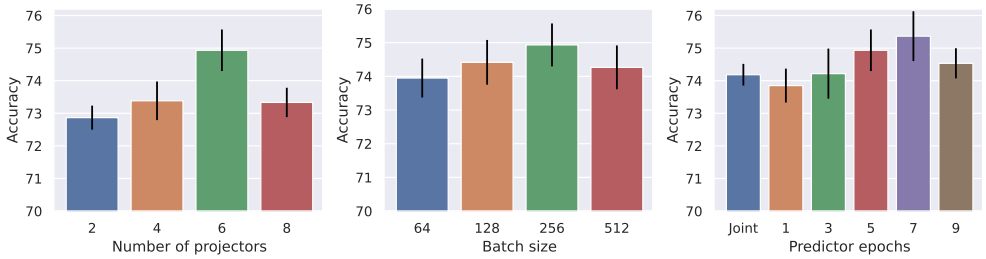

Figure 3: Test accuracy with different hyperparameters on Kvasir. **Left:** Number of random projectors. **Middle:** Batch size. **Right:** Predictor training setting.

which is representative of the other datasets. While we have observed similar trends in other datasets, the significance may vary depending on the specific characteristics of each.

Number of projectors: We evaluated the impact of the number of random projectors K on linear evaluation accuracy using Kvasir data. We plotted the mean accuracy and standard deviation across 3 runs for $K = 2$, 4, 6 and 8 in Figure 3. The results indicate that the learned representation has a higher linear probing accuracy as the number of projectors increases until it reaches 6. Beyond $K = 6$, there is the possibility of reoccurring representations generated by the projectors, which can introduce bias into the encoder's training process. This could happen as the gradient descent optimization process might excessively favor redundant features, potentially leading to decreased performance. Based on these findings, we used $K = 6$ for our experiments on Kvasir.

Batch size: Because the selection of diverse projectors in LFR relies on one representative batch of data, we examine the sensitivity of LFR to training batch sizes. We plotted the linear accuracy with batch sizes of 64, 128, 256, and 512, and reported the mean accuracy and standard deviation across 3 runs in Figure 3. Our results indicate that LFR has relatively stable accuracies across batch sizes, with no significant difference between them. However, the best performing batch size on the Kvasir dataset was 256, and thus we used this batch size for our subsequent experiments.

Predictor training epochs: Another option in LFR is how the encoder f_θ and predictors H_Φ are trained. We tested joint training, where all models are updated together, and alternating training with various numbers of epochs for the predictors between each encoder epoch. From Figure 3, our findings on Kvasir indicate that updating the predictors for several epochs can improve performance compared to joint training, suggesting that more optimal predictors provide better learned representations.

Embedding dimension: We conducted an analysis on the effect of latent dimension on accuracy. Figure 4 (Left) demonstrates that as the latent dimension increases, there is a corresponding improvement in performance accuracy. However, when the latent dimension is more than 2048, the performance increase is minimal. Therefore, we used 2048 for our experiments.

To complement the image-based results on Kvasir, we also used the Theorem dataset to evaluate the performance of LFR and baseline SSRL approaches across latent dimension sizes. Figure 4 (Right) 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.

Sample size for projector selection As illustrated above, we use a sufficiently large set of data to select a diverse set of random projectors. In this section, we investigated the effect of the sample

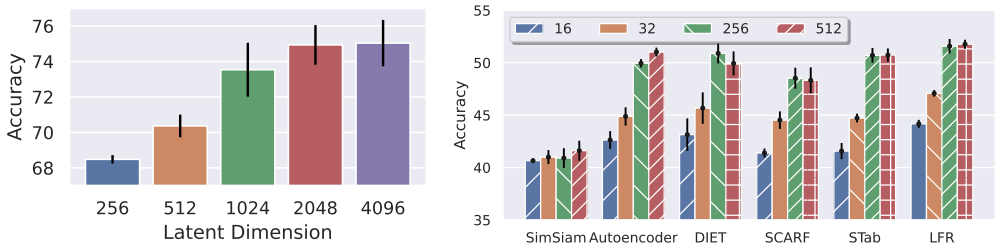

Figure 4: Effect of embedding dimension. **Left:** On Kvasir, LFR benefits from increased latent dimensionality but plateaus when the dimension is high enough. **Right:** A comparison across methods on the Theorem dataset.

size for random projector selection with the Kvasir dataset. As shown in Table 8, the sample size does not heavily affect the accuracy (the fluctuation is less than 1.0%). Therefore, for the ease of implementation, we use the same size of the batch size in training, i.e. 256 for Kvasir.

Table 8: Effect of sample size during random projector selection

Sample size	64	128	256	512
Accuracy	74.1 ± 0.4	74.0 ± 0.4	74.9 ± 0.6	74.1 ± 0.4

D.3 Fine-tuning performance

In Section 3 we evaluated SSRL methods on downstream tasks using linear evaluation. SSRL methods are sometimes finetuned to a downstream task, so in Table 9 we compare LFR's finetuning performance to baseline algorithms.

During the finetuning phase, we employed a linear layer combined with labeled data to enhance the performance of the encoder. The architecture setup was consistent with that detailed for downstream evaluation. However, recognizing the significant volume of data in larger datasets such as MIMIC-III and HEPMASS, we chose a semi-supervised learning approach in those cases. In this strategy, we randomly selected 10% of labeled data from these datasets to facilitate the fine-tuning process. This decision was driven by the computational resources required when dealing with extensive data. Conversely, for smaller datasets, we conducted fine-tuning using the complete set of available labeled data, enabling us to evaluate the model's performance across the entirety of the datasets.

Through the fine-tuning 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.

Table 9: Performance comparison among the SSRL methods with finetuning.

	Time series			Tabular			Image
	HAR	Epilepsy	MIMIC-III	Income	Theorem	HEPMASS	Kvasir
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	83.2 ± 0.2
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	75.0 ± 0.8
DIET	**95.6 ± 0.5**	97.8 ± 0.1	48.4 ± 0.1	85.2 ± 0.1	52.4 ± 0.9	-	74.4 ± 0.3
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	74.5 ± 0.6
SimCLR	93.7 ± 1.1	97.8 ± 0.2	48.6 ± 0.8	-	-	-	74.5 ± 0.6
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	**75.5 ± 0.7**

D.4 Random Projectors Visualization

To illustrate the behaviour of individual random projections, and the role of diversity selection, we trained LFR on a small image-based dataset, CIFAR-10[1], and examined the nearest-neighbour representations for different projectors. First we selected a query image at random, then for each of 100 randomly initialized projectors, we encoded all training images, and found the nearest encoded representations to the query. Distances in embedding space are measured by cosine similarity. We then visually compared the nearest neighbours that were selected by pairs of projectors which were deemed most diverse or similar according to the DPP criterion from Equation (8). The pairs with the highest diversity (Diverse 1 and Diverse 2) and the highest similarity (Similar 1 and Similar 2) are shown in Figure 5.

The results in Figure 5 show that our similarity measure is effective at selecting projectors that focus on diverse features. For example, in Figure 5 on the frog query (top left) we see that the projector in the first row represented a white “edge" feature, since the nearest neighbours all share that feature from the query but are otherwise semantically different. The projector in the second row appears to focus on shape, and its nearest neighbours are very different from the first projector. On the other hand, looking at the bottom right set of images, the most similar projector pair selects very similar nearest neighbors with 2 out of 5 nearest neighbors being identical. This qualitative study suggests that promoting diversity in the projectors can lead to the capture of a broader range of features. Then, diverse features will be available to the representation model during training, potentially resulting in a richer set of semantic knowledge and ultimately better performance on downstream tasks. This interpretation is supported by the evidence in Section D.2 that encouraging projector diversity results in better performance on linear evaluation accuracy.

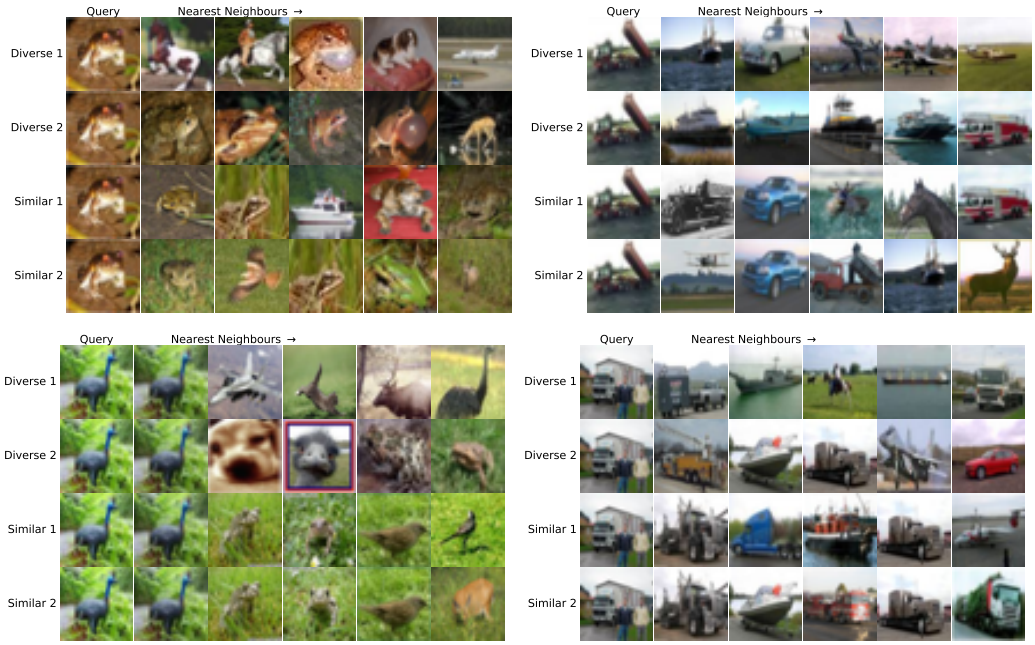

Figure 5: Nearest neighbors identified by randomly initialized projector models. Each row shows the query image (left) and its five nearest neighbours identified by a projector. The top two rows display the most dissimilar projectors, while the bottom two rows give the most similar projectors.

E 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.

[1]CIFAR-10 was chosen for this qualitative study rather than the other datasets we used because it has human-interpretable features and visualizations.

The experiments took a total of 102 GPU hours, including all baseline experiments. The MIMIC-III experiments were conducted with an NVIDIA A100 GPU with 40GB of memory, except for TS-TCC which was again conducted on a TITAN V with 12 GB of memory, and cost 608 GPU hours, including all baseline experiments. The Kvasir experiments were conducted using a Tesla V100 GPU with 32 GB of memory, and they took a total of 1095 GPU hours, including all baseline experiments. The tabular dataset experiments with Income, Theorem, and HEPMASS were conducted on an NVIDIA TITAN V GPU with 12 GB of memory. The experiments took a total of 70 GPU hours, including all baseline experiments.