Towards Self-Supervised Foundation Models for Critical Care Time Series

Katja Naasunnguaq Jagd
Technical University of Denmark
knjagd@hotmail.com

Rachael DeVries
University of Copenhagen
Novo Nordisk A/S
rachael.devries@bio.ku.dk

Ole Winther
Technical University of Denmark
University of Copenhagen
ole.winther@bio.ku.dk

Abstract

Domain-specific foundation models for healthcare have expanded rapidly in recent years, yet foundation models for critical care time series remain relatively underexplored due to the limited size and availability of datasets. In this work, we introduce early-stage pre-trained foundation models for critical care time series based on the Bi-Axial Transformer (BAT), trained on pooled electronic health record datasets. We demonstrate effective transfer learning by fine-tuning the models on a dataset distinct from the training sources for mortality prediction, where it outperforms supervised baselines, particularly for smaller datasets. These contributions highlight the potential of self-supervised foundation models for critical care times series to support generalizable and robust clinical applications in resource-limited settings. **Code Availability**: `https://github.com/Katja-Jagd/YAIB`

1 Introduction and related works

Foundation models built on Transformers [1] have achieved remarkable success across several domains, including natural language processing [2, 3] and computer vision [4, 5, 6]. Despite this success, general-purpose foundation models often perform poorly on healthcare applications, which are impeded by the complexities of medical data and the scarcity of publicly available labeled datasets [7]. Consequently, the development of healthcare-specific foundation models has accelerated rapidly since 2018, targeting diverse applications such as clinical natural language processing [8, 9], medical imaging [10, 11], omics analysis [12, 13], video and audio interpretation [14, 15] and multi-modality [16, 17]. Foundation models for Electronic Health Records (EHRs) have primarily focused on structured EHR data, such as modelling and predicting ICD-10 codes [18, 19]. Although these models can scale to large patient populations, they remain limited in their ability to capture physiological patterns [20]. One area that remains relatively underexplored is the development of foundation models for critical care time series data [21]. Models trained in this domain are often hindered by issues of reproducibility [22], limited by simple learning paradigms that rely on one supervised task [23, 24, 25] or are trained on a small, homogeneous dataset [26, 27, 28]. It has been demonstrated that these models do not transfer well to new clinical settings [21]. Recent work from Burger et al. [20] are the first to combine pooling of several critical care time series datasets with self-supervised pre-training, and thereby delivering a critical care time series foundation model, ICareFM [20]. Both the model and code are currently not open-source.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

To push critical care time series data towards foundation model development, we modify the Bi-axial Transformer (BAT) architecture, presented in the work of DeVries et al. [29], for self-supervised pre-training and conduct all experiments within the Yet Another ICU Benchmark (YAIB) framework [22], ensuring transparency, reproducibility and pooling of datasets. Our contributions are as follows: 1) We release the first ICU-specific models for foundational capabilities with an open-source, reproducible repository, 2) we demonstrate its ability to transfer effectively to an unseen dataset distinct from the training sources and a new downstream clinical task, outperforming supervised baselines, and 3) we show that it performs especially well in low-data regimes, highlighting the potential of self-supervised pre-training for resource-limited clinical settings that may not achieve robust performance from supervised models.

2 Methodology

Let $\mathcal{D} := \bigcup_{k=1}^{K} \mathcal{D}_k$, where a critical care dataset $\mathcal{D}_k$ is defined as a set of tuples, $\{(\mathbf{X}_i,\ \mathbf{p}_i,\ \mathbf{h}_i)\}_{i=1}^{N_k}$. Every patient $i \in N_k$ is represented with a multivariate time series of measurements $\mathbf{X}_i \in \mathbb{R}^{T_i \times D}$ from monitoring devices and laboratory tests, with K datasets containing the same measurement types D. As each $\mathbf{X}_i$ is irregularly sampled across both t and d, we also record measurement times $\mathbf{h}_i \in \mathbb{R}^{T_i}$. The time series is supplemented with patient demographics $\mathbf{p}_i$ which are fixed for the duration of T.

In critical care applications, it is often of interest to predict a class label, such as $y_i \in \{0, 1\}$ for mortality status, given $(\mathbf{X}_i, \mathbf{p}_i, \mathbf{h}_i)$. Popular approaches split a single dataset $\mathcal{D}_1$ into train/test/validation subsets, and perform supervised training to predict y_i. We hypothesize that combining and pre-training on additional datasets $\{\mathcal{D}_2, ...\mathcal{D}_K\}$, in a self-supervised fashion will result in a model that learns richer, more generalized representations of sensor identities and their measurement distributions.

2.1 Pretraining and fine-tuning

We begin with self-supervised pre-training with forecasting as the prediction task. Specifically, $\mathbf{X}_i$ sampled from the auxiliary datasets $\{\mathcal{D}_2, ...\mathcal{D}_K\}$ are split into an observation window $\mathbf{X}_i^{\text{obs}} \in \mathbb{R}^{T^{\text{obs}} \times D}$ and a forecasting window $\mathbf{X}_i^{\text{for}} \in \mathbb{R}^{T^{\text{for}} \times D}$. The learning objective is to predict $\mathbf{X}_i^{\text{for}}$ given $(\mathbf{X}_i^{\text{obs}}, \mathbf{p}_i, \mathbf{h}_i^{\text{obs}})$. Due to the irregular sampling across h_i and sparsity in X_i, the forecasting must be sample-specific. To handle sparsity and ensure temporal alignment, we leverage a masked loss:

$$\mathcal{L}^{\text{Pre}} = \frac{1}{\sum_{k=2}^{K} N_k} \sum_{k=2}^{K} \sum_{i=1}^{N_k} \left\| \mathbf{M}_i^{\text{for}} \odot \left(\hat{\mathbf{X}}_i^{\text{for}} - \mathbf{X}_i^{\text{for}} \right) \right\|_F^2$$

where the mask $\mathbf{M}_i^{\text{for}}$ ensures that only observed values in the forecasting window contribute to the loss.

We then fine-tune and test on $\mathcal{D}_1$ with supervised learning, where the objective is our initial goal of predicting the mortality status $y_i \in \{0, 1\}$ as a binary classification task. We used the binary cross-entropy loss:

$$\mathcal{L}^{\text{Fine}} = -\frac{1}{N_1} \sum_{i=1}^{N_1} \Big[y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \Big]$$

2.2 Bi-axial Transformer (BAT)

The Bi-Axial Transformer (BAT) [29] is a generalization of the Axial Transformer [30] with parallel axial attention along each dimension. BAT attends to both the temporal and clinical feature axes through axial attention mechanisms, and explicitly accounts for missing values. The model architecture is described in detail in Appendix A.1. BAT shows promising results in prediction tasks for clinical irregular multivariate time series data with state-of-the-art performance for sepsis classification, and competitive performance on mortality prediction when compared to several models. Investigation into model attention maps revealed evidence of BAT learning from informative missingness, and it showed an increased robustness to sparsity in comparison to other Transformer models. These properties make BAT a strong candidate for modeling clinical multivariate time series. BAT's prediction head was adapted in this work to support both binary classification, outputting $\hat{y} \in \{0, 1\}$, and forecasting, outputting $\hat{\mathbf{X}}^{\text{for}} \in \mathbb{R}^{T^{\text{for}} \times D}$, as illustrated in Figure 2.

2.3 Yet another ICU benchmark (YAIB)

Yet another ICU benchmark (YAIB) provided by van de Water et al. [22] is a modular, end-to-end framework supporting transparent benchmarking for clinical machine learning on ICU data (Appendix A.2). YAIB supports several publicly available EHR datasets and enables cohort selection, harmonization, and a variety of supervised learning tasks, with several implemented machine learning and deep learning models. This work has extended the YAIB framework to contain a self-supervised learning module, detailed in Appendix A.3, as well as the previously described BAT architecture. Building on YAIB's emphasis on transparent and reproducible benchmarking, we use the framework to evaluate our self-supervised models in a controlled setting.

3 Experiments

All experiments were performed using the Medical Information Mart for Intensive Care Database versions III and IV (MIMIC-III & IV, [31, 32]) and The eICU Collaborative Research Database (eICU [33]). Further information about the three datasets and their preprocessing can be found in Appendix A.4. A t-SNE [34] analysis of the preprocessed and harmonized datasets was performed, and Figure 4 indicates that they occupy a similar overall distribution. Despite the datasets overlapping in data distributions, prior work has shown that models trained on one dataset often fail to transfer effectively to others during inference [21, 35].

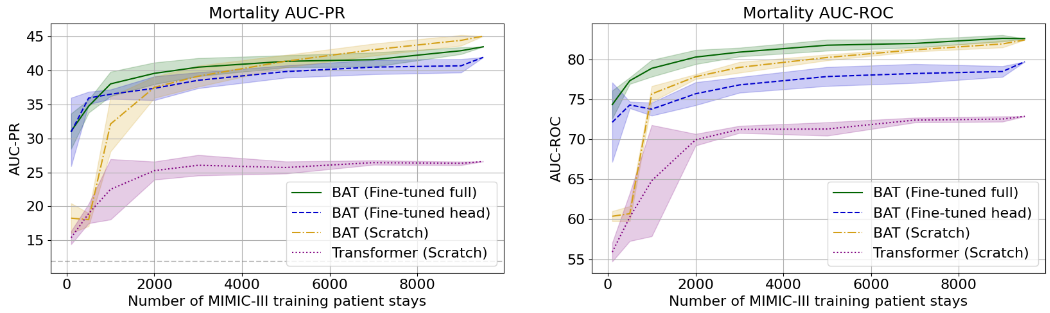
3.1 Baseline comparison

BAT was pre-trained with self-supervised forecasting on a pooled version of the datasets, with one held-out for fine-tuning. Pre-training followed the data splitting strategy described in Appendix A.6, and details on the models can be found in Appendix A.7. The pre-trained models were evaluated on the ability to be fine-tuned and generalize to the held-out dataset for mortality prediction. To test model robustness, fine-tuning was performed on varying subsets of the held-out data, with training set sizes ranging from 100 to 9,506 samples.

Figure 1 illustrates the results of pre-training on the pooled dataset consisting of eICU & MIMIC-IV and fine-tuning on MIMIC-III. The mean and standard deviations of AUC-PR and AUC-ROC over all training set sizes reflects five random subsamples of the training sets with preserved class imbalance. The pre-trained model was compared to two baseline models trained from scratch in a supervised manner: one using the same architecture (BAT) and another based on a vanilla Transformer that does not natively handle missing values and instead relies on mean-imputed data. The AUC-ROC performances, and partially the AUC-PR performance, revealed that the pre-trained model outperformed the baseline models on all training set sizes, but most significantly on dataset sizes < 5000 samples. Furthermore, fine-tuning only the binary classification head of the pre-trained model yields lower but overall comparable performance when compared to full-model fine-tuning. Similar results were seen when MIMIC-IV and eICU were held out (Table 1). AUC-ROC and AUC-PR results for all held-out datasets on all dataset sizes are provided in Appendix A.8. When evaluating the performance of the full and head-only fine-tuned variants in a joint manner, the pre-trained models were outperformed by the from-scratch models in a limited number of cases. Specifically, these exceptions occurred for select AUC-PR values: held-out MIMIC-III at dataset sizes 5,000–9,506 and held-out eICU at dataset size 7,000 (Table16). In all other cases, including all AUC-ROC values, the pre-trained models outperformed the from-scratch models.

4 Discussion

We evaluate BAT as a model architecture for a critical care time series foundation model and find that it outperforms initial supervised baselines on held-out datasets distinct from the training sources for mortality prediction, with training set sizes up to $\sim$10,000 samples. Fine-tuning only the binary



(a) **(b)**

Figure 1: Performance of fine-tuned model (pre-trained on eICU + MIMIC-IV) and supervised models trained from scratch on MIMIC-III. **(a)** AUC-PR (grey dashed line indicates the positive class prevalence) and **(b)** AUC-ROC across training set sizes ranging from 100 to 9,506 samples. Performance values reflect the mean ± standard deviation over five random subsamples of the training set (with preserved class imbalance). Models include BAT full & head fine-tuning, and models trained from scratch: BAT and Transformer.

Table 1: Average model performance, (AUC-PR ± sd) across multiple dataset sizes from MIMIC-III, MIMIC-IV, and eICU. Performance values reflect the mean ± standard deviation over five random subsamples of the training set (with preserved class imbalance). The pre-trained BAT models are fine-tuned and the baseline models are trained from scratch on the subsets. Highest performance for each dataset size is in bold, second highest is underlined.

Fine-tuning/ Training dataset	Dataset size	BAT (Fine-tuned full)	BAT (Fine-tuned head)	BAT (Scratch)	Transformer (Scratch)
MIMIC-III	500	34.74 ± 0.96	**35.94 ± 0.92**	18.05 ± 0.96	18.95 ± 1.43
	3000	**40.48 ± 1.33**	38.55 ± 1.17	39.15 ± 1.54	26.05 ± 1.51
	9000	42.90 ± 0.50	40.68 ± 0.99	**44.42 ± 0.78**	26.33 ± 0.26
MIMIC-IV	500	28.69 ± 1.61	**29.28 ± 2.50**	24.88 ± 2.11	7.82 ± 2.18
	3000	**38.62 ± 0.89**	35.05 ± 0.68	32.73 ± 1.71	18.51 ± 1.34
	9000	**42.35 ± 1.06**	39.00 ± 0.80	39.60 ± 1.59	20.43 ± 0.80
eICU	500	25.09 ± 5.02	**26.82 ± 0.93**	18.28 ± 0.98	7.76 ± 2.99
	3000	29.68 ± 0.83	**30.91 ± 0.48**	29.32 ± 1.83	13.51 ± 1.23
	9000	**32.97 ± 0.66**	31.12 ± 0.71	32.30 ± 0.90	15.31 ± 1.15

classification head of the pre-trained model achieves performance comparable to full-model fine-tuning, suggesting that the learned embeddings are both informative and transferable to downstream tasks on unseen datasets.

The pattern of performance gains across the three foundation models is consistent. The largest improvements over from-scratch models are seen with the smallest training sets (100–1,000 samples), while differences become smaller as dataset size increases (∼10,000 samples). This suggests that pre-training is most useful in situations with limited labeled data, which is common in clinical practice, whereas from-scratch models may perform adequately when larger datasets are available. Together, these results highlight the value of such a model in clinical settings where labeled data and computational resources are limited. Overall, this work demonstrates the feasibility and benefits of training foundation models for critical care time series data within a transparent, reproducible framework [22], extended for self-supervised pre-training to enable transfer learning and more robust, generalizable models.

4.1 Limitations

Our experiments were limited to MIMIC-III, MIMIC-IV, and eICU. To reach a training set size that is competitive with other foundation models, future work will need to incorporate additional datasets, such as those presented in Burger et al. [21]. Compared with fields like natural language processing, the number and size of publicly available ICU and critical care datasets remain very limited. Expanding the training data may therefore also require incorporating time series datasets from other domains, such as weather [36] or electricity consumption [37]. This would allow us to assess whether exposure to broader time series distributions improves model performance, or if domain-specific data is necessary given the sparse and irregular nature of critical care records, as suggested by other healthcare foundation models [7].

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

[3] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI Technical Report*, 2018. URL https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL https://arxiv.org/abs/2010.11929.

[5] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021. URL https://arxiv.org/abs/2103.14030.

[6] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention, 2021. URL https://arxiv.org/abs/2012.12877.

[7] Wasif Khan, Seowung Leem, Kyle B See, Joshua K Wong, Shaoting Zhang, and Ruogu Fang. A comprehensive survey of foundation models in medicine. *Ieee Reviews in Biomedical Engineering*, PP(99):1–22, 2025. ISSN 19411189, 19373333. doi: 10.1109/RBME.2025.3531360.

[8] Chen Zeming, Romanou Angelika, Bonnet Antoine, Hernández-Cano Alejandro, AlKhamissi Badr, Matoba Kyle, Salvi Francesco, Pagliardini Matteo, Fan Simin, Köpf Andreas, Mohtashami Amirkeivan, Sallinen Alexandre, Swamy Vinitra, Sakhaeirad Alireza, Krawczuk Igor, Bayazit Deniz, Marmet Axel, Li Mi, Boillat-Blanco Noémie, Keitel Kristina, Elkin Javier A., Robert Blaise, Montariol Syrielle, Bressan Silvia, Chen D., Demers Vincent, Emery N., Glasson Nicolas, Mensah Paulina Boadiwaa, Miauton Alix, Roemer S., Siebert Johan N, Starvaggi Carl, Suttels Véronique, Tan Rainer, Taylor Richard A., Toit Jacques du, Hartley Mary-Anne, Jaggi Martin, and Bosselut Antoine. Meditron: Open medical foundation models adapted for clinical practice. *Research Square (research Square)*, 2024. doi: 10.21203/rs.3.rs-4139743/v1.

[9] Qianqian Xie, Qingyu Chen, Aokun Chen, Cheng Peng, Yan Hu, Fongci Lin, Xueqing Peng, Jimin Huang, Jeffrey Zhang, Vipina Keloth, Xinyu Zhou, Huan He, Lucila Ohno-Machado, Yonghui Wu, Hua Xu, and Jiang Bian. Me-llama: Foundation large language models for medical applications. *Research Square*, 2024. ISSN 26935015. doi: 10.21203/rs.3.rs-4240043/v1.

[10] Jun Ma, Yuting He, Feifei Li, Lin Han, Chenyu You, and Bo Wang. Segment anything in medical images. *Nature Communications*, 15(1):654, 2024. ISSN 20411723. doi: 10.1038/s41467-024-44824-z.

[11] Joseph Cox, Peng Liu, Skylar E Stolte, Yunchao Yang, Kang Liu, Kyle B See, Huiwen Ju, and Ruogu Fang. Brainsegfounder: Towards 3d foundation models for neuroimage segmentation. *Arxiv*, 2024. ISSN 23318422.

[12] Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana Davuluri, and Han Liu. Dnabert-2: Efficient foundation model and benchmark for multi-species genome. 2024.

[13] Albi Celaj, Alice Jiexin Gao, Tammy T.Y. Lau, Erle M. Holgersen, Alston Lo, Varun Lodaya, Christopher B. Cole, and [Missing Name] Denroche. An rna foundation model enables discovery of disease mechanisms and candidate therapeutics. *bioRxiv*, 2023. doi: 10.1101/2023.09.20.558508. URL https://www.biorxiv.org/content/early/2023/09/26/2023.09.20.558508.

[14] Ruipu Luo, Ziwang Zhao, Min Yang, Junwei Dong, Da Li, Pengcheng Lu, Tao Wang, Linmei Hu, Minghui Qiu, and Zhongyu Wei. Valley: Video assistant with large language model enhanced ability. 2023.

[15] Zhao Wang, Chang Liu, Shaoting Zhang, and Qi Dou. Foundation model for endoscopy video analysis via large-scale self-supervised pre-train. *Medical Image Computing and Computer Assisted Intervention – Miccai 2023*, pages 101–111, 2023. doi: 10.1007/978-3-031-43996-4_10.

[16] Sheng Zhang, Yanbo Xu, Naoto Usuyama, Hanwen Xu, Jaspreet Bagga, Robert Tinn, Sam Preston, Rajesh Rao, Mu Wei, Naveen Valluri, Cliff Wong, Andrea Tupini, Yu Wang, Matt Mazzola, Swadheen Shukla, Lars Liden, Jianfeng Gao, Angela Crabtree, Brian Piening, Carlo Bifulco, Matthew P. Lungren, Tristan Naumann, Sheng Wang, and Hoifung Poon. Biomedclip: a multimodal biomedical foundation model pretrained from fifteen million scientific image-text pairs. 2025.

[17] Kun Yuan, Vinkle Srivastav, Tong Yu, Joël L Lavanchy, Jacques Marescaux, Pietro Mascagni, Nassir Navab, and Nicolas Padoy. Learning multi-modal representations by watching hundreds of surgical video lectures. *Medical Image Analysis*, 105:103644, 2025. ISSN 13618423, 13618415. doi: 10.1016/j.media.2025.103644.

[18] Michael Wornow, Ethan Steinberg, Rahul Thapa, Jason A. Fries, and Nigam H. Shah. Ehrshot: An ehr benchmark for few-shot evaluation of foundation models. *Advances in Neural Information Processing Systems*, 36, 2023. ISSN 10495258.

[19] Ethan Steinberg, Jason Fries, Yizhe Xu, and Nigam Shah. Motor: A time-to-event foundation model for structured medical records, 2023. URL https://arxiv.org/abs/2301.03150.

[20] Manuel Burger, Daphné Chopard, Malte Londschien, Fedor Sergeev, Hugo Yèche, Rita Kuznetsova, Martin Faltys, Eike Gerdes, Polina Leshetkina, Peter Bühlmann, and Gunnar Rätsch. A foundation model for intensive care: Unlocking generalization across tasks and domains at scale. *medRxiv*, 2025. doi: 10.1101/2025.07.25.25331635. URL https://www.medrxiv.org/content/early/2025/07/25/2025.07.25.25331635.

[21] Manuel Burger, Fedor Sergeev, Malte Londschien, Daphné Chopard, Hugo Yèche, Eike Gerdes, Polina Leshetkina, Alexander Morgenroth, Zeynep Babür, Jasmina Bogojeska, Martin Faltys, Rita Kuznetsova, and Gunnar Rätsch. Towards foundation models for critical care time series, 2024. URL https://arxiv.org/abs/2411.16346.

[22] Robin van de Water, Hendrik Schmidt, Paul Elbers, Patrick Thoral, Bert Arnrich, and Patrick Rockenschaub. Yet another icu benchmark: A flexible multi-center framework for clinical ml. *12th International Conference on Learning Representations, Iclr 2024*, 2024.

[23] Max Horn, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. Set functions for time series. *37th International Conference on Machine Learning, Icml 2020*, 168147-6: 4303–4313, 2020.

[24] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, 8(1):6085, 2018. ISSN 20452322. doi: 10.1038/s41598-018-24271-9.

[25] Yicheng Luo, Bowen Zhang, Zhen Liu, and Qianli Ma. Hi-patch: Hierarchical patch GNN for irregular multivariate time series modeling, 2025. URL https://openreview.net/forum?id=OGtUfA6Amo.

[26] Hrayr Harutyunyan, Hrant Khachatrian, David C. Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific Data*, 6(1):96, 2019. ISSN 20524463. doi: 10.1038/s41597-019-0103-9.

[27] Hugo Yèche, Rita Kuznetsova, Marc Zimmermann, Matthias Hüser, Xinrui Lyu, Martin Faltys, and Gunnar Rätsch. Hirid-icu-benchmark – a comprehensive machine learning benchmark on high-resolution icu data, 2022. URL https://arxiv.org/abs/2111.08536.

[28] Adibvafa Fallahpour, Mahshid Alinoori, Wenqian Ye, Xu Cao, Arash Afkanpour, and Amrit Krishnan. Ehrmamba: Towards generalizable and scalable foundation models for electronic health records. *Proceedings of Machine Learning Research*, 259:291–307, 2024. ISSN 26403498.

[29] Rachael DeVries, Casper Christensen, Marie Lisandra Zepeda Mendoza, and Ole Winther. Bi-axial transformers: Addressing the increasing complexity of ehr classification, 2025. URL https://arxiv.org/abs/2508.12418.

[30] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*, 2019.

[31] Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3(1):160035, 2016. ISSN 20524463. doi: 10.1038/sdata.2016.35.

[32] Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Benjamin Moody, Brian Gow, Li-wei H. Lehman, Leo A. Celi, and Roger G. Mark. Mimic-iv, a freely accessible electronic health record dataset. *Scientific Data*, 10(1):1, 2023. ISSN 20524463. doi: 10.1038/s41597-022-01899-x.

[33] Tom J. Pollard, Alistair E. W. Johnson, Jesse D. Raffa, Leo A. Celi, Roger G. Mark, and Omar Badawi. The eicu collaborative research database, a freely available multi-center database for critical care research. *Scientific Data*, 5(1):180178, 2018. ISSN 20524463. doi: 10.1038/sdata.2018.178.

[34] Laurens Van Der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2625, 2008. ISSN 15337928, 15324435.

[35] Patrick Rockenschaub, Adam Hilbert, Tabea Kossen, Paul Elbers, Falk Von Dincklage, Vince Istvan Madai, and Dietmar Frey. The impact of multi-institution datasets on the generalizability of machine learning prediction models in the icu. *Critical Care Medicine*, 52(11):1710–1721, 2024. ISSN 15300293, 00903493. doi: 10.1097/CCM.0000000000006359.

[36] Weather data from the max planck institute for biogeochemistry. URL https://www.bgc-jena.mpg.de/wetter/.

[37] Artur Trindade. Electricity load diagrams, 2015. URL https://doi.org/10.24432/C58C86.

[38] R. van de Water. YAIB-cohorts, 2024. URL https://github.com/rvandewater/YAIB-cohorts. GitHub repository.

[39] Robin van de Water, Hendrik Schmidt, and Patrick Rockenschaub. Yet Another ICU Benchmark (YAIB). URL https://github.com/rvandewater/YAIB.

[40] Nicolas Bennett, Drago Plečko, Ida Fong Ukor, Nicolai Meinshausen, and Peter Bühlmann. ricu: R's interface to intensive care data. *Gigascience*, 12:giad041, 2023. ISSN 2047217x. doi: 10.1093/gigascience/giad041.

[41] Sindhu Tipirneni and Chandan K. Reddy. Self-supervised transformer for sparse and irregularly sampled multivariate clinical time-series. *Acm Transactions on Knowledge Discovery From Data*, 16(6):105, 2022. ISSN 1556472x, 15564681. doi: 10.1145/3516367.

[42] Hrishikesh Patel, Ruihong Qiu, Adam Irwin, Shazia Sadiq, and Sen Wang. Emit- event-based masked auto encoding for irregular time series. *Arxiv (cornell University)*, 2024. doi: 10.48550/arXiv.2409.16554.

[43] Alex Labach, Aslesha Pokhrel, Xiao Shi Huang, Saba Zuberi, Seung Eun Yi, Maksims Volkovs, Tomi Poutanen, and Rahul G. Krishnan. Duett: Dual event time transformer for electronic health records. 2023.

[44] Alistair Johnson, Tom Pollard, and Roger Mark. MIMIC-III Clinical Database Care-Vue subset (version 1.4). `https://doi.org/10.13026/8a4q-w170`, 2022. PhysioNet. RRID:SCR_007345.

A Appendix

A.1 BAT architecture

One Transformer-based model designed for irregular multivariate time series data is Bi-Axial Transformer (BAT) introduced in the work of DeVries et al. [29]. Unlike classical Transformers that use an encoder-decoder structure for sequence generation, BAT relies solely on encoder-style self-attention to model dependencies across both the temporal and feature/sensor dimensions. This axial attention structure allows BAT to capture rich interactions within and across time steps and modalities. To address common challenges in irregular multivariate time series data, such as missing values and heterogeneous input types, BAT incorporates missingness indicators directly into its input embeddings. Figure 2 (Adapted from Figure 2 in DeVries et al. [29]) illustrates the overall architecture of BAT. The input to the model is a multivariate time series of shape $D \times T$, where D is the number of features and T is the number of time steps. Each input is embedded using a combination of the observed value (including missingness), a learned feature identity embedding, and a continuous time-based positional encoding. This results in an embedded tensor of shape $D \times T \times E$, where E is the embedding dimension. The embedded input is then processed by bi-axial attention layers, which apply self-attention separately along the time and feature dimensions. The output from the attention layers is subsequently pooled and concatenated with the static features, P, which include non-time-varying demographic variables such as age and sex. The BAT model was adapted in this work to have two different prediction heads: one for a supervised learning setup via binary classification, predicting $\hat{y} \in \{0, 1\}$, and another for a self-supervised learning setup via forecasting, predicting $\hat{\mathbf{X}}^{\text{for}} \in \mathbb{R}^{T^{\text{for}} \times D}$.

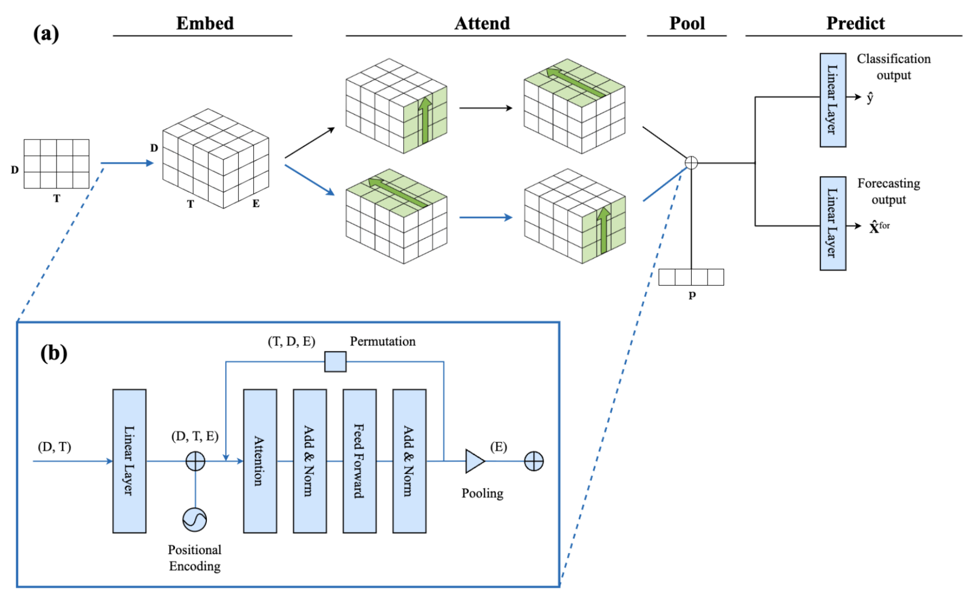


Figure 2: Overview of the Bi-Axial Transformer (BAT) architecture. (a) Shows the full model architecture and data representation, and (b) Shows an attention track indicated by the blue arrows. D is the number of time-varying features, T the number of time steps, E the embedding size, and P the static features. The model supports both binary classification and forecasting via separate prediction heads. This figure is adapted from Figure 2 in DeVries et al. [29].

A.2 Yet another ICU benchmark framework

The Yet Another ICU Benchmark (YAIB) framework [22] provides a modular, end-to-end solution for clinical machine learning on ICU data, explicitly designed to address key limitations regarding reproducibility in the field. An illustration of the framework is shown in Figure 3 (Inspired by Figure 1 in van de Water et al. [22]). It consists of two repositories: `YAIB-cohorts` [38], which handles dataset harmonization and cohort construction, and the main `YAIB` repository [39], which manages model training and evaluation. The `YAIB-cohorts` repository builds on the open-source `R` package `ricu` [40] to harmonize multiple ICU datasets using a unified, concept-based abstraction of clinical variables. It supports five publicly available EHR datasets (MIMIC-III, MIMIC-IV, eICU, HiRID, AUMCdb) and maps their contents into a common structure with consistent semantic definitions and temporal alignment, and supports integration of new datasets. This harmonization enables standardized cohort construction, label definition, and data extraction across datasets, facilitating multi-center analyses and reproducible experimental setups. The main `YAIB` repository then provides the downstream machine learning pipeline for supervised modeling, including pre-processing, feature extraction, and evaluation. We entended the framework to support self-supervised pre-training, described in detail in Appendix A.3, as well as add the BAT architecture to the selection of models. The models in this work show lower performance on the mortality prediction task compared to similar studies [41, 42, 43], particularly in class sensitive metric, AUC-PR. As van de Water et al. [22] notes, variations in preprocessing pipelines, task and cohort definitions etc. and limited transparency hinder reproducibility, especially when code is unavailable. In contrast to most prior work that experiment with performance-boosting strategies, e.g. upsampling of the minority class, our main focus of this work has been on demonstrating the transfer learning potential of robust self-supervised models rather than surpassing state-of-the-art results. We used a weighted loss available in the YAIB framework to handle the high class imbalance, but did not investigate further performance-boosting strategies. This work benchmarks pre-trained models against two baselines, aiming to provide a comprehensive and fair comparison throughout the entire machine learning pipeline.

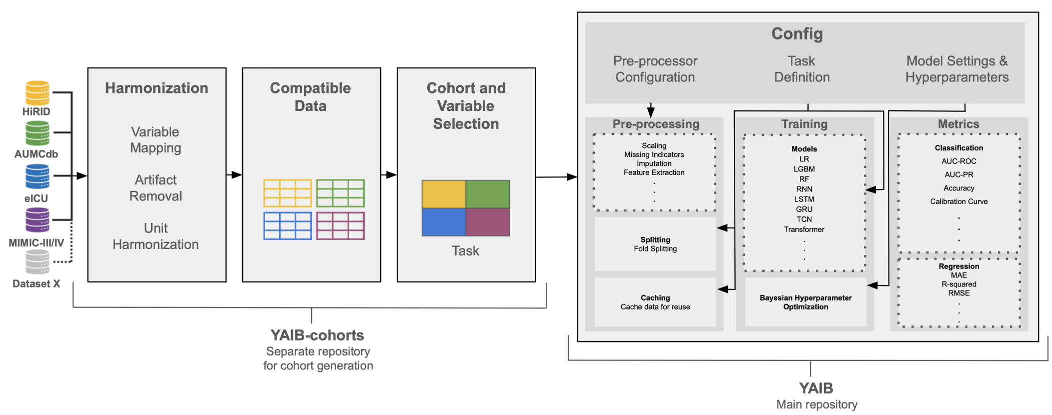


Figure 3: Overview of the Yet Another ICU Benchmark (YAIB) pipeline. The left side illustrates the creation of harmonized ICU cohorts, implemented in a separate repository, `YAIB-cohorts` [38]. The right side represents the machine learning component of the pipeline, contained in the main `YAIB` repository [39], which covers preprocessing, model training, and evaluation. Dotted-line components indicate extensible modules that follow a standardized interface. This figure is inspired from Figure 1 in van de Water et al. [22].

A.3 Implementation of self-supervised learning

The self-supervised learning objective is implemented as a forecasting task, where future values are predicted based on past values. The approach in this work is inspired by the dynamic sampling method proposed by Tipirneni and Reddy [41], in which observation and forecasting windows are dynamically constructed during batch loading. A set of constraints is introduced to govern the selection of observation and forecasting windows. These constraints are designed to ensure both sufficient historical context and generating a valid forecasting window, thereby improving the overall quality and consistency of the training data. The overall goal is to select a valid time index where the observation and forecasting window are split. This work performs the search for the valid time index in a batch-wise manner. For each batch element, a patient is randomly sampled and candidate time indices are identified. These indices are filtered based on the following constraints:

1. **Sparsity check:** The selected index must correspond to a time point where there is at least one observed value in the observation window.
2. **Minimum Observation Length:** The index must be at least L time steps into the time series to ensure sufficient historical context. For this work, $L = 12$ hours.
3. **Forecasting Window Availability:** The index must allow room for a complete forecasting window of length H. For this work, $H = 2$ hours.

If valid time indices are found, one is randomly selected as the split point between the observation window and the forecasting window. If no valid index is found, a new patient is sampled and the process is repeated. This approach ensures that the model is trained on different points within each hospital stay, with varying observation window lengths, effectively exposing it to diverse temporal contexts and clinical information.The sampling method are defined in Algorithm 1.

Algorithm 1 Dynamic sampling of observation and forecasting window during batch loading

```
function CALL(batch)
    (data, mask) ← LOADBATCH(batch)
    (B, C, T) ← SHAPE(data)
    t_1 ← None, tries ← 0, max_tries ← B
    while t_1 = None and tries < max_tries do
        i ← RANDOMINTEGER(0, B − 1)
        valid_times ← {t | mask[i, t] = True}
        valid_times ← {t ∈ valid_times | t ≥ L}
        if valid_times = ∅ then
            tries ← tries + 1
            continue
        end if
        max_index ← max(valid_times)
        valid_times ← {t ∈ valid_times | t ≤ max_index − H}
        if valid_times = ∅ then
            tries ← tries + 1
            continue
        end if
        t_1 ← RANDOMCHOICE(valid_times)
    end while
    if t_1 = None then
        raise Error("No valid index found in batch")
    end if
    t_0 ← max(0, t_1 − max_obs)
    t_2 ← t_1 + forecast_horizon
    X_obs ← data[:, :, t_0 : t_1]
    M_obs ← mask[:, :, t_0 : t_1]
    X_for ← data[:, :, t_1 : t_2]
    M_for ← mask[:, :, t_1 : t_2]
    return (X_obs, M_obs, X_for, M_for)
end function
```

A.4 Datasets, cohort definitions and preprocessing

The three publicly available ICU datasets used in this work are MIMIC-III [31], MIMIC-IV [32], and the eICU [33]. These datasets contain structured clinical data from ICU stays and are widely used in the development of machine learning models for critical care. Table 2 summarizes key statistics for the three datasets. As this study involves pooling of the datasets into one larger dataset, it is important to avoid patient overlap between them, as this could lead to data leakage in downstream experiments. Therefore, a filtered subset of the original MIMIC-III dataset that excludes all patients also found in MIMIC-IV is used. The overlap exists because MIMIC-IV was developed to extend MIMIC-III by including more recent and higher-resolution data. The subset of the original MIMIC-III dataset used in this work is called MIMIC-III Clinical Database CareVue subset [44], which limits MIMIC-III to records from 2001–2008 thereby excluding patient stays in the overlapping time period. Furthermore, a second filter was applied to remove any individual who also appears in MIMIC-IV, as some patients had an earlier stay recorded in MIMIC-III and a later hospital stay captured in MIMIC-IV. Feature selection was performed by van de Water et al. [22] based on availability across all benchmarked datasets available in YAIB [39], with an emphasis on consistency to support cross-dataset experiments. A total of 52 clinical features were used as input for model development, consisting of 4 static and 48 time-varying variables. A full list of the features used and their units are provided in Table 3.

Table 2: Dataset statistics for MIMIC-III, MIMIC-IV, and eICU. BIDMC = Beth Israel Deaconess Medical Center. *The subset of the MIMIC-III data used in this work is the MIMIC-III CareVue subset [44]; values in parentheses represent statistics from the full MIMIC-III dataset.

Dataset	**Admissions**	**Collection period**	**Origin**	**Hospital**	**Mortality (Positive class)**
MIMIC-III*	27k* (40k)	2001–2008* (2001–2012)	U.S.	BIDMC	11.9%
MIMIC-IV	73k	2008–2019	U.S.	BIDMC	7.3%
eICU	200k	2014–2015	U.S.	208 hospitals across the U.S.	5.5%

Patient cohorts were constructed using the `YAIB-cohorts` repository [38]. Data were temporally aligned and resampled at one-hour resolution. Uniform exclusion criteria were applied across all datasets and tasks: **(1)** invalid ICU stay timing (e.g., negative length of stay), **(2)** ICU stay < 6 hours, **(3)** fewer than four valid time points, **(4)** measurement gaps > 12 hours, and **(5)** age < 18 years at ICU admission. These filters preceded task-specific cohort definitions. For mortality classification, the input window was the first 24 hours post-ICU admission; stays < 30 hours were excluded to prevent causal leakage [22]. The outcome label was mortality = 1 if the patient died during the same hospital admission. The YAIB framework requires task-specific definitions with inclusion/exclusion criteria and label generation, making it incompatible with self-supervised learning where labels are not used. Since YAIB cannot generate unlabeled cohorts, we selected the Length of Stay task, which imposed no additional exclusions beyond the five main criteria mentioned, thereby maximizing size of the pre-training dataset. Length of stay labels were still generated as a consequence of YAIB's supervised design but were simply discarded resulting in a unlabeled cohort generation. This unlabeled cohort was the patient cohort used for pre-training. This approach maintained full compatibility with the existing framework while avoiding the need for significant changes to its design.

Preprocessing was carried out using YAIB's main repository `YAIB` [39], applied after cohort extraction. This included the addition of missingness indicators, forward-fill imputation within each ICU stay, and mean imputation for values without prior observations, using statistics computed from the training set to avoid data leakage. All features were standardized to zero mean and unit variance based on training split statistics. Data were split into training, validation, and test sets using cross-validation folds defined during preprocessing, ensuring that preprocessing steps such as scaling and imputation were fitted only on the training data. All preprocessing decisions followed the default setup specified by the `base_classification_preprocessor` class in the main repository, `YAIB`.

Table 3: Clinical features and units

Feature	Unit
Static	
Age at hospital admission	years
Female sex	–
Patient height	cm
Patient weight	kg
Time-varying	
Albumin	g/dL
Alkaline phosphatase	IU/L
Alanine aminotransferase	IU/L
Aspartate aminotransferase	IU/L
Band form neutrophils	%
Base excess	mmol/L
Bicarbonate	mmol/L
Bilirubin (direct)	mg/dL
Bilirubin (total)	mg/dL
Blood pressure (diastolic)	mmHg
Blood pressure (systolic)	mmHg
Blood urea nitrogen	mg/dL
Calcium	mg/dL
Calcium ionized	mmol/L
Chloride	mmol/L
CO_2 partial pressure	mmHg
C-reactive protein	mg/L
Creatinine	mg/dL
Creatine kinase	IU/L
Creatine kinase MB	ng/mL
Fibrinogen	mg/dL
Fraction of inspired oxygen	%
Glucose	mg/dL
Haemoglobin	g/dL
Heart rate	beats/minute
International normalised ratio (INR)	–
Lactate	mmol/L
Lymphocytes	%
Magnesium	mg/dL
Mean arterial pressure	mmHg
Mean cell haemoglobin	pg
Mean corpuscular haemoglobin concentration	%
Mean corpuscular volume	fL
Methaemoglobin	%
Neutrophils	%
O_2 partial pressure	mmHg
Oxygen saturation	%
Partial thromboplastin time	sec
pH of blood	–
Phosphate	mg/dL
Platelets	1,000/μL
Potassium	mmol/L
Respiratory rate	breaths/minute
Sodium	mmol/L
Temperature	$^\circ$C
Troponin T	ng/mL
Urine output	mL
White blood cells	1,000/μL

A.5 Data distributions

A t-SNE [34] analysis of the preprocessed and harmonized datasets was performed to asses the data distributions of the three datasets before modeling. The results shown in Figure 4. It reveals that the datasets appear to lie within the same general distribution, which might be a result of similar clinical practices and patient populations.

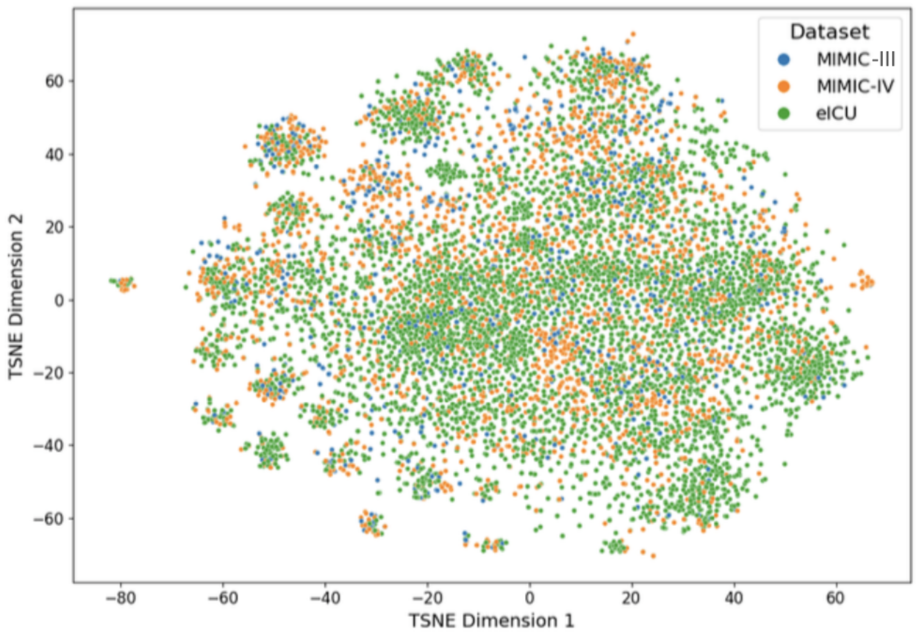


Figure 4: Two-dimensional t-SNE projection [34] of the harmonized and preprocessed datasets used in this study: MIMIC-III, MIMIC-IV, and eICU. Each point represents a time step of a patient stay.

A.6 Hyperparameters and data splits

Pre-training of the multi-dataset models was performed by splitting each dataset into an 80/20 train–test split, and then further splitting the training portion 80/20 to create a validation set. The baseline models in the fine-tuning experiment used the same data splits as the pre-trained models.

This appendix provides the optimal hyperparameters used across all experiments. Hyperparameter tuning was performed by randomly selecting hyperparameters, in selected ranges, for 25 models and choosing the configuration that either resulted in the lowest test loss (for pre-training) or the highest test AUC-PR (for supervised training). During fine-tuning, the learning rates were further tuned using a grid search over 10 learning rates, while the other hyperparameters were kept the same as those used during pre-training.

Table 4: Final hyperparameters for pre-training of BAT on pooled datasets MIMIC-III + MIMIC-IV.

Component	Hyperparameters
Model	attn_dropout = 0.4, dropout = 0.1, heads = 4, layers = 12, pooling = max, use_mask = False, value_embed_size = 128
Trainer	batch_size = 32, epochs = 200, patience = 50, min_delta = 1e-4
Optimizer	lr = 1.955e-5, weight_decay = 2.703e-6
Forecasting	forecast_horizon = 2, sensors_count = 48

Table 5: Final hyperparameters for pre-training of BAT on pooled datasets eICU + MIMIC-III.

Component	Hyperparameters
Model	attn_dropout = 0.1, dropout = 0.1, heads = 2, layers = 6, pooling = max, use_mask = False, value_embed_size = 128
Trainer	batch_size = 64, epochs = 200, patience = 50, min_delta = 1e-4
Optimizer	lr = 8.834e-5, weight_decay = 9.254e-6
Forecasting	forecast_horizon = 2, sensors_count = 48

Table 6: Final hyperparameters for pre-training of BAT on pooled datasets eICU + MIMIC-IV.

Component	Hyperparameters
Model	attn_dropout = 0.1, dropout = 0.1, heads = 4, layers = 6, pooling = max, use_mask = False, value_embed_size = 128
Trainer	batch_size = 64, epochs = 200, patience = 50, min_delta = 1e-4
Optimizer	lr = 2.496e-5, weight_decay = 7.567e-6
Forecasting	forecast_horizon = 2, sensors_count = 48

Table 7: Final hyperparameters for fine-tuning BAT on the datasets MIMIC-III, MIMIC-IV, and eICU. All other hyperparameters are kept the same as during pre-training and a lr_scheduler = ExponentialLR(gamma = 0.95) is added.

Dataset	Dataset size	Learning rate (Fine-tune full)	Learning rate (Fine-tune head)
MIMIC-III	100–1000	1e-4	5e-2
	2000–9506	5e-5	1e-2
MIMIC-IV	100–1000	6.55e-4	7e-2
	2000–9506	5e-5	7e-3
eICU	100–1000	1.15e-4	6e-2
	2000–9506	7e-5	7e-3

Table 8: Final hyperparameters for supervised training of BAT on MIMIC-III.

Component	Hyperparameters
Model	attn_dropout = 0.4, dropout = 0.2, heads = 2, layers = 2, pooling = max, use_mask = False, value_embed_size = 128
Trainer	batch_size = 256, epochs = 200, patience = 10, min_delta = 1e-4
Optimizer	lr = 3.551e-5, weight_decay = 3.024e-5

Table 9: Final hyperparameters for supervised training of BAT on MIMIC-IV.

Component	Hyperparameters
Model	attn_dropout = 0.1, dropout = 0.2, heads = 2, layers = 12, pooling = max, use_mask = False, value_embed_size = 128
Trainer	batch_size = 64, epochs = 200, patience = 10, min_delta = 1e-4
Optimizer	lr = 8.377e-6, weight_decay = 9.013e-6

Table 10: Final hyperparameters for supervised training of BAT on eICU.

Component	Hyperparameters
Model	attn_dropout = 0.4, dropout = 0.1, heads = 1, layers = 6, pooling = max, use_mask = False, value_embed_size = 64
Trainer	batch_size = 64, epochs = 200, patience = 10, min_delta = 1e-4
Optimizer	lr = 5.093e-5, weight_decay = 5.115e-5

Table 11: Final hyperparameters for supervised training of Transformer on MIMIC-III.

Component	Hyperparameters
Model	dropout = 0.1, dropout_att = 0.1, heads = 3, depth = 5, hidden = 512, ff_hidden_mult = 2, l1_reg = 2.864e-5
Trainer	batch_size = 256, epochs = 200, patience = 10, min_delta = 1e-4
Optimizer	lr = 3.768e-5, weight_decay = 8.619e-6

Table 12: Final hyperparameters for supervised training of Transformer on MIMIC-IV.

Component	Hyperparameters
Model	dropout = 0.1, dropout_att = 0.1, heads = 3, depth = 20, hidden = 128, ff_hidden_mult = 1, l1_reg = 4.160e-5
Trainer	batch_size = 64, epochs = 200, patience = 10, min_delta = 1e-4
Optimizer	lr = 1.513e-5, weight_decay = 6.356e-5

Table 13: Final hyperparameters for supervised training of Transformer on eICU.

Component	Hyperparameters
Model	dropout = 0.1, dropout_att = 0.2, heads = 1, depth = 20, hidden = 512, ff_hidden_mult = 4, l1_reg = 7.080e-5
Trainer	batch_size = 128, epochs = 200, patience = 10, min_delta = 1e-4
Optimizer	lr = 1.792e-5, weight_decay = 5.929e-5

A.7 Models and code availability

All code related to this project is available at: `https://github.com/Katja-Jagd/YAIB`. This repository was originally forked from `https://github.com/rvandewater/YAIB` [39] and has been expanded to support the experiments and methods presented in this work.

Table 14: Pre-trained models on combinations of the three datasets of this work (MIMIC-III, MIMIC-IV, and eICU). Details on the models, including training and computational resources, are summarized.

	Pre-trained Models		
Pooled pre-training datasets	MIMIC-IV + eICU	MIMIC-III + eICU	MIMIC-III + MIMIC-IV
Fine-tuning dataset	MIMIC-III	MIMIC-IV	eICU
#Params	3.8M	2.9M	7.4M
Pre-training dataset size	273k	227k	100k
Pre-training positive class	6.0%	6.26%	8.54%
Fine-tuning positive class	11.9%	7.3%	5.5%
Pre-training time	$\sim$ 40h	$\sim$ 25h	$\sim$ 30h
Pre-training GPU	1 x A100 (40 GB)	1 x A100 (40 GB)	1 x A100 (40 GB)
Fine-tuning time	$\leq$ 20 min	$\leq$ 20 min	$\leq$ 20 min
Fine-tuning GPU	1 x V100 (16 GB)	1 x V100 (16 GB)	1 x V100 (16 GB)

A.8 Fine-tuning results

Table 15: Model performance, AUC-ROC, across dataset size ranging from 100 to 9,506 on MIMIC-III, MIMIC-IV, and eICU. Performance values reflect the mean ± standard deviation over five random subsamples of the training set (with preserved class imbalance). The pre-trained BAT models are fine-tuned and the baseline models are trained from scratch on the subsets. Highest performance for each dataset size is indicated in bold, and second highest is underlined.

Fine-tuning/ Training dataset	Dataset size	BAT (Fine-tuned full)	BAT (Fine-tuned head)	BAT (Scratch)	Transformer (Scratch)
MIMIC-III	100	**74.38 ± 1.76**	72.17 ± 4.98	60.38 ± 0.64	55.89 ± 1.18
	500	**77.38 ± 0.41**	74.34 ± 0.41	60.70 ± 0.94	60.28 ± 3.01
	1000	**78.93 ± 1.05**	73.81 ± 0.81	75.72 ± 0.98	64.84 ± 6.97
	2000	**80.34 ± 0.89**	75.74 ± 1.44	77.88 ± 0.30	69.97 ± 0.73
	3000	**80.96 ± 0.55**	76.85 ± 0.98	79.05 ± 0.67	71.27 ± 0.44
	5000	**81.83 ± 0.69**	77.90 ± 1.19	80.29 ± 0.36	71.32 ± 0.84
	7000	**82.05 ± 0.49**	78.28 ± 1.19	81.25 ± 0.22	72.44 ± 0.30
	9000	**82.69 ± 0.41**	78.53 ± 0.67	81.99 ± 0.44	72.57 ± 0.33
	9560	**82.66 ± 0.00**	79.76 ± 0.00	82.49 ± 0.00	72.91 ± 0.00
MIMIC-IV	100	63.30 ± 5.55	**71.18 ± 3.27**	66.77 ± 3.04	50.10 ± 2.55
	500	**79.22 ± 0.80**	76.58 ± 0.93	76.09 ± 1.59	48.15 ± 7.26
	1000	**82.49 ± 0.46**	78.59 ± 0.56	78.15 ± 1.83	62.67 ± 10.44
	2000	**84.38 ± 0.57**	80.54 ± 0.97	79.15 ± 1.68	71.31 ± 1.22
	3000	**85.11 ± 0.63**	81.45 ± 0.82	81.30 ± 1.46	72.62 ± 1.50
	5000	**85.94 ± 0.46**	84.01 ± 0.77	84.00 ± 0.60	74.17 ± 0.52
	7000	**86.62 ± 0.32**	85.03 ± 0.31	84.33 ± 0.57	74.61 ± 0.71
	9000	**86.93 ± 0.32**	85.54 ± 0.62	84.59 ± 0.67	74.88 ± 0.81
	9506	**86.97 ± 0.25**	85.43 ± 0.46	84.91 ± 0.70	74.89 ± 0.54
eICU	100	60.86 ± 12.39	**73.44 ± 2.89**	69.26 ± 2.36	50.03 ± 9.42
	500	**76.96 ± 4.41**	75.27 ± 1.27	73.85 ± 1.72	54.15 ± 11.53
	1000	**79.18 ± 1.13**	77.88 ± 2.06	76.91 ± 1.42	60.82 ± 9.10
	2000	**80.74 ± 0.41**	80.59 ± 0.69	78.41 ± 0.99	69.03 ± 1.23
	3000	81.25 ± 0.90	**81.46 ± 0.85**	79.22 ± 0.76	69.23 ± 1.49
	5000	81.95 ± 0.35	**82.21 ± 0.84**	79.84 ± 0.34	70.32 ± 1.19
	7000	82.73 ± 0.50	**83.06 ± 0.58**	79.89 ± 1.03	70.81 ± 0.77
	9000	**83.86 ± 0.38**	83.63 ± 0.49	79.51 ± 0.95	71.35 ± 0.98
	9506	83.40 ± 0.67	**83.64 ± 0.43**	80.45 ± 0.98	71.18 ± 0.92

Table 16: Model performance, AUC-PR, across dataset size ranging from 100 to 9,506 on MIMIC-III, MIMIC-IV, and eICU. Performance values reflect the mean ± standard deviation over five random subsamples of the training set (with preserved class imbalance). The pre-trained BAT models are fine-tuned and the baseline models are trained from scratch on the subsets. Highest performance for each dataset size is indicated in bold, and second highest is underlined.

Fine-tuning/ Training dataset	Dataset size	BAT (Fine-tuned full)	BAT (Fine-tuned head)	BAT (Scratch)	Transformer (Scratch)
hline	100	**31.07 ± 2.59**	30.95 ± 5.03	18.27 ± 2.19	15.39 ± 0.96
	500	34.74 ± 0.96	**35.94 ± 0.92**	18.05 ± 0.96	18.95 ± 1.43
	1000	**38.02 ± 1.79**	36.50 ± 0.69	32.12 ± 4.04	22.51 ± 4.44
	2000	**39.57 ± 1.59**	37.51 ± 1.55	37.37 ± 1.75	25.26 ± 1.33
MIMIC-III	3000	**40.48 ± 1.33**	39.15 ± 1.54	38.55 ± 1.17	26.05 ± 1.51
	5000	41.34 ± 0.88	39.83 ± 0.90	**41.36 ± 0.91**	25.72 ± 0.89
	7000	41.58 ± 1.05	40.49 ± 1.05	**43.05 ± 0.88**	26.44 ± 0.32
	9000	42.90 ± 0.50	40.68 ± 0.99	**44.42 ± 0.78**	26.33 ± 0.26
	9506	43.48 ± 0.00	41.93 ± 0.00	**45.05 ± 0.00**	26.61 ± 0.00
	100	13.42 ± 2.77	**23.58 ± 4.96**	16.58 ± 2.20	8.73 ± 1.01
	500	28.69 ± 1.61	**29.28 ± 2.50**	24.88 ± 2.11	7.82 ± 2.18
	1000	**33.26 ± 2.02**	31.43 ± 2.22	27.97 ± 0.70	13.17 ± 3.66
	2000	**37.56 ± 0.45**	33.03 ± 0.44	31.22 ± 1.35	17.36 ± 1.12
MIMIC-IV	3000	**38.62 ± 0.89**	35.05 ± 0.68	32.73 ± 1.71	18.51 ± 1.34
	5000	**39.80 ± 1.04**	37.99 ± 1.16	35.74 ± 0.99	19.41 ± 0.56
	7000	**41.54 ± 0.72**	38.55 ± 1.02	38.50 ± 1.03	19.83 ± 1.23
	9000	**42.35 ± 1.06**	39.60 ± 1.59	39.00 ± 0.80	20.43 ± 0.80
	9506	**42.37 ± 0.84**	39.78 ± 0.89	39.53 ± 0.84	20.47 ± 0.59
	100	9.82 ± 3.09	**22.97 ± 4.58**	18.87 ± 1.95	7.57 ± 3.03
	500	25.09 ± 5.02	**26.82 ± 0.93**	18.28 ± 0.98	7.76 ± 2.99
	1000	**29.08 ± 1.37**	28.20 ± 1.50	24.29 ± 2.74	9.97 ± 3.10
	2000	29.04 ± 1.11	**29.98 ± 0.66**	28.11 ± 1.93	13.56 ± 0.96
eICU	3000	29.68 ± 0.83	**30.91 ± 0.48**	29.32 ± 1.83	13.51 ± 1.23
	5000	**31.34 ± 0.36**	31.21 ± 0.22	30.78 ± 1.31	14.78 ± 0.53
	7000	31.84 ± 0.63	31.22 ± 0.54	**32.17 ± 0.92**	14.88 ± 0.73
	9000	**32.97 ± 0.66**	31.12 ± 0.71	32.30 ± 0.90	15.31 ± 1.15
	9506	**32.55 ± 0.88**	31.85 ± 0.90	31.76 ± 0.66	15.09 ± 0.98