

TIME SERIES MACHINE LEARNING FOR HOSPITALIZED PATIENT MONITORING: ADDRESSING MISSING DATA, MISSING LABELS, AND HIGH FALSE ALARM RATES

A dissertation

submitted by

Preetish Rath

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

in

Computer Science

TUFTS UNIVERSITY

August 2024

© Copyright 2024 by Preetish Rath

Adviser: Professor Michael C. Hughes

Abstract

The advent of big data and machine learning (ML) has the potential to enhance the quality and timeliness of care for hospitalized patients. Through analyzing time series data of lab results and vital signs, machine learning systems are capable of delivering predictions of the imminent risks of in-hospital mortality or critical disease development. However, the efficacy of such models is hampered by three predominant challenges: (1) current systems often produce too many false alarms, which compromise the model’s reliability and acceptance into clinical practice; (2) most ML models struggle to handle the issue of missing data, which is caused by differences in measurement frequencies across channels or by decisions made by clinicians and (3) ML models require expert annotated supervisory labels to train, that are costly and labor-intensive to acquire. In this thesis, we address the challenge of high false alarm rates through a novel objective that controls false alarms via a direct constraint on precision. Next, we present probabilistic models to handle missing features in both regularly and irregularly spaced time-series models. For regularly spaced time-series models, we incorporate a probabilistic approach that handles missing features via exact marginalization, thereby avoiding predictions that depend on specific guesses of the missing value. In irregularly spaced time-series models, we propose a generative model that learns the data-dependent missingness mechanism, ultimately yielding superior extrapolative performance compared to alternatives. Finally, to manage limited labels, we leverage a semi-supervised training objective that outperforms alternatives by simultaneously maximizing generative performance across all features (both labeled and unlabeled), while ensuring high-quality predictions on the labeled data. The solutions proposed here address significant challenges in clinical risk prediction models, paving the way for their integration into routine hospital care.

Dedicated to my dear mother. The ones we love are always in our hearts.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my parents, Dr. Pratap C. Rath and (late) Dr. Sabita Panda and my sister, Dr. Pallavi Rath. Their unwavering support, encouragement, and belief in me have been the cornerstone of my journey. Their sacrifices, love, and guidance have made this accomplishment possible, and I am forever grateful. I would also like to extend my heartfelt thanks to my advisor, Dr. Michael Hughes. His mentorship, insightful advice, and constant support have been invaluable throughout my research. His guidance has not only shaped my academic growth but also inspired me to strive for excellence. I am sincerely thankful to my dissertation committee for their invaluable feedback on my thesis. Their expertise and insightful comments have greatly enhanced the quality of this work. To my family, thank you for your endless love, understanding, and patience. Your support has been my source of strength, and your faith in my abilities has motivated me to overcome challenges and achieve my goals. I am also deeply appreciative of all my lab mates and mentors who have contributed to this journey. Your collaboration, camaraderie, and shared knowledge have enriched my research experience. Thank you for your encouragement and for fostering an environment of learning and growth. This thesis is a testament to the collective support and guidance I have received from each of you. Thank you.

Contents

List of Tables	ix
List of Figures	xi
1 Executive Summary	1
2 Introduction	5
2.0.1 Potential for enhancing patient care with electronic health records and machine learning	5
2.0.2 Electronic health records datasets for predicting patient deterioration	6
2.0.2.0.1 Extracting time-stamped labs and vital measurements:	7
2.0.2.0.2 Deriving outcomes:	7
2.0.3 Complications in operationalizing EHR based ML models in clinical settings	8
2.0.4 Outline	9
3 Optimizing Early Warning Classifiers to Control False Alarms via a Minimum Precision Constraint	10
3.1 Early Warning Systems : Current Landscape and Issues	10
3.2 Background	12
3.2.1 Evaluation metrics for binary classifiers	13
3.2.2 Suggested Optimization Objective	14

3.2.3	Comparison of Our Proposed Objective to Neyman-Pearson Theory	15
3.2.3.0.1	Confusion Matrix Comparison	16
3.2.4	Baseline: Post-Hoc Threshold Search	17
3.3	Related Works	17
3.4	Methods	20
3.4.1	Previous Bounds using Hinge Loss	21
3.4.2	New Tighter Sigmoid Bounds	22
3.4.3	Practical Implementation	25
3.5	Experimental Evaluation	27
3.5.1	Synthetic Data Generation	27
3.5.2	Results on Synthetic Data	29
3.5.3	Results on Semi-Synthetic Data	30
3.5.4	Per-Sequence Mortality Prediction	31
3.5.5	Deployment Scenario for Mortality Alerts	33
3.5.5.0.1	MIMIC	33
3.5.5.0.2	eICU	33
3.5.6	Comparison to other possible bounds	35
3.5.7	Sensitivity to batch size	36
3.5.8	Demonstration of Models Satisfying Precision Constraint during Training	38
3.5.9	Receiver Operating Curve (ROC) and Precision-Recall (PR) Curves for Sequence Level Predictions of Mortality Risk	39
3.6	Discussion and Conclusion	39
4	Predicting In-Hospital Patient Outcomes from Time Series with Missing Data via a Semi-supervised Hidden Markov Model	43
4.1	Challenges in Predictive Modeling of Critical Care: Addressing Missing Features and Limited Labels	43
4.1.1	Benefits Of Our Generative Modeling Approach	46

4.1.2	Is Our Model Implicit or Explicit?	46
4.2	Methods	48
4.2.1	Problem formulation	48
4.2.2	Model : Prediction Constrained HMMs for Semi-supervised Classification and Ordinal Regression	50
4.2.3	Justifying Usage of The Prediction Constrained (PC) Framework	55
4.2.4	Baseline neural nets for fully-supervised classification with miss- ingness	57
4.2.5	Baseline neural nets for semi-supervised time series classifica- tion without missingness	58
4.2.6	Justification of the $O(TK^2)$ belief computation runtime for HMMs	60
4.3	Results	62
4.3.1	Toy Data Experiment	62
4.3.2	Real Data Experiments	63
4.3.3	Data	64
4.3.4	Mortality Prediction Results	65
4.3.5	Length of Stay Prediction	69
4.3.6	Interpretation of Learned PCHMM States	69
4.3.7	Training times for each model	71
4.3.8	Parameter counts and Hyperparameter search for each model .	72
4.4	Conclusion	73
5	Probabilistic Modeling of Data-Driven Missingness in Irregular Time Series Medical Records	75
5.1	Introduction	75
5.2	Prior Work	77
5.3	Background	81
5.3.0.0.1	Problem: Irregular time series with miss- ingness.	81

5.3.1	Continuous Recurrent Unit (CRU)	82
5.3.1.0.1	CRU overview.	82
5.3.1.0.2	CRU prior on z	83
5.3.1.0.3	CRU likelihood of y given z	84
5.3.1.0.4	Posterior of z given y	84
5.3.1.0.5	CRU summary.	85
5.4	Methods	86
5.4.1	CRU model for seen features only	87
5.4.2	CRU-FM generative model	87
5.4.3	Training	89
5.4.4	Implementation Details	89
5.4.5	Evaluating Extrapolations	90
5.5	Experimental Results	91
5.5.1	Synthetic Experiments	91
5.5.2	Results on Real EHR Datasets	95
5.5.2.0.1	Results: Overall extrapolation error.	96
5.5.2.0.2	Results: Individual feature extrapolation.	100
5.5.2.0.3	Qualitative visuals of extrapolations.	101
5.5.2.0.4	Alternative method: CRU for concatenation of features and missingness mask.	102
5.5.2.0.5	Predicting future missingness of a channel.	102
5.5.2.0.6	Extrapolation performance on extended list of channels	103
5.6	Conclusion	103
6	Conclusion and Outlook	106
6.0.1	Limitations and Future Work	107
	Bibliography	109

List of Tables

3.1	Confusion Matrix	16
3.2	Precision-recall performance on the semi-synthetic task	30
3.3	Precision-recall performance for per-sequence in-hospital mortality predic- tion on MIMIC-III and eICU	31
4.2	MIMIC-IV Dataset Split and Statistics	65
4.4	eICU Dataset Split and Statistics	65
4.1	MIMIC-IV features	66
4.3	eICU features	67
4.5	AUPRC with confidence intervals for mortality prediction with all labels known	68
4.6	Comparison of aggregation strategies of the HMM beliefs, before be- ing passed as features to the predictor	69
4.7	Computation time (seconds/epoch) required per model	71
4.8	Range of parameter counts and number of jobs for hyperparameter search per model	72
4.9	Hyperparameters for all the models	73
5.1	List of features for Physionet	98
5.2	List of features for MIMIC-IV	99
5.3	List of features for eICU	100
5.4	Test RMSE of CRU (w/missing mask as input) and our CRU-FM on eICU dataset	102
5.5	Performance comparison for predicting future missingness of a channel	103

5.6 Extrapolation performance on extended list of channels on 3 EHR
datasets comparing 5 baselines with our CRU-FM 104

List of Figures

2.1	Increasing trend of Machine Learning papers focusing on EHR over the last 15 years	6
3.1	Illustration of the zero-one functions and sigmoid bounds used to count false positives and true positives	19
3.2	Comparison of decision boundaries from BCE, hinge, adversarial prediction and our sigmoid bounds	28
3.3	Improvement in deployment scenario performance for predicting in-hospital mortality (in MIMIC) with our sigmoid bounds	34
3.4	Improvement in deployment scenario performance for predicting in-hospital mortality (in eICU) with our sigmoid bounds	35
3.5	Decision boundaries learned on toy data with weighted cross entropy baseline	36
3.6	Sigmoid bound objective's sensitivity to batch size	37
3.7	Evidence of successful false-alarm-averse training of a model for MIMIC mortality risk prediction	38
3.8	ROC curves for models trained on MIMIC using different optimization objectives	40
3.9	ROC curves for models trained on eICU using different optimization objectives	41
4.1	Visualization of likelihood function for ordinal regression with 3 classes using the logistic distribution to simulate latent function noise	53
4.2	Demonstration of proposed PC-HMM on an "easy" binary classifier task where naive imputation fails	63

4.3	AUPRC versus amount of labeled data for in-hospital mortality early warning classifiers on two large EHR datasets	68
4.4	Interpretation of learned PC-HMM for mortality prediction	70
4.5	Interpretability of learned PC-HMM for Length of stay prediction . .	71
4.6	AUPRC versus amount of labeled data for length-of-stay prediction on MIMIC-IV at 3 ordinal labels (> 3 , > 7 , and > 11 days)	72
4.7	AUPRC versus amount of labeled data for length-of-stay prediction on eICU at 3 ordinal labels (> 3 , > 5 , and > 7 days)	72
5.1	Diagram of our proposed CRU-FM model	86
5.2	Examples of generated sequences for the synthetic task	91
5.3	Synthetic task results comparing CRU-FM to competitors	93
5.4	Demonstration of our CRU-FM model's ability to learn the data-dependent missingness mechanism	95
5.5	Comparison of extrapolating measurements with our CRU-FM v/s 5 competitor methods on 3 EHR datasets	97
5.6	Illustrative examples of predicted trajectories from the eICU dataset, demonstrating that our CRU-FM outperforms the CRU in both extrapolation and interpolation of sequences.	101

Chapter 1

Executive Summary

The rise of big data and machine learning (ML) has the potential to improve the quality and timeliness of medical care for hospitalized patients. Machine learning is a branch of artificial intelligence that involves training algorithms to recognize patterns and make decisions based on data. Given a time series of recent lab results and vital signs, an ML system can make personalized predictions of short-term risk of in-hospital mortality or onset of a life-threatening disease.

Three key challenges stand in the way of effective ML models in these applications:

- **High False Alarm Rate:** Deployed models frequently generate an excessive number of false alarms, undermining their utility and trustworthiness in clinical settings.
- **Missingness:** At each moment an observation is made, most labs and vitals are not updated either due to differences in frequency of measurements or due to the caretaker's judgment of the patient's condition. Common open-access electronic health records (EHR) datasets, such as MIMIC-III [Pollard and Johnson III, 2016], PhysioNet [Goldberger et al., 2000], and eICU [Pollard et al., 2018a] contain missingness rates of $>80\%$ [Che et al., 2018a].
- **Limited Labels:** Supervisory labels for every sequence may be limited due to the expense and time investment needed to obtain accurate disease annotations from medical staff. Several disease cohorts have missing labels in open-access electronic health records (EHR) datasets.

The existing solutions to these problems in the modern analysis of EHRs are inadequate. Existing deep models handle feature missingness by either using overly simplistic imputation strategies [Harutyunyan et al., 2019a, Purushotham et al.,

2018a] or by imputing deterministically without uncertainty [Cao et al., 2018a, Che et al., 2018a]. Moreover, these models often neglect data-dependent missingness patterns (for e.g. certain lab measurements are only collected for critically ill patients), are sensitive to the initialization of missing values [Cao et al., 2018a], or require hundreds of thousands of parameters with a lack of clinical interpretability [Cao et al., 2018a, Harutyunyan et al., 2019a, Purushotham et al., 2018a]. Later, when these models are deployed, they often lead up to 90% of all alarms being false alarms [Fernandes et al., 2020]. Another common issue is that ML models rely heavily on labels and obtaining precise annotations of disease labels from clinicians is often laborious and expensive.

The goal of my thesis is to address these 3 barriers to deploying ML prediction systems in clinical care settings. I will introduce new probabilistic models to handle missing data, limited labels and new paradigms for training to address false alarms and unlabeled data.

First, in work published at AISTATS 2022 *, I introduced a novel training objective that can directly constrain false alarm rates in binary classifiers [Rath and Hughes, 2022]. This objective defines differentiable bounds on the total counts of true and false positives that are tighter than previous work, enabling effective gradient-based learning. Clinical prediction models trained with this objective require false alarms rates to be below a tolerable value (specified by physicians), thus ensuring that clinical resources are directed at patients who are truly at-risk. When predicting in-hospital mortality on two large EHR datasets, experiments showed that our proposed objective led to absolute gains of 0.01-0.15 in recall while satisfying the desired false alarm rate.

My second project, published at 2 workshops, addresses both feature missingness and limited labels [Lobo et al., 2023, Rath et al., 2022]. We proposed a solution based on the prediction constrained hidden Markov model (PC-HMM)[†]. This builds on previous work [Hope et al., 2021, Hughes et al., 2018, Rabiner and Juang, 1986a]

*Code URL: github.com/tufts-ml/false-alarm-control

[†]Code URL: github.com/tufts-ml/pchmm-missing-data-limited-labels/

by adding a critical ability to handle missing data. Our proposed PC-HMM model [Rath et al., 2022] is trained with an optimization objective that balances the HMM’s useful generative properties (handling missingness via marginalization) with the essential need for its learned representations to enable effective prediction in the clinical task. Experiments with our PC-HMM model showed absolute improvements of 0.02-0.08 in area under the precision-recall curve (AUPRC) over several state-of-the-art supervised and semi-supervised deep learning models at various levels of missing labels while predicting in-ICU mortality. Additionally, it is straightforward for domain experts to interpret our model due to its small number of parameters, each with a mechanistic role. Analysis revealed that our model enabled interpretability into patient groups that were at-risk. For example our model identified vulnerable patient groups as those with high blood-urea nitrogen (above 20 mg/dL) and high creatinine(above 1.45 mg/dL), which are common indicators of kidney problems [Baha et al., 2021].

In the third project, we develop models for managing irregular time-series data, particularly when dealing with data-dependent missingness, a mechanism referred to as missing at random (MAR)[‡]. This approach is critical for EHR time series, where data collection is often based on data-driven clinical judgments regarding patient health changes. Many existing models either entirely neglect this assumption [Bruun Ipsen et al., 2020, Rubanova et al., 2019b], or opt to instead input missingness indicators directly into deep learning networks [Che et al., 2018a]. Such methods are inadequate because they fail to learn the generative mechanism for missingness. While our earlier proposed PC-HMM model should be able to handle this scenario, we transition towards a continuous time model to accommodate data collected at irregular intervals. To handle irregular timing of measurements as well as data-dependent missingness scenarios, we extend a encoder-decoder stochastic differential equation (SDE) architecture named the continuous recurrent unit (CRU) [Schirmer et al., 2022] to form a generative model that jointly produces an irregular time series of features and missingness indicators. In experiments across 3 health records

[‡]Code URL: github.com/tufts-ml/data-driven-missingness-cru-irregular-timeseries

datasets, our integrated model (CRU-FM) demonstrates lower extrapolation error than several alternative models, highlighting the importance of learning the data-dependent missingness mechanism in critical care time-series applications.

Ultimately, using the tools developed in this thesis, we envision a future where deployed ML models could use routine physiological measurements (vitals, lab tests, and demographics) collected irregularly over time with missingness, to accurately identify patients at risk and generate alarms that clinicians could rely on to signal need for early intervention, even when large labeled training sets are not easily available.

Chapter 2

Introduction

2.0.1 Potential for enhancing patient care with electronic health records and machine learning

Electronic health records (EHRs) fundamentally serve as clinical information systems designed to collect, maintain, and display electronic data that is gathered over time through the provision of healthcare services. In recent years, the expansive growth of EHRs and the development of sophisticated machine learning (ML) techniques have provided opportunities to enhance patient care in hospital settings, where the stakes are high and the demands are incessant. The integration of vast arrays of digital health information into EHR systems, ranging from patient vital signs and laboratory test results to medication orders and clinical notes, provides the ideal platform for deploying advanced ML algorithms. These algorithms aim to harness the predictive potential of complex, time-series data accumulated during patient hospital stays, with the promise of offering clinicians real-time insights that could anticipate patient trajectories, prevent adverse events, and tailor treatments more effectively. This capability to transform extensive data into actionable insights presents a vital bridge to the application of machine learning in creating robust early warning systems that significantly improve patient safety and treatment outcomes.

Predicting future patient outcomes from observed time series of labs and vitals in critical care settings has drawn significant recent research interest, catalyzed by the release of open-access datasets [Johnson et al., 2020, Pollard et al., 2018a], benchmark classification tasks [Harutyunyan et al., 2019b, Jarrett et al., 2021, Purushotham et al., 2018b, Wang et al., 2020b], and compelling clinical deployments [Hyland et al., 2020b, Sendak et al., 2020]. Figure 2.1 shows the growing interest in this area of research. One particular application of interest to clinicians in developing such EHR based models is early warning alert systems. An early warning alert warns clinicians

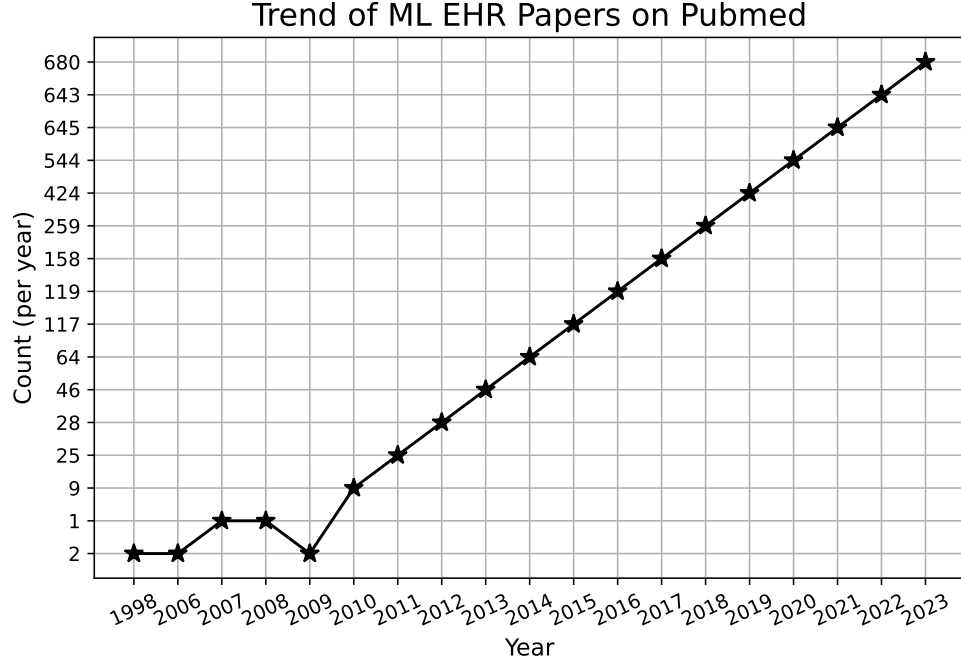


Figure 2.1: Increasing trend of Machine Learning papers focusing on EHR over the last 15 years

of impending deterioration in the patient’s health. Several proxy labels are used to predict deterioration in the patient such as admission to the ICU, onset of a disease, increased length of stay at the hospital or in-hospital/in-ICU mortality. In our work, we focus on early prediction of mortality (as a binary outcome) to avoid focusing on a specific disease cohort. Later, in chapter 4 we extend our models to predict the length of stay as an ordinal outcome, demonstrating their versatility across a range of clinical scenarios.

2.0.2 Electronic health records datasets for predicting patient deterioration

In this section, we elaborate on how we utilize 2 publicly available EHR datasets: MIMIC [Pollard and Johnson III, 2016] and eICU [Pollard et al., 2018b]. These datasets are rich with time-stamped vital signs, laboratory measurements, medications, care plan information, diagnosis, patient history, microbiology cultures, and

billing information. In our work, we mainly focus on utilizing laboratory measurements and vital signs.

2.0.2.0.1 Extracting time-stamped labs and vital measurements: Deriving labs and vital measurements from the datasets is straightforward due to the standardized formats for data storage. For instance, MIMIC utilizes a system of internal identifiers, known as ITEMIDs, to categorize and store various types of clinical data including labs and vitals. Laboratory test results are recorded in the LABEVENTS table, which includes details such as the ITEMID—a unique identifier for each lab test—along with the *HADM_ID* (hospital admission ID), *CHARTTIME* (timestamp of the lab result), and the lab values themselves. The *VALUENUM* field captures the numerical results of the test, and *VALUEUOM* denotes the unit of measurement. Similarly, vital signs are meticulously logged in the CHARTEVENTS table, a comprehensive repository that chronicles all charted data, including routine vital signs like heart rate, blood pressure, respiratory rate, temperature, and oxygen saturation. Each entry in this table is tagged with an ITEMID, indicating the type of data recorded, complete with timestamps, values, and occasionally additional information such as whether the data was manually entered or automatically recorded.

2.0.2.0.2 Deriving outcomes: Outcomes such as mortality and length of stay in the hospital can often be directly obtained from the ADMISSIONS table, which contains entries for the date and time of death during the hospital admission. Features and outcomes for each patients are linked from multiple tables (such as PATIENTS, ADMISSIONS, and ICUSTAYS) to get a comprehensive view of the patient’s timeline at the hospital.

2.0.3 Complications in operationalizing EHR based ML models in clinical settings

Despite the public availability of large scale EHR datasets, the practical application of adopting ML models in clinical settings is often hampered by significant hurdles. First among these is the prevalence of missing data in EHRs—an inevitable consequence of the sporadic nature of healthcare monitoring and data entry practices. Vital signs or lab results might be recorded irregularly, entries may be omitted during acute patient care, or data may simply be lost or incorrectly entered. Roughly 60 – 80% of hourly measurements are missing in datasets such as MIMIC and eICU [Purushotham et al., 2018a, Rath et al., 2022]. This missing data poses a critical challenge for ML models, which typically require complete datasets to function optimally. Without robust mechanisms to handle gaps in data, the accuracy of predictions and the effectiveness of decision-support systems can be severely compromised.

Moreover, the labels required to train these predictive models—such as indications of disease onset or patient outcomes—are often sparse. Label sparsity occurs because the intensive labor required to annotate medical records accurately is both time-consuming and costly, restricting the availability of well-labeled datasets [Lam et al., 2021, Wessler et al., 2023]. Given the constraints on label sparsity, there is a compelling need for machine learning models that can learn effectively from limited labeled data, making semi-supervised learning approaches particularly valuable in the healthcare context. Semi-supervised learning models are designed to capitalize on the large amounts of unlabeled data that are typically available in EHR systems, using them to enhance learning and improve model accuracy even when comprehensive, expert-labeled datasets are not available.

Another significant challenge is the high rate of false alarms generated by many ML systems. These false alarms can arise from over-sensitive models that err on the side of caution, flagging potential issues that do not pose real threats to patient health. While initially appearing as a cautious strategy, a high rate of false alarms can lead to 'alarm fatigue' among healthcare providers, reducing their attention

to alerts and decreasing their trust in the system’s accuracy. This can ultimately undermine the utility of ML applications in clinical settings, as healthcare providers begin to disregard alerts, potentially missing critical warnings that require immediate attention [Sendelbach and Funk, 2013].

Collectively, these challenges—missing data, sparse labels, and a high rate of false alarms—degrade the performance and trustworthiness of ML-based predictive systems in healthcare. Addressing these issues is not merely a technical challenge but a prerequisite for ensuring that the promise of ML to enhance patient care can be fully realized in a practical, sustainable, and clinically beneficial manner. As such, developing robust ML models that can effectively navigate these complexities is critical for the advancement of intelligent healthcare systems that clinicians can rely on to support their decision-making in real-time.

2.0.4 Outline

Having provided a background on clinical prediction models and obstacles in operationalizing these models in practice, below is an outline on the subsequent chapters in this thesis.

Chapter 3 delves into optimizing early warning systems to control false alarms through a minimum precision constraint. It elaborates on new tractable bounds for maximizing recall and empirical success in realistic clinical tasks. Chapter 4 introduces methods for predicting patient outcomes from time-series EHR with missing entries and limited labels by employing a semi-supervised Hidden Markov Model to enhance prediction accuracy and manage data sparsity. Chapter 5 discusses probabilistic modeling of data-driven missingness in irregular time series medical records, proposing a new model that better handles the irregularity and missingness common in electronic health records. Finally, Chapter 6 offers a discussion and outlook, reflecting on the impacts and future directions of the research. This structure underscores a progressive deepening into specific challenges and innovations in machine learning for healthcare monitoring.

Chapter 3

Optimizing Early Warning Classifiers to Control False Alarms via a Minimum Precision Constraint

3.1 Early Warning Systems : Current Landscape and Issues

Machine learning has led to state-of-the-art early warning alert systems for many high-stakes applications, from public health to finance to earthquake safety [Al Banna et al., 2020, Escobar et al., 2020, Mousavi et al., 2019]. In this thesis, we are motivated by applications in healthcare, especially recent early warning alert systems for critical care hospital settings [Hyland et al., 2020b, Sendak et al., 2020, Wellner et al., 2017]. In such settings, alerts are intended to trigger extra attention from clinical staff on specific patients predicted to be at risk of adverse outcomes. Clinicians have limited time that could have many productive uses; it is critical that automated alerts identify patients who truly need help and do not suggest those who do not need attention.

An unneeded alert, also known as a *false alarm* or a *false positive*, has two detrimental consequences. In the moment, it pulls resources away from where they could be better used. In the long term, too many false alarms can cause clinical staff to distrust the alerts all together, a phenomenon known as *alarm fatigue* [Cvach, 2012, Deb and Claudio, 2015, Sendelbach and Funk, 2013]. Some false alarms are inevitable, especially in clinical tasks where the adverse outcome to predict is quite rare. In the mortality risk task we study later, only about 8-10% of patients die in the hospital, and most survive many hours before death. If a system is able to issue

alerts throughout a patient’s stay, it is critical that the model is designed to avoid alarm fatigue and deliver overall net benefit. If the tool fails to limit false alarms to an acceptable rate, alerts may be ignored completely.

The key challenge of designing alert systems is thus to *balance* the false alarm rate (related to *precision*) with the true positive rate (known as *recall*) [Romero-Brufau et al., 2015]. In the clinic, recall measures the fraction of truly at-risk patients correctly identified by an alert. Unfortunately, standard objectives for training early-warning classifiers, such as binary cross entropy (BCE), are not designed specifically to address false alarms. Many early warning systems for clinical settings [Futoma et al., 2017, Hyland et al., 2020b] are trained using such objectives and only balance false alarm concerns in a secondary threshold selection or early stopping stage *after* training. Such post-hoc adjustment is limited and may not identify the ideal tradeoff between recall and precision. Our experiments show the deficiency of binary cross entropy even with post-hoc adjustment or per-class weighting.

To overcome this challenge, we show that any binary classifier trainable via stochastic gradient descent (SGD) can, by changing its loss function, be steered toward solutions that offer better control of false alarms. Our proposed objective seeks to maximize the number of truly at-risk patients who are helped by alerts (maximize recall), subject to meeting a guarantee on the fraction of all alarms that will be false.

In this chapter, we present two key contributions:

1. New tractable bounds for maximizing recall subject to a minimum precision constraint. While it is easy to compute a model’s precision on a concrete dataset, it is not tractable to compute gradients with respect to precision, making optimization difficult. Previous work by Eban et al. [2017] suggested surrogate bounds based on the hinge loss that are amenable to SGD. However, we find these bounds are too loose and lead to suboptimal performance. We thus derive a family of bounds based on the logistic sigmoid function that can be made arbitrarily tight. We find our new bounds achieve far better performance.

2. Demonstration of empirical success on a realistic clinical task. While

our constrained optimization objective has been suggested before for general-purpose settings [Eban et al., 2017], we find that for clinical early warning systems it is particularly suitable yet not thoroughly studied. Recent methods for false alarm control [Fathony and Kolter, 2020, Tsoi et al., 2021] focus evaluation on small, repurposed tabular datasets, not realistic scenarios for early warning system deployment. This leaves open the critical question of performance in real-world clinical settings. Our experiments on two large EHR datasets suggest that our proposed objective can lead to absolute gains of 0.01-0.15 in recall while satisfying the desired precision constraint (see Sec. 3.5.4 and 3.5.5).

We view these contributions as critical steps to achieving effective early warning systems in clinical settings. We emphasize that none of our technical innovations are specific to healthcare; early warning systems in any domain can benefit from our approach.

3.2 Background

When developing an early warning alarm system, our goal is to estimate a prediction model with parameters θ that can consume a feature vector $x \in \mathbb{R}^D$ and produce a real-valued score $f_\theta(x)$ indicating confidence that some (rare) outcome will occur. Large negative scores indicate certainty the outcome will not occur and large positive scores indicate certainty it will occur. Using a scalar threshold b , we can translate this score into a binary *decision* $\hat{y}(x)$, which equals 1 if $f_\theta(x) > b$ and 0 otherwise. In the intended use case, a positive decision causes the early warning system to trigger an *alarm*.

To train this model, we’ll assume we have a dataset X, Y of N labeled examples: $X = \{x_n\}_{n=1}^N, Y = \{y_n\}_{n=1}^N$, with known binary label $y_n \in \{0, 1\}$ indicating which outcome happened to the example at index n with features $x_n \in \mathbb{R}^D$. In many cases, the outcome of interest (such as mortality) is rare. We’ll refer to the rare label of interest as “positive” or 1, and the common label as “negative” or 0. Let N_+ denote the total count of positive true labels in a dataset: $N_+ = \sum_{n=1}^N y_n$. Similarly, N_- gives

the count of negative true labels.

3.2.1 Evaluation metrics for binary classifiers

Many performance metrics exist for binary classifiers, each appropriate for different goals [Romero-Brufau et al., 2015]. We review relevant metrics here.

Binary cross entropy (BCE) is defined as

$$\text{BCE}(\theta, X, Y) = \sum_{n=1}^N \log \left(\sigma(f_{\theta}(x_n))^{y_n} (\sigma(-f_{\theta}(x_n)))^{1-y_n} \right).$$

Here, $\sigma(f) = \frac{1}{1+e^{-f}}$ is the logistic sigmoid function, which maps real-valued inputs $f \in \mathbb{R}$ to the unit interval $0 \leq \sigma(f) \leq 1$. BCE is the most common loss used to train binary classifiers, motivated as a smooth upper bound on error rate (after scaling by $\frac{1}{N}$) as well as via maximum likelihood arguments. However, for problems where the positive class is rare but critical, neither error rate nor BCE can capture the key applied questions as all types of mistakes are treated equally.

True positive count. A “true positive” is an example whose true label is positive and predicted label is also positive. We define a dataset’s true positive count as:

$$\text{tpc}(\theta, X, Y) = \sum_{n:y_n=1} \hat{y}_{\theta}(x_n) = \sum_{n:y_n=1} z(f_{\theta}(x_n) - b).$$

Here, $z(\cdot)$ is the zero-one function, which is one if its argument has positive sign and zero otherwise. This count is bounded: $0 \leq \text{tpc} \leq N_+$.

False positive count. A “false positive” is an example whose true label is negative but whose predicted label is positive. False positives mean the early warning system produces a false alarm. We define a dataset’s false positive count as:

$$\text{fpc}(\theta, X, Y) = \sum_{n:y_n=0} \hat{y}_{\theta}(x_n) = \sum_{n:y_n=0} z(f_{\theta}(x_n) - b).$$

This count is also bounded: $0 \leq \text{fpc} \leq N_-$.

Recall. Recall is the fraction of all truly positive examples that are also called

positive by the classifier:

$$\text{recall}(\theta, X, Y) = \frac{\text{tpc}(\theta, X, Y)}{N_+(Y)}, \quad (3.1)$$

where $N_+(Y)$ counts the positive labels in dataset Y .

Precision (aka True Alarm Rate). Precision is defined as the fraction of all positive alerts produced by the classifier that are truly positive,

$$\text{prec}(\theta, X, Y) = \frac{\text{tpc}(\theta, X, Y)}{\text{tpc}(\theta, X, Y) + \text{fpc}(\theta, X, Y)},$$

where the denominator counts all alerts (positive calls), which must either be true positive or false positive. Another name for precision is the “true alarm rate”. Precision is equal to one minus the false alarm rate.

3.2.2 Suggested Optimization Objective

Consider a clinical prediction task trying to identify patients at risk of a rare adverse outcome, so that additional interventions can be prioritized for these patients. Interventions are a limited resource with a cost (otherwise they could be applied to all patients). At minimum, hospital staff time is limited, and thus an alarm raised for a patient who does not need extra care means that other (perhaps more needy) patients do not get attention. Furthermore, if staff become habituated to viewing alarms as rarely related to the intended outcome and thus unhelpful, they may be inclined to ignore them. Similar concerns about false alarms shape early warning systems in many other domains.

It is thus critical to view designing an effective early warning system as satisfying two goals. First, guaranteeing the false alarm rate is below some critical value established in conversation with stakeholders to ensure utility and avoid alarm fatigue. Second, achieving alerts that would successfully alarm (and hopefully help mitigate) the most possible adverse events.

These goals can be naturally formalized as the following optimization problem:

$$\max_{\theta} \text{recall}(\theta, X, Y), \text{ s.t. } \text{prec}(\theta, X, Y) \geq \alpha, \quad (3.2)$$

where α is set to the minimum desired precision (equivalently, $1 - \alpha$ is the maximum false alarm rate). While this objective has been suggested before [Eban et al., 2017], in practice most ML early warning systems are not trained to satisfy some target minimum precision.

One could ask why we intend on constraining precision rather than recall. To answer this, we argue that our proposed precision constraint reflects the limited resources of the system (staff time and trust), while constraining recall does not. If we constrained recall to at least 85%, the system could only alarm for 85% of needy patients when 90% or 95% could be saved with little degradation of precision.

3.2.3 Comparison of Our Proposed Objective to Neyman-Pearson Theory

The Neyman-Pearson paradigm for binary classification focuses on maximizing the recall (i.e., minimizing the type II error) while ensuring that the type I error is controlled below a user-specified threshold α [Tong et al., 2020]. The optimization objective can be formally expressed as:

$$\max_{\theta} \text{recall}(\theta, X, Y), \text{ s.t. } \text{tnr}(\theta, X, Y) \geq \alpha, \quad (3.3)$$

where $\text{tnr}(\theta, X, Y)$ refers to the true negative rate, representing the proportion of correctly identified negatives.

While this objective is valid, our proposed objective focuses on precision rather than the true negative rate. The emphasis on precision ensures that the model prioritizes correctly identifying the positive class (at-risk patients) rather than merely achieving a high true negative rate. This distinction is critical in heavily imbalanced medical datasets, where the negative class (normal patients) is often the majority.

Our primary objective is to maximize the effectiveness of identifying the positive class, which directly impacts patients at-risk.

3.2.3.0.1 Confusion Matrix Comparison To illustrate this difference, consider the confusion matrix below:

	Predicted Positive	Predicted Negative
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

Table 3.1: Confusion Matrix

In the Neyman-Pearson framework, the focus is on controlling the true negative rate (TNR), defined as:

$$\text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

This approach emphasizes minimizing the false positive rate to maintain a high true negative rate. However, in clinical settings, especially with imbalanced datasets, the number of true negatives (TN) is typically much larger than the number of true positives (TP). Thus, even a model with a high TNR might still have a low precision if the number of false positives (FP) is relatively high compared to true positives.

Our proposed objective, which focuses on precision, is defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

By constraining precision, our model ensures that the majority of positive predictions are correct, thereby directly reducing the false positive rate in a more balanced manner relative to the true positives. This focus is crucial for clinical applications where false alarms (false positives) can lead to alarm fatigue among healthcare providers, reducing the overall effectiveness of the monitoring system.

While the Neyman-Pearson approach ensures a high true negative rate, our precision-constrained objective better addresses the practical needs of clinical settings by prioritizing the accurate identification of at-risk patients, thus providing

more reliable and actionable predictions.

3.2.4 Baseline: Post-Hoc Threshold Search

After training a binary classifier to minimize cross entropy (or some other suitable objective), we can always perform a post-hoc search procedure to better identify a decision-making threshold b that meets desired performance criteria. In our applications, if we have a maximum allowable false alarm rate in mind, we can fix the parameterized score function $f_{\theta}(\cdot)$ obtained via training and simply select among a grid of candidate threshold values the one that best satisfies the original objective in Eq. (3.2). Graphically, for a linear classifier this is akin to trying all possible decision boundary hyperplanes parallel to the one produced via original training. As a one-dimensional grid search, precision and recall at each threshold can be easily computed, without any gradients or bounds needed.

This approach is helpful but far from optimal, as shown in Fig. 3.2, where we consider a synthetic dataset with $D = 2$ features where we wish to achieve a minimum precision of $\alpha = 0.9$. Even with this post-hoc search, the optimal linear classifier trained via BCE cannot exceed a precision of 0.68, well below the desired precision of 0.9. Later experiments show that post-hoc search also delivers sub-par results in real clinical tasks.

3.3 Related Works

Optimization methods. Many previous efforts have focused on optimization objectives that target classifier performance metrics to try to balance precision and recall in some fashion [Burges et al., 2006, Lipton et al., 2014, Rakotomamonjy, 2004, Yue et al., 2007]. As we review below, most of these methods either pursue different objectives (less appropriate for our use case than Eq. (3.2)), have limited scalability to large datasets, or have deficiencies in approximation quality.

Optimizing area under the precision-recall curve (AURPC) has been pursued by Metzler and Croft [2005] and Caruana and Niculescu-Mizil [2006], as well as recently

by Ramzi et al. [2021] and Qi et al. [2021], who both propose tractable surrogate losses for gradient-based optimization. However, optimizing AUPRC cannot maximize the quality of recall at a specific precision value, as our later method can. Figure 3.9 provides a real-data example of where one method dominates in AUPRC, but another (ours) achieves a better recall at the desired false alarm rate. Similar concerns exist for approaches that optimize the F-measure, such as the sigmoid approximations of Tsoi et al. [2021] or the pseudo-linear methods of Narasimhan et al. [2015] and Puthiya Parambath et al. [2014].

Fathony and Kolter [2020]’s adversarial prediction (AP) framework allows gradient-based training for a variety of non-decomposable objectives (such as our Eq. (3.2)). However, their AP method suffers from scalability issues, with runtime *cubic* in the number of examples in each minibatch. Our work is conceptually simpler and keeps runtime *linear* in the number of examples. Scalability is also a concern for the methods of Joachims [2005], which pursue optimizing precision at fixed recall at cost quadratic in the number of examples.

We build upon the work of Eban et al. [2017], which developed a framework for gradient-based learning of non-decomposable losses like our objective in Eq. (3.2). Eban et al. accomplish this via tractable hinge-loss bounds of the counts of true and false positives (as detailed in Sec. 3.4.1). However, we find in practice these hinge bounds are so loose (see Fig. 3.1) that they cannot solve the problem: solutions often do not satisfy the desired precision constraint (see Fig. 3.2). We will show that our tighter bounds and more careful treatment of the optimization problem lead to noticeably better solutions. Our surrogate bound ideas originally appeared in a preliminary workshop paper [Rath and Hughes, 2021]. This present paper offers more rigorous conceptual justification as well as substantially expanded empirical evaluation (comparing to more alternative methods as well as more realistic “continual monitoring” tasks).

Clinical methods. Some false alarm control methods have been directly developed for use in a hospital (Chambrin [2001], Antink et al. [2016], Eerikäinen et al. [2016]). Au-Yeung et al. [2019] reduce false alarms in the ICU via post-hoc feature

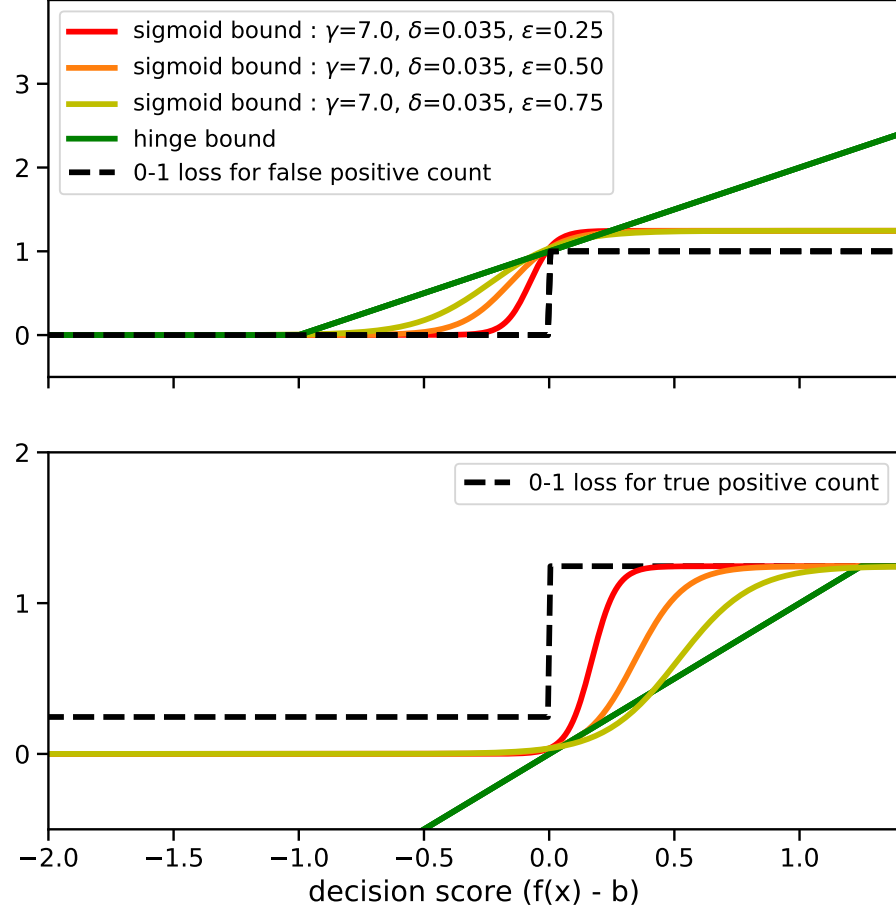


Figure 3.1: Illustration of the zero-one functions used to count false positives (*top*) and true positives (*bottom*), as well as bounds that make gradient-based learning feasible. In both plots, our family of sigmoid bounds is noticeably tighter than the hinge bound of Eban et al. [2017]. *Bottom panel:* We bound a 0-1 loss that is *vertically-shifted* by $+\gamma\delta$, so our bound is non-negative for all inputs (unlike the hinge) and thus maintains the validity of our precision constraint as discussed around Eq. (3.5).

selection, but do not encode limits on false alarms into any training objective. Hever et al. [2019] reduce false alarms for arrhythmia by training random forests to match expert rules. However, such rules are not easily available for all tasks. Our work allows direct control of the desired precision level and applies to many models and tasks.

3.4 Methods

Our ideal optimization objective given a training dataset $X, Y = \{x_n, y_n\}_{n=1}^N$ is to maximize recall subject to a minimum precision constraint, as in Eq. (3.2). While we can easily evaluate this objective for any candidate parameters θ , *training* our model – that is, finding good values for parameters θ given a dataset – is difficult. The key barrier is that the functions involved are based on sums of the flat zero-one function (shown in Fig. 3.1) and thus have gradient values equal to zero at almost all θ values. We could use gradient-free methods, but these are inefficient when θ is high-dimensional as it will be in most real settings involving many input variables. We need to transform this problem so that modern gradient-based learning will be effective.

Eban et al. [2017] suggest a promising route that leads to an unconstrained objective with non-zero gradients, which we review here and build upon. First, recognizing that maximizing recall is equivalent to maximizing the true positive count, can rewrite our problem in terms of true and false positive counts:

$$\max_{\theta} \text{tpc}(\theta), \quad \text{subj. to: } \frac{\text{tpc}(\theta)}{\text{tpc}(\theta) + \text{fpc}(\theta)} \geq \alpha. \quad (3.4)$$

Here for simplicity we write the counts as a function of parameters θ alone. While these counts do depend on the observed training data X and Y , we assume this data is known and fixed throughout training.

Next, we can rewrite our problem in an equivalent “standardized” form by framing the constraint as a function that must be less than or equal to zero, and minimizing the negative instead of maximizing:

$$\begin{aligned} \min_{\theta} \quad & -\text{tpc}(\theta), \\ \text{subj. to:} \quad & \underbrace{-\text{tpc}(\theta) + \frac{\alpha}{1-\alpha}\text{fpc}(\theta)}_{g(\theta)} \leq 0. \end{aligned} \quad (3.5)$$

We emphasize that this transformation is only valid when the sum of $\text{tpc} + \text{fpc}$ is

strictly positive, as we need to multiply both sides of the constraint in Eq. (3.4) by this sum while preserving the intended inequality.

Using well-established optimization theory [Chong and Żak, 2013], we can transform this to an equivalent unconstrained optimization problem via the penalty method with Lagrange multiplier $\lambda > 0$:

$$\min_{\theta} -\text{tpc}(\theta) + \lambda \max(0, g(\theta)) \quad (3.6)$$

Here, $g(\theta)$ is the penalty function defined in Eq. (3.5). If this penalty function is zero or less than zero, this means the desired constraint is *satisfied*: the precision is at least α .

This unconstrained formulation avoids the need to handle a difficult non-linear constraint, but still faces the key problem that the objective as a function of θ is *flat*. That is, infinitesimal changes in θ are unlikely to move any example’s prediction from one side of the decision boundary to the other, and thus both tpc and fpc counts will remain the same for almost all small changes to θ . This flatness is a problem because it means any attempt at gradient-based training will not move the parameters θ from their original (poor performing) values.

3.4.1 Previous Bounds using Hinge Loss

Eban et al. [2017] obtain tractability - informative non-zero gradients at almost all possible θ parameter values - by defining non-flat *bounds* on the counts using the hinge loss function $h(y, a) = \max(0, 1 - s(y)a)$, where $a \in \mathbb{R}$ is a real value and $s(y)$ is the sign function, which returns +1 if $y = 1$ and -1 if $y = 0$.

First, the false positive count is upper bounded by

$$\text{fpc}^h(\theta) = \sum_{n:y_n=0} h(y_n, f_{\theta}(x_n)). \quad (3.7)$$

Similarly, the true positive count is lower bounded by

$$\text{tpc}^h(\theta) = \sum_{n:y_n=1} 1 - h(y_n, f_{\theta}(x_n)). \quad (3.8)$$

These bounds can replace the tpc and fpc terms in Eq. (3.6), which is then minimized using a gradient-based solver. These hinge bounds are visualized in Fig. 3.1, where we can observe two key problems. First, they are *loose* bounds: for just one term in the sum each bound can differ by 1 or more from the ideal value. Aggregated across all N_- or N_+ terms, the total error can be substantial. The looseness of the hinge bounds of Eban et al. [2017] can result in a fundamentally different optimal decision boundary than our proposed tighter sigmoid bounds (introduced later in Sec. 3.4.2), as illustrated in Fig 3.2. Second, the sum $\text{fpc}^h + \text{tpc}^h$ could be less than zero, as tpc^h may be negative. This condition would spoil the interpretation of the objective in Eq. (3.6) as equivalent to guaranteeing a specific minimum precision α , since we derived it assuming the denominator in Eq. (3.4) was strictly positive.

3.4.2 New Tighter Sigmoid Bounds

We now derive two *families* of functions that bound the zero-one function, one family of upper bounds (for the false positive count) and one family of lower bounds (for the true postive count). These are illustrated in Fig. 3.1. Both bounds are formed by shifting and scaling the logistic sigmoid function $\sigma(a) = \frac{1}{1+e^{-a}}$. Concrete bounds can be obtained by fixing tolerance hyperparameters to specific values. By varying tolerances, a user can make the bounds tighter (more accurate but with flatter gradients) or looser (more tractable). Reasonable settings deliver tighter bounds than the hinge bounds of Eban et al. [2017]. Furthermore, we'll show that under all conditions our bounds meet the positivity condition required by the denominator in Eq. (3.4).

Upper bound on fpc. We first seek a non-flat *upper bound* on the zero-one function, denoted $u(a)$. We want this smooth function to meet the following conditions necessary for a tight upper bound:

$$\begin{aligned} u(-\infty) &\rightarrow 0 & u(-\epsilon) &\approx \delta \\ u(+\infty) &\rightarrow 1 + \gamma\delta & u(0) &\approx 1 + \delta \end{aligned} \tag{3.9}$$

where tolerance factors $\epsilon > 0, \delta > 0$ and scaling factor $\gamma \geq 1$ are specified by the user. Large values make the function smoother; small values make the bound tighter. In the limit as $\gamma \rightarrow 1, \epsilon \rightarrow 0, \delta \rightarrow 0$, our function converges to the zero-one function.

A natural choice for $u(a)$ is to make it a sigmoid whose amplitude is $1 + \gamma\delta$ with learnable slope and shift,

$$u_{m,b}(a) = (1 + \gamma\delta)\sigma(ma + b), \quad (3.10)$$

where the two parameters are slope $m \in \mathbb{R}$ and intercept $b \in \mathbb{R}$. By definition, this class of function satisfies the two limit conditions in Eq. (3.9). To meet the two approximation constraints, we can numerically solve for the optimal parameters that minimize squared error:

$$\hat{m}, \hat{b} = \arg \min_{m \in \mathbb{R}, b \in \mathbb{R}} (\delta - u_{m,b}(-\epsilon))^2 + (1 + \delta - u_{m,b}(0))^2.$$

Thus, for user-specified tolerance parameters γ, δ, ϵ , we can obtain a tight non-flat differentiable upper bound on the false positive count:

$$\text{fpc}^\sigma(\theta) = \sum_{n: y_n=0} (1 + \gamma\delta)\sigma(\hat{m}f_\theta(x_n) + \hat{b}). \quad (3.11)$$

If we concretely select a modest setting of our tolerance parameters – $\gamma = 7.00$, $\delta = 0.035$, $\epsilon = 0.75$ – we use BFGS to solve the minimization problem and yield slope $\hat{m} = 6.85$ and shift $\hat{b} = 1.59$. Our choice of tolerance parameters being deemed ‘modest’ was based on the finding that setting γ, δ, ϵ yielding tighter bounds did not significantly impact performance on synthetic.

Vertically shifting tpc to enable valid lower bounds without changing the optimal parameter. When defining a lower bound for true positive counts, the bound should be strictly positive for all inputs, so that the denominator positivity condition in Eq. (3.4) is satisfied. First, we pick a small positive constant $\gamma\delta$ (where $\gamma > 0, \delta > 0$ are not necessarily the same values used to define fpc above). Now, we redefine the true positive count function by adding this small constant to each

zero-one function:

$$\text{tpc}_{\gamma,\delta}(\theta) = \sum_{n:y_n=1} \gamma\delta + z(f_\theta(x_n)) \quad (3.12)$$

This vertical shift is visualized in Fig. 3.1.

Instead of the original unconstrained problem in Eq. 3.6, we then solve a revised optimization problem:

$$\begin{aligned} \min_{\theta} -\text{tpc}_{\gamma,\delta}(\theta) + \lambda \max(0, g_{\gamma,\delta}(\theta)), \\ g_{\gamma,\delta}(\theta) = -\text{tpc}_{\gamma,\delta}(\theta) + \frac{\alpha}{1-\alpha} \text{fpc}(\theta) + \gamma\delta N_+. \end{aligned} \quad (3.13)$$

Here the constraint function $g_{\gamma,\delta}$ differs from the original g by substituting $\text{tpc}_{\gamma,\delta}$ and adding constant $\gamma\delta N_+$.

Lemma 3.4.1 *Vertically shifting the true positive count does not alter the optimal parameter. Given fixed values of scalars $\lambda, \alpha, \gamma, \delta$, both Eq (3.6) and Eq. (3.13) have the same optimal parameter θ^* .*

Proof: For all possible θ , we have $g(\theta) = g_{\gamma,\delta}(\theta)$, because inside $g_{\gamma,\delta}$ the constant added in the $-\text{tpc}_{\gamma,\delta}$ term is exactly canceled by the extra additive constant term $\gamma\delta N_+$. Thus, the penalty terms in the two objectives will always yield the same values. The minimization objectives thus differ only by an additive constant (tpc vs. $\text{tpc}_{\gamma,\delta}$) and will have the same local optima. $\square$

Lower bound for the shifted tpc. We now seek a lower bound ℓ of our vertically-shifted zero-one function, meeting the conditions:

$$\begin{aligned} \ell(-\infty) &\rightarrow 0 & \ell(0) &\approx \delta \\ \ell(+\infty) &\rightarrow 1 + \gamma\delta & \ell(+\tilde{\epsilon}) &\approx 1 + \delta \end{aligned} \quad (3.14)$$

Again tolerance factors $\epsilon > 0, \delta > 0, \gamma \geq 1$ are specified by the user, with similar interpretation as before. We emphasize that this lower bound ℓ is a distinct function from

the upper bound u , as shown in Fig. 3.1. Even if we set the tolerance parameters the same for both bounds (which is not necessary but done for simplicity), the optimization desiderata in Eqs. (3.9) and (3.14) are different, yielding distinct functions ℓ and u .

To obtain a function ℓ meeting the goals of Eq. (3.14), we define a sigmoid function with amplitude $1 + \gamma\delta$ using the same functional form as Eq. (3.10). Given tolerances γ, δ, ϵ , our lower bound on the vertically-shifted true positive count becomes

$$\text{tpc}_{\gamma\delta}^{\sigma}(\theta) = \sum_{n:y_n=1} (1 + \gamma\delta) \sigma(\tilde{m}f_{\theta}(x_n) + \tilde{b}). \quad (3.15)$$

Again, there are two parameters: slope $\tilde{m} \in \mathbb{R}$ and intercept $\tilde{b} \in \mathbb{R}$. Using the same optimization strategies as for the upper bound above, we solve for the values of $\tilde{m}, \tilde{b}$ parameters that best match the desiderata in Eq. (3.14). Concretely fixing tolerances to $\gamma = 7.00$, $\delta = 0.035$, $\epsilon = 0.75$, we obtain $\tilde{m} = 6.85$ and $\tilde{b} = -3.54$.

By plugging our lower bound for tpc and our upper bound for fpc into Eq. (3.13), we achieve a tractable objective amenable to gradient-based training. A strong advantage of using strict *bounds* as we do is the guarantee that if the optimal parameter θ^* found by optimizing Eq. (3.13) satisfies the constraint penalty (the g term evaluates to less than zero), then the precision will be at least α . Simply using differentiable approximations of the zero-one function instead of strict bounds would not provide such a guarantee.

3.4.3 Practical Implementation

Our proposed bounds allow tractable gradient-based optimization of our unconstrained objective in Eq. (3.13). Here we discuss practical engineering efforts that make these bounds usable on large real-world datasets.

Avoid local optima with many initializations. Linear-boundary classifiers like logistic regression (minimizing BCE) or support vector machines (minimizing hinge loss), have *convex* training problems. However, due to our sigmoid bounds

even when our classifier f_θ is a generalized linear model, our objective is *non-convex* and thus sensitive to initialization. To avoid poor local optima, we take the best of many random initializations (in terms of validation-set performance), where each run is initialized via a “Glorot” procedure [Glorot and Bengio, 2010]. While keeping the best of many increases runtime if runs are done sequentially, separate initializations are easily parallelized across many computers so that overall runtime is no worse than the longest individual run. Even without full parallelization, we argue our approach is *worthwhile* for the substantial gains in improved recall at the desired precision, especially because training happens much less often than prediction in a deployed alert system. Alternatively, one could develop a homotopy approach [Watson, 1989] to navigate the non-convex landscape and avoid local-optima. However, the success of the homotopy method heavily relies on the choice of the initial simple problem and the path defined using a family of pre-selected parametric function. If the homotopy is not well-designed, it might not provide a useful path, or it might lead to numerical instability.

Minibatch gradient descent. If datasets are small enough, we can optimize our unconstrained objective in Eq. (3.13) using gradients computed from *all* examples. To scale to larger datasets we use stochastic estimates of the gradient from a minibatch and update parameters via the Adam optimizer [Kingma and Ba, 2014b], trying learning rates from 0.0005 to 0.005 and selecting the best performing one on the validation set.

Using minibatches, however, means that stochastic estimates of the gradient of the $\max(0, g(\theta))$ term in Eq. (3.13) are not unbiased, because the maximum is performed *outside* the sum over all training examples. To mitigate this possible bias, we simply select large minibatches (500 or more examples). While not formally unbiased, in practice we find this choice is effective. We can reach high quality solutions that meet the intended precision when assessed on the entire training set. To avoid any bias, we could incrementally update per-batch gradients as well as their sum over iterations for a predefined set of batches, an approach known as Stochastic Average Gradient [Schmidt et al., 2017].

Batch size and penalty hyperparameters. In the supplement, we study the sensitivity of performance to batch size. Batches of 512 or 1024 examples are preferred over larger batches. We found that trying two λ values, 1000 and 10000, was sufficient to meet the desired false alarm constraint ($g(\theta) \leq 0$ is satisfied after training). We use these λ values in all experiments.

Runtime cost analysis. Our method’s training cost will scale *linearly* in the number of examples in each batch. Computing our unconstrained objective in Eq. (3.13) at a specific θ can be done by evaluating each of the terms fpc^σ and $\text{tpc}_{\gamma,\delta}^\sigma$ once, which requires just one sigmoidal bound function evaluation ($\ell(\cdot)$ or $u(\cdot)$) for each example (see (3.11) and (3.15)). Computing the *gradient* has the same linear complexity as the loss itself, using well-known complexity results for back-propagation.

Chosen objective in practice. In pursuit of our ideal but intractable objective in Eq. (3.2), optimizing our tractable unconstrained objective in Eq. (3.13) is guaranteed to produce a valid solution to the original objective if the penalty function $g_{\gamma,\delta}$ is less than or equal to zero. In practice, we find that often optimizing Eq. (3.13) is too strict and does not yield a satisfactory result. However, another strategy, dropping the additive constant $\gamma\delta N_+$ term in $g_{\gamma,\delta}$ and then post-hoc verifying the desired precision constraint is satisfied, works well. We recommend trying both ways and using validation-set performance to select the best.

3.5 Experimental Evaluation

3.5.1 Synthetic Data Generation

We generated positive and negative examples by sampling from 2-D Gaussian distributions as described below :

For the positive class, three blobs are created from 3 different Gaussians with means positioned at $[0.7, 2.5]$, $[0.7, 1.0]$, and $[0.7, 0.0]$ and variances $[0.06, 0.1]$, $[0.1, 0.1]$ and $[0.06, 0.06]$ respectively. The number of samples in these blobs are 30, 60, and 30, respectively.

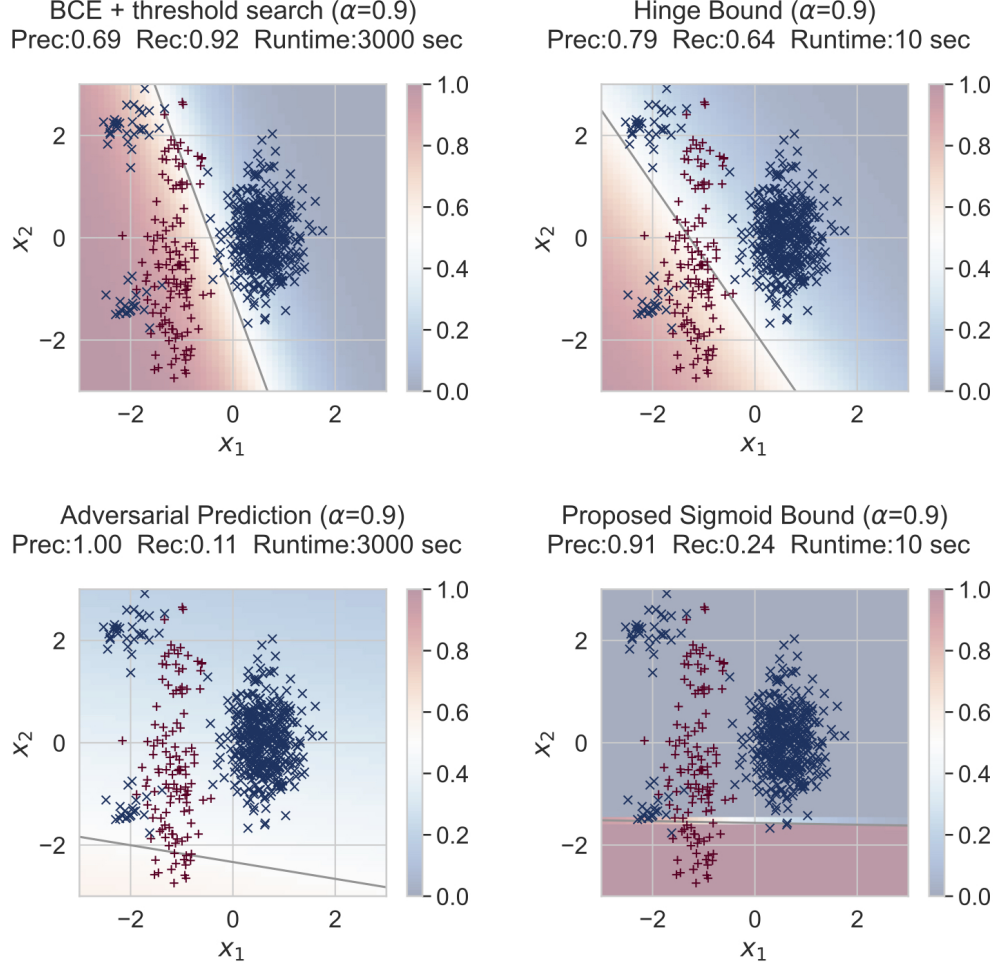


Figure 3.2: Linear decision boundaries (black lines) found by various methods on toy data when stated goal is to maximize recall subject to a minimum precision of $\alpha = 0.9$. Subtitles report precision, recall, and total runtime for each method’s solution. Red and blue markers represent positive and negative examples, respectively, for the toy dataset used for training (Sec. 3.5.2). *Top Left*: Even with post-hoc search, binary cross entropy (BCE) cannot exceed a precision of 0.684. *Top Right*: Despite including the desired $\alpha = 0.9$ precision constraint in its objective, Eban et al. [2017]’s hinge bound cannot satisfy the constraint. *Bottom*: Both Fathony and Kolter [2020]’s AP method and our proposed sigmoid bound satisfy the precision constraint; ours delivers 2x the recall of the AP solution in $1/300^{th}$ of the time.

Similarly, for the negative class, blobs are positioned at $[2.25, 1.5]$, $[0.0, 3.0]$, and $[0.0, 0.5]$, and variances $[0.1, 0.2]$, $[0.05, 0.05]$ and $[0.05, 0.05]$ with respective sample sizes of 400, 30, and 20. We intentionally sample more negative data-points than positive data-points to create a dataset with class-imbalance, which is common in health record datasets.

3.5.2 Results on Synthetic Data

To gain insight, we designed a toy classification task with 2 features where the solutions favored by different objectives can be easily visualized. Our goal is to train a logistic regression (LR) classifier whose decisions maximize recall subject to a minimum precision constraint of $\alpha = 0.9$. We illustrate the advantage of our logistic sigmoid bound empirically by comparing it with 5 other objectives. Fig. 3.2 shows all training examples ($N_+ = 120, N_- = 450$) together with the learned linear decision boundaries for four possible optimization strategies: BCE with post-hoc threshold selection, Eban et al. [2017]’s hinge-bound approach, Fathony and Kolter [2020]’s adversarial prediction (AP) method, and our proposed sigmoid bound approach. We further compare 2 other methods, class-weighted cross entropy and a log loss surrogate bound in Sec.3.5.6.

Although the BCE solution might look reasonable, recall that our specific goal is to maximize recall subject to a precision of at least 0.9. The BCE solution gets only 0.69 precision even with post-hoc threshold search, and thus remains quite far from the desired precision of 0.9. Similarly, the hinge bound achieves only 0.79 precision, also falling short of our goal. Both the AP method and our approach can satisfy the desired precision constraint. However, our proposed sigmoid bound achieves noticeably better recall of 0.24, which is over double the 0.11 recall found by AP.

To investigate bound quality, we verified that even after initializing at the sigmoid bound’s better solution, optimizing Eq. (3.6) using the hinge bound prefers to wander away, suggesting the inadequacy of this loose bound (see our code repository for a reproducible example).

Finally, we performed an exhaustive grid search of the 3 LR parameters (two weights, one bias), optimizing the ideal objective in Eq. (3.2). (This search is only feasible in low-dimensional problems.) The grid search optimum (not shown) reached precision 0.9 and recall 0.24, suggesting our sigmoid bound’s precision 0.96 and recall 0.20 is close to the optimal to beat for this dataset.

Method	Precision			Recall			Runtime in sec.
	Train	Valid	Test	Train	Valid	Test	
BCE + threshold search	0.70	0.69	0.69	0.61	0.72	0.61	15
Hinge Bound	0.63	0.61	0.63	0.72	0.75	0.71	15
Adversarial Prediction (AP)	0.73	0.67	0.75	0.55	0.66	0.60	9000
Sigmoid Bound (ours)	0.80	0.72	0.74	0.67	0.77	0.76	15

Table 3.2: Precision-recall performance on the semi-synthetic task (Sec. 3.5.3), where the goal is to maximize recall subject to a minimum precision of $\alpha = 0.8$. Only our proposed sigmoid bound satisfies this goal on the training set. Our method also delivers better recall on the test set than all alternatives by at least 0.05 while keeping runtime fast.

3.5.3 Results on Semi-Synthetic Data

Real-world prediction tasks often include many features, most of which are only weakly relevant to the outcome. To assess our method’s robustness to such features, we designed a *semi-synthetic* task. Adapting the toy generation process that produced the 2-dim. feature vectors x and binary labels y in Fig. 3.2, we append to each x vector a 98-dimensional feature vector representing vitals, labs, and demographics for one randomly-chosen patient-stay in the MIMIC-III dataset [Johnson et al., 2016], generating 684 train, 513 valid, and 512 test examples. Our goal is to enforce a minimum precision of $\alpha = 0.8$. We deliberately pursue a slightly lower α here than in Sec. 3.5.2 to illustrate the flexibility of our methods to meet any desired precision constraint. Naturally, lowering the required minimum precision leads to better recall compared to the results in Fig. 3.2. To avoid local optima, we run our method many times (25 random seeds, 2 initialization scale factors, and 2 λ values), selecting the run that maximizes validation recall while meeting the precision constraint.

Table 3.2 shows that in this higher-dimensional problem (100 total features), only our method can meet the desired 0.8 precision even on the training set (others range from 0.63-0.73). Our method also exceeds others on test-set recall by at least 0.05. In terms of runtime, most methods are fast (15 seconds), but the AP method [Fathony and Kolter, 2020] shows poor scalability, requiring 2.5 hours due to its cubic scaling. This poor scalability prevented our use of the AP method on the larger real-world tasks below.

	Method	Recall on MIMIC		Recall on eICU	
		Train	Test	Train	Test
Logistic Regression	BCE + threshold search	.57 (.57, .58)	.54 (.53, .55)	.10 (.09, .10)	.09 (.08, .09)
	Hinge Bound (Eban et al.)	.70 (.69, .70)	.62 (.61, .63)	.02 (.01, .02)	.02 (.01, .02)
	Sigmoid Bound (ours)	.76 (.75, .77)	.68 (.67, .70)	.21 (.20, .22)	.19 (.19, .20)
1-layer MLP	BCE + threshold search	.77 (.76, .77)	.69 (.68, .71)	.32 (.32, .33)	.28 (.27, .29)
	Hinge Bound (Eban et al.)	.74 (.73, .75)	.64 (.62, .65)	.36 (.35, .36)	.23 (.23, .25)
	Sigmoid Bound (ours)	.82 (.82, .83)	.72 (.70, .73)	.38 (.38, .39)	.30 (.29, .30)

Table 3.3: Precision-recall performance for per-sequence in-hospital mortality prediction on MIMIC-III and eICU (Sec. 3.5.4), using target precision $\alpha = 0.9$ on train and $\alpha' = 0.8$ on validation. We show the 50th (5th, 95th) percentiles from 20 randomly-drawn train/validation/test partitions. While all methods meet the desired precision constraint after threshold search, our sigmoid bound method achieves better recall on test data, especially for logistic regression models.

3.5.4 Per-Sequence Mortality Prediction

We now assess our method’s precision-recall performance at mortality risk predictions on two large open-access datasets of critical care EHR: MIMIC-III (Medical Information Mart for Intensive Care) and eICU. Our goal is to predict in-hospital mortality after observing vitals and labs over the entire sequence of a patient’s stay, up to just before discharge or death. We enforce a minimum precision value of $\alpha = 0.9$ on the training set. We found $\alpha = 0.9$ challenging to meet on heldout data even when satisfied at training, so to select the best runs we enforce a minimum validation precision $\alpha' = 0.8$.

MIMIC-III data. The MIMIC-III dataset [Johnson et al., 2016] contains deidentified open-access EHR data from a Boston hospital from 2002-2012. We extract 2 demographics, 10 vitals, and 94 lab measurements discretized to hourly bins for 34472 patient stays using the MIMIC-Extract pipeline [Wang et al., 2020a]. Our train/valid./test splits have 20682/6895/6895 patient-stays, with $\sim 9.5\%$ patient stays resulting in death. Each patient-stay’s multivariate time-series is transformed into a feature vector as follows. For each raw feature, we apply 7 summarization functions (missing indicator; time since last non-missing; plus min, max, median, slope, and variance of non-missing values) to 4 possible time windows (0-100%, 50%-100%, last 16 hr, and last 24 hr). The resulting feature vector has size $106 \times 7 \times 4 = 2968$.

eICU data. The eICU Collaborative Research Database [Pollard et al., 2018b] contains data from 59 critical care units throughout the U.S. We extract 3 demographics, 8 vitals, and 6 lab measurements discretized to hourly bins using eICU Extract [Wang et al., 2020c] for 72670 patient stays. Our train/valid/test splits have 43642/14509/14518 patient-stays, with $\sim 8.2\%$ resulting in death. We featurize each patient-stay the same way as MIMIC, yielding feature vectors of size $17 \times 7 \times 4 = 417$.

Models. On both datasets, we tried two models: logistic regression (LR) and a multi-layer perceptron (MLP) with 1 hidden layer (32 hidden units; RELU activation). We search 25 random seeds, 2 initialization scale factors, 2 λ values, and 5 weight decays.

Mortality Prediction Results. The recall performance of each method on both datasets can be found in Table 3.3. Across 20 independent randomly-chosen train-test partitions of both datasets, we find that all methods can satisfy the desired precision constraint (all methods exceed 0.9 on training and 0.8 on validation). However, our proposed sigmoid bound method consistently achieves better test-set recall values at the selected operating threshold (chosen on validation data). Compared to the BCE baseline, using a linear model we see our method boost the test-set recall (median across partitions) from 0.540 to 0.684 on MIMIC and from 0.087 to 0.195 on eICU. As expected, gains from the MLP are smaller (as a predictor gets more flexible, it can get closer to the optimal non-linear boundary). However, the gains are still noticeable: recall improves from 0.689 to 0.717 on MIMIC and from 0.280 to 0.298 on eICU. Eban et al. [2017]’s hinge bound method is *worse* than the BCE baseline in terms of test-set recall on three of the four dataset-model combinations tested.

Across Table 3.3, recall values in on eICU are noticeably lower than MIMIC (BCE achieves test recall of 0.5-0.7 on MIMIC but only 0.08-0.28 for eICU). This occurs despite the desired minimum precision value remaining the same ($\alpha = 0.9$) and being satisfied in all cases. The drop in recall performance likely occurs because we have more available informative clinical measurements for MIMIC than eICU (106 vs. 17). The eICU task is also slightly more difficult: only 8% of eICU sequences have

positive labels compared to 9.5% for MIMIC. In this challenging setting, enforcing high precision means some modest tradeoffs with recall. We emphasize that without enforcing sufficiently high precision, alarm fatigue could render the whole system ineffective.

To better understand the gains of our method in terms of common evaluation criteria, we present ROC and PR curves for all methods in the appendix. Our method shows consistent gains in both curves.

3.5.5 Deployment Scenario for Mortality Alerts

3.5.5.0.1 MIMIC In order to realistically assess our proposed method as an early warning alert for patients at risk of deterioration, we consider now evaluating predictions made *every 12 hours* throughout a patient’s stay, with the outcome set to in-hospital mortality *within the next 24 hours*. For a patient stay lasting just over two days, we would create and evaluate four feature vectors, representing predictions made after 12, 24, 36, and 48 hours. This realistic setting exaggerates class imbalance (only 3% of examples are positive). To compensate, we lower the desired minimum precision to $\alpha = 0.7$ on train and $\alpha' = 0.6$ on validation. For simplicity, we featurize by applying the same summary functions used above on 2 windows (0-100% and 90-100%).

Fig. 3.3 illustrates the precision-recall tradeoff on test data for LR models in this scenario. At the chosen operating threshold (marked as a cross), our proposed sigmoid bound achieves better recall and precision than alternatives. We show a similar improvement in recall and precision in the deployment scenario for eICU in Fig. 3.4 in the appendix.

3.5.5.0.2 eICU In order to assess our proposed method realistically as an early warning alert for patients at risk of deterioration, we consider now evaluating predictions made *every 12 hours* throughout a patient’s stay, with the outcome set to in-hospital mortality *within the next 24 hours*. For a patient stay lasting just over two days, we would create and evaluate four feature vectors, representing predictions

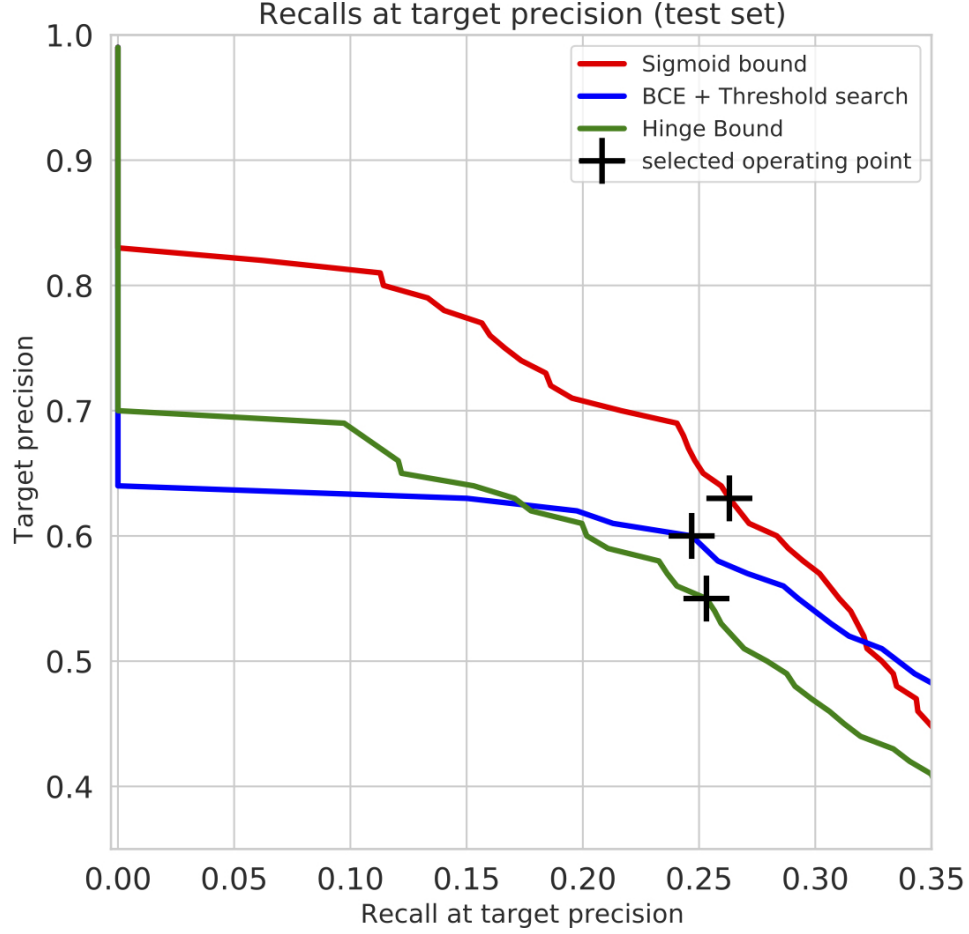


Figure 3.3: Predicting in-hospital mortality on MIMIC in a deployable scenario (making a prediction every 12 hours), using target precisions of $\alpha = 0.7$ on train and $\alpha' = 0.6$ on valid. Our proposed sigmoid bound improves precision by 0.026 (100's of fewer false alarms across the 6895 patient-stays in the test set), and improves recall by 0.01 (100's more truly at-risk patients identified).

made after 12, 24, 36, and 48 hours. This realistic setting exaggerates class imbalance (only 2.3% of examples are positive). To compensate, we lower the desired minimum precision to $\alpha = 0.7$ on train and $\alpha' = 0.6$ on validation. For simplicity, we featurize by applying the same summary functions used above on 2 windows (0-100% and 90-100%).

Fig. 3.4 illustrates the precision-recall tradeoff on test data for LR models in this scenario. At the chosen operating point (marked as a cross), our proposed sigmoid bound achieves better recall and precision than alternatives.

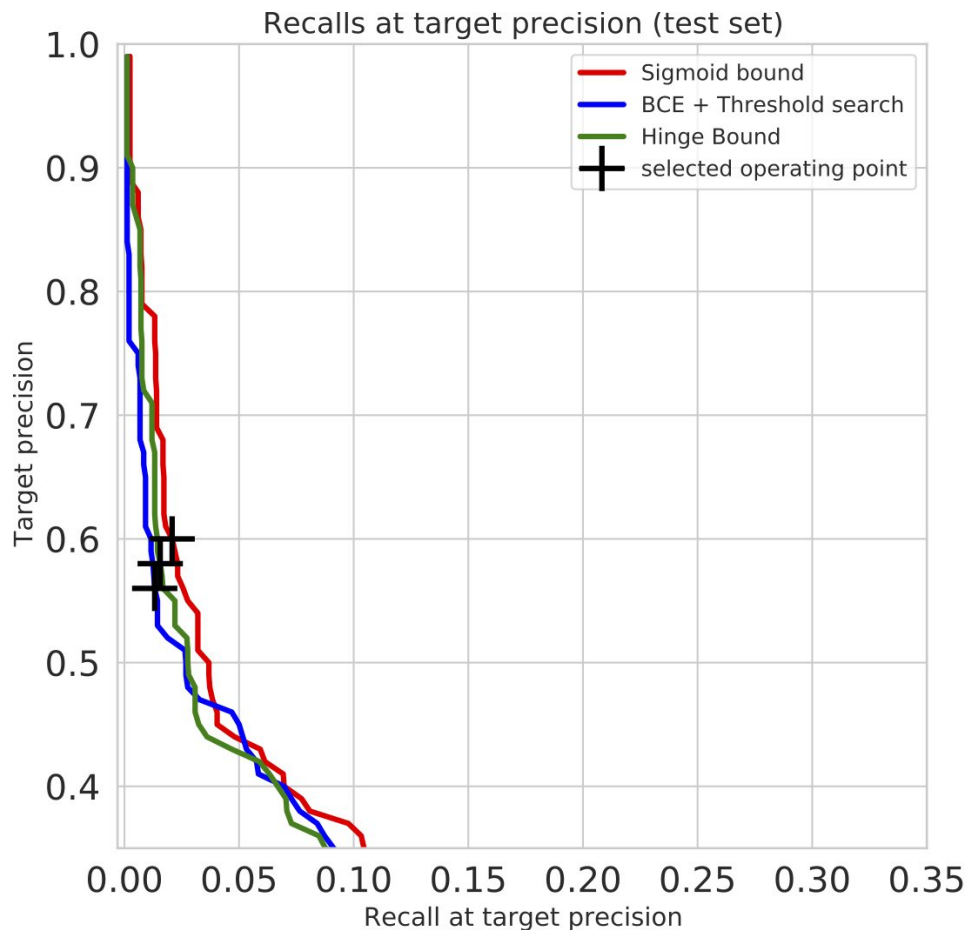


Figure 3.4: Predicting in-hospital mortality on eICU in a deployable scenario (making a prediction every 12 hours), using target precisions of $\alpha = 0.7$ on train and $\alpha' = 0.6$ on valid. Our proposed sigmoid bound improves precision by 0.02 (100's of fewer false alarms across the 14518 patient-stays in the test set), and improves recall by 0.014 (100's more truly at-risk patients identified).

3.5.6 Comparison to other possible bounds

We tried 2 alternative methods on our toy 2D example from Sec. 5.1. First, we tried a **weighted cross entropy** classifier, varying the cost of positive examples across 12 log-spaced values from 0.01 to 1000 (See figure 3.5). None of these met the desired precision constraint of 0.9 (the best of these got precision 0.7). Second, we tried a **log loss** surrogate bound (instead of the hinge in Eban et al's approach). Again, this did not meet our desired 0.9 precision (it got precision=0.72, recall=0.56).

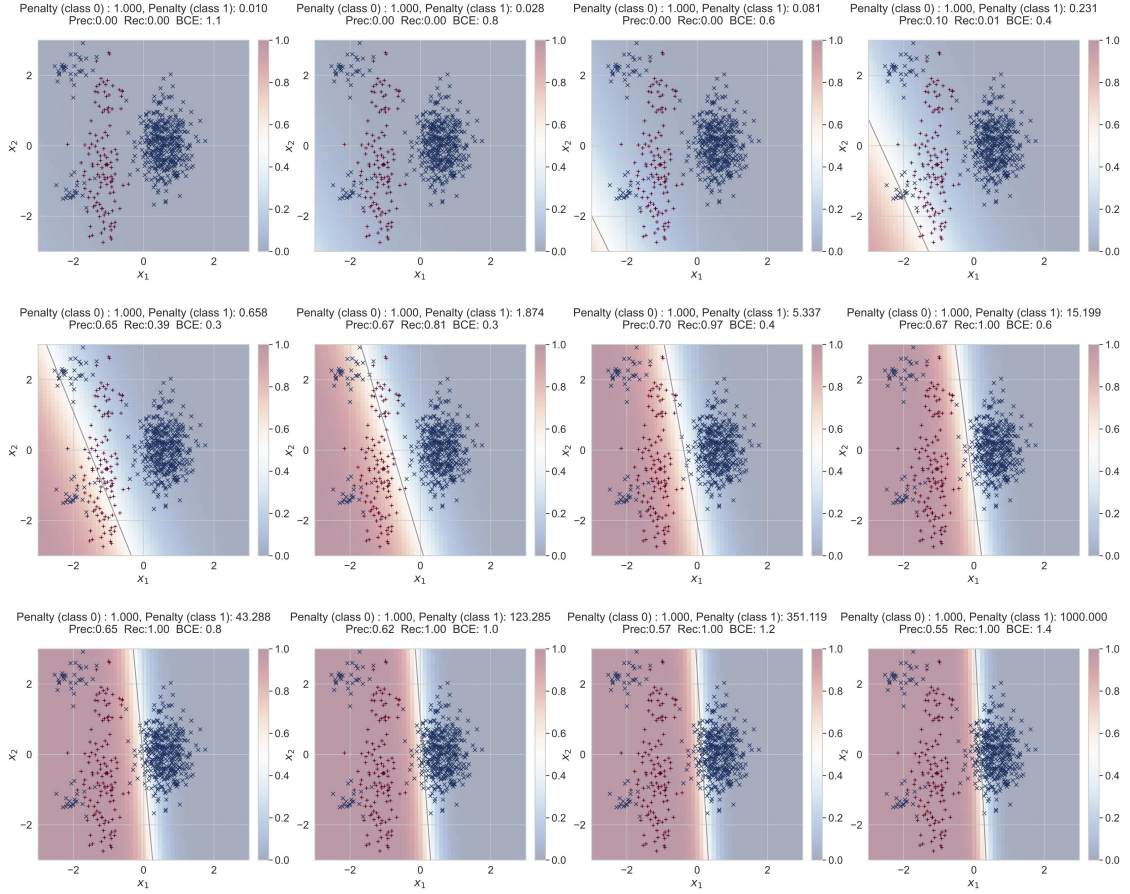


Figure 3.5: We tried a **weighted cross entropy** classifier, varying the cost of positive examples (R7) across 12 log-spaced values from 0.01 to 1000. None of these met the desired precision constraint of 0.9 (the best of these got precision 0.7)

3.5.7 Sensitivity to batch size

We study our method’s sensitivity to batch size in Fig. 3.6, by plotting the precision and recall achieved after convergence for many random initializations at many possible settings of batch size (from 512 up to the entire dataset), while training on the MIMIC mortality risk prediction task in the main paper. From Fig. 3.6, we conclude that while all batch sizes have some initializations outperform others in terms of precision and recall, overall the best runs using batch sizes of 512 (orange dots) and 1024 (green dots) seem to reach higher precision and recall values than the best runs at other batch sizes. Only runs with 512 and 1024 have precision exceed 0.25 on heldout data (validation and test). We suspect that the “folk wisdom” that

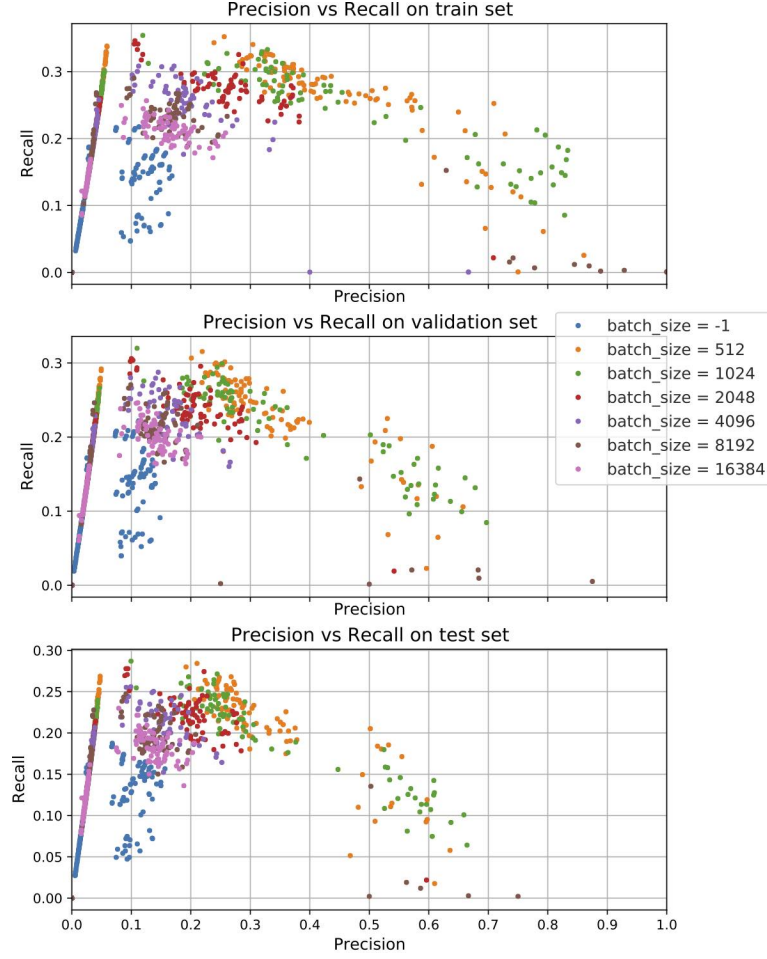


Figure 3.6: Sensitivity of precision and recall as we vary the batch size from 512 up to the size of the entire training set. Rows show the training, validation, and test set performance on the MIMIC-III dataset mortality risk prediction task. Each dot represents the performance of one run of our sigmoid bound method optimized to convergence from one random initialization; we plot many runs for each batch-size setting to show the range of possible local optima. The best runs with batch sizes of 512 and 1024 surpass the best runs at any other tested batch size, as shown by the cluster of orange and green points that reach 0.45-0.75 precision and 0.18-0.26 recall on the test set, when training with the precision constraint, $\alpha = 0.7$. Batch size = -1 indicates the full training set.

small batches add helpful “noise” to stochastic gradient descent in order to avoid bad local optima may be the reason that 512 and 1024 work better than the larger tested batch sizes.

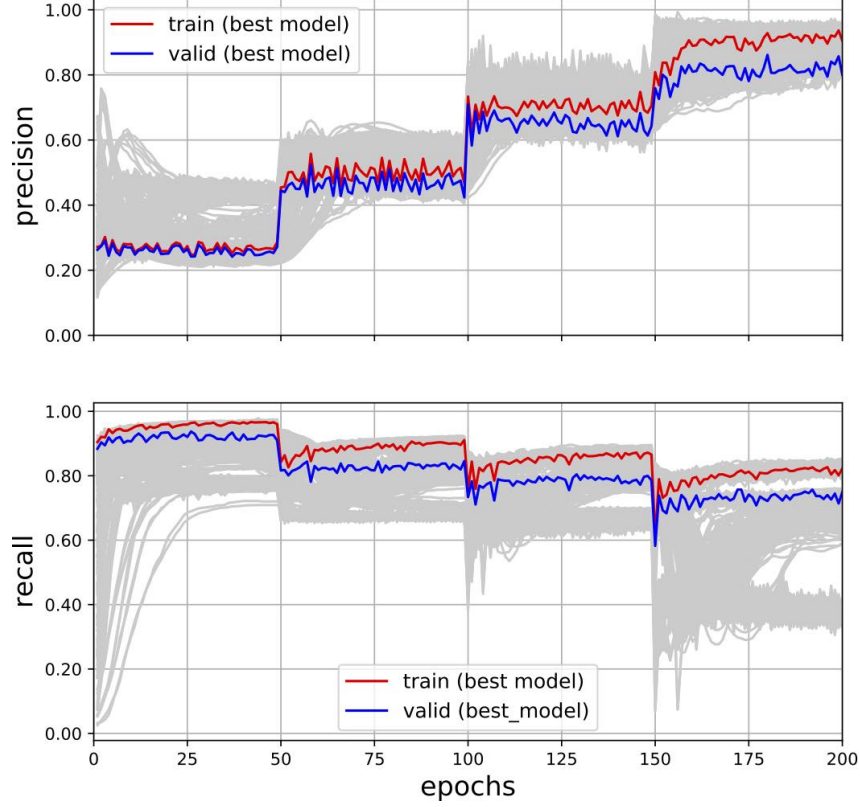


Figure 3.7: Evidence of successful false-alarm-averse training of a model for MIMIC mortality risk prediction. Every 50 epochs, the desired minimum precision α is stair-stepped up, beginning at 0.225, moving at 0.225 increments, and ending at 0.9. After each step, the model’s empirical precision is quickly improved to approximately satisfy the desired α via gradient-based learning. Each gray line is one run from a random initialization, the best performing run (maximizing recall while satisfying our precision goals on training and validation) is highlighted in color.

3.5.8 Demonstration of Models Satisfying Precision Constraint during Training

In figure 3.7, we demonstrate our model’s ability to ramp up to any desired precision during training. While training the MLP for sequence level predictions on MIMIC, we gradually step up the precision constraint (α) to 0.9 at every 50 epochs, and monitor the precision and recall. Consequently, we see that the model’s empirical precision also improved to approximately satisfy the precision constraint (α). Note that ramping up the precision constraint gradually over epochs is only shown for visual validation of our models, and is not the optimal strategy during training, due to local optima at intermediate precision constraints. Training with a fixed precision

constraint (α), using many random initializations, is the optimal strategy to train the models with our sigmoid bound.

3.5.9 Receiver Operating Curve (ROC) and Precision-Recall (PR) Curves for Sequence Level Predictions of Mortality Risk

In Fig. 3.8, and Fig. 3.9, we provide the full ROC and PR curves for all possible objectives for training logistic regression on MIMIC and eICU. We see that our proposed method delivers noticeable gains especially in the precision-recall curve, which we argue is more suitable to early warning systems.

3.6 Discussion and Conclusion

We have developed new bounds based on the sigmoid function that allow tractable gradient-based optimization of any differentiable classifier to meet a desired precision constraint. Our experiments evaluating in-hospital mortality prediction on two large hospital datasets suggest the feasibility of our approach in real early warning applications.

Limitations. A key drawback of our method is vulnerability to local optima due to non-convexity of the sigmoid bound. Given reasonable compute power at training time, we find we can mitigate local optima by taking the best of many runs.

For a deployment, setting α requires input from stakeholders (at what false alarm rate would alerts be ignored or not add value?). As in most real-world ML, we expect a modest but noticeable drop between train and heldout performance. As a rule of thumb, if the desired heldout precision is 0.5, we recommend setting α a bit higher (say $\alpha = 0.6$) for training then enforce $\alpha' = 0.5$ on a validation set to be sure desired performance is achieved. Our method is sensitive to other hyperparameters, including the penalty $\lambda > 0$ and the tolerances γ, δ, ϵ that govern the tractability-tightness tradeoff for our bound. While we found reasonable values for our tasks, future applications may need to tune values for better performance.

Finally, although our objective can control false alarms on the training set, this

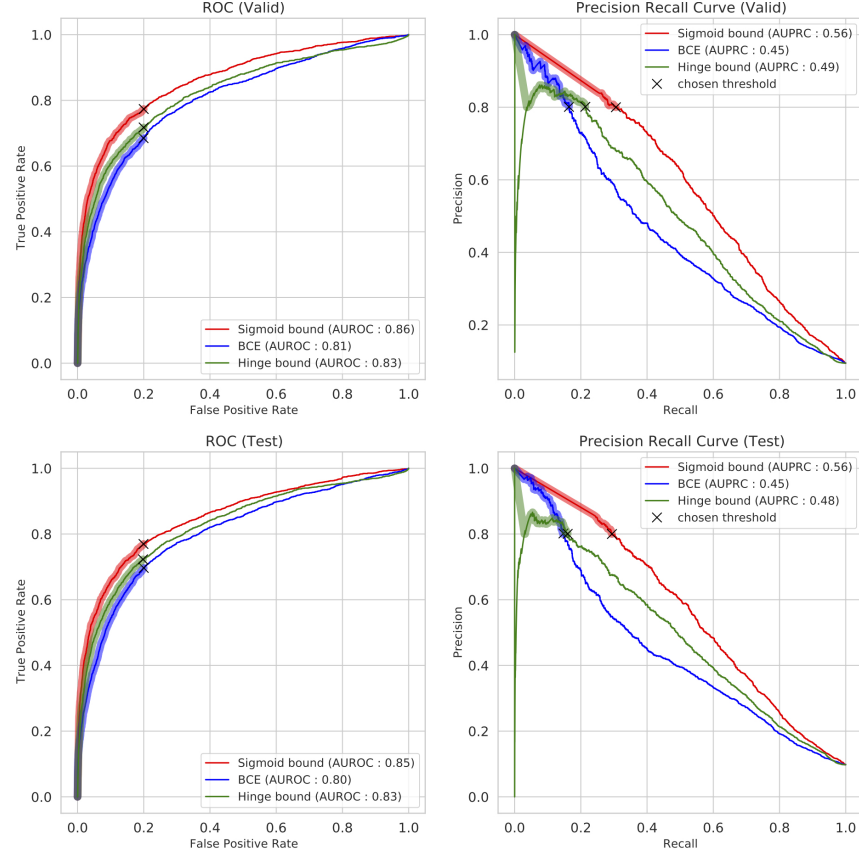


Figure 3.8: Receiver operating curve (ROC, left column) and precision-recall curve (PR, right column) for logistic regression models trained using different optimization objectives on the MIMIC dataset (top row: validation set; bottom row: test set). Each curve shows tradeoffs in performance metrics possible while varying the decision threshold b of a single method. *Left column:* Receiver operating curve (ROC), comparing true positive rate (TPR, y-axis) to false positive rate (FPR, x-axis). *Right column:* Precision-recall curve, comparing precision (also known as positive predictive value (PPV), y-axis) to recall (also known as TPR, x-axis). Our sigmoid bound and Eban et al. [2017]’s hinge bound were trained to maximize recall subject to a constraint on precision: above 0.9 on training set; above 0.8 on validation set. *Darker line segments* indicate thresholds satisfying the desired precision above 0.8 on validation set. *The black “X” marker* indicates the selected operating point for each method (chosen to maximize our optimization objective on the validation set).

does not guarantee this constraint will be satisfied on heldout data. Guaranteeing generalization is an open research question.

Advantages. The primary advantage of our method is the ability to steer models to satisfy a maximum desired false alarm rate, in order to avoid alarm fatigue, while

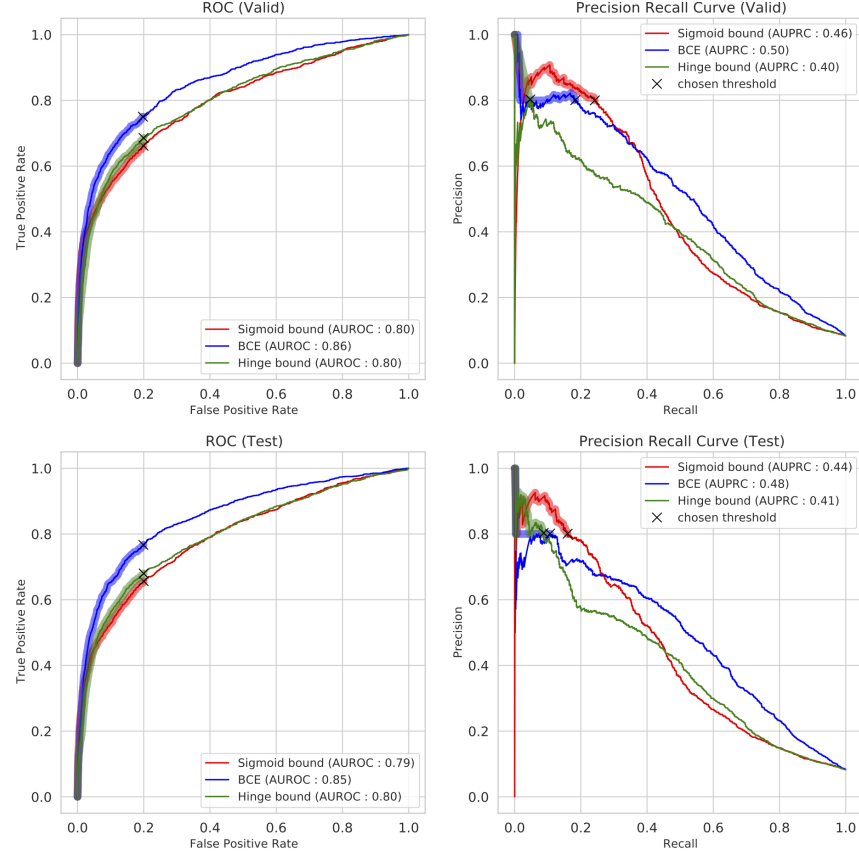


Figure 3.9: Receiver operating curve (ROC, left column) and precision-recall curve (PR, right column) for logistic regression models trained using different optimization objectives on the eICU dataset (top row: validation set; bottom row: test set). Each curve shows tradeoffs in performance metrics possible while varying the decision threshold b of a single method. *Left column:* Receiver operating curve (ROC), comparing true positive rate (TPR, y-axis) to false positive rate (FPR, x-axis). *Right column:* Precision-recall curve, comparing precision (also known as positive predictive value (PPV), y-axis) to recall (also known as TPR, x-axis). Our sigmoid bound and Eban et al. [2017]’s hinge bound were trained to maximize recall subject to a constraint on precision: above 0.9 on training set; above 0.8 on validation set. *Darker line segments* indicate thresholds satisfying the desired precision above 0.8 on validation set. *The black “X” marker* indicates the selected operating point for each method (chosen to maximize our optimization objective on the validation set).

maintaining linear runtime. As demonstrated throughout our experiments, our objective clearly outperforms alternatives like post-hoc threshold search at maximizing recall at a desired minimum precision.

Future work could use our bounds on more flexible classifiers that are trainable via SGD, such as recurrent NNs, convolutional NNs, or graph NNs. For improved

interpretability, one could explore integrating our loss with L0 and L1 norm weight sparsity penalties [Tibshirani, 1996, Ustun and Rudin, 2016].

Chapter 4

Predicting In-Hospital Patient Outcomes from Time Series with Missing Data via a Semi-supervised Hidden Markov Model

4.1 Challenges in Predictive Modeling of Critical Care: Addressing Missing Features and Limited Labels

Recent research has increasingly concentrated on utilizing time-series data of laboratory results and vital signs to predict patient outcomes in critical care settings. This chapter explores two specific challenges associated with these predictive models. First, *feature missingness is common in patient records either due to varied frequency of measurements of various labs and vitals, or if certain physiological measurements are deemed irrelevant for diagnosis*. Second, for many diagnostic tasks, obtaining outcome labels to supervise training is prohibitively expensive, leading to *small labeled sets*. Both challenges require methods that inherently address uncertainty. We argue that simple probabilistic models – hidden Markov models (HMMs) [Ghahramani, 2001, Rabiner and Juang, 1986b] with a few dozen states – have been under-explored yet provide a natural solution. Our experiments show that HMMs with 5-20 states, requiring minimal computational resources, are capable of elegantly capturing temporal trends across various physiological channels, resulting in prediction performance competitive with large sequential neural nets.

To address the problem of *feature missingness*, most efforts to date targeted at critical care time series consider two kinds of approaches: (1) a preprocessing step that imputes missing values via a simple heuristic like population mean filling or forward filling [Harutyunyan et al., 2019b, Harvey et al., 2023, Purushotham et al.,

2018b], followed by a standard time series classifier, or (2) a deep learning approach that performs iterative refinement of missing values following an initial guess [Cao et al., 2018b, Che et al., 2018b, Yoon et al., 2018]. Unlike these approaches, HMMs are probabilistic models that elegantly handle missingness via exact marginalization, avoiding any imputation on the way to prediction. Recent work has also explored other *deep probabilistic* approaches to missingness that hybridize neural nets with Gaussian processes [Futoma et al., 2017], latent or stochastic differential equations [De Brouwer et al., 2019b, Rubanova et al., 2019a], Hawkes processes [Mei and Eisner, 2017], or encoder-decoder models [Li and Marlin, 2020b]. Unlike these approaches, HMMs do not need tens of thousands of parameters or specialized GPU hardware to make good predictions.

To address the *small labeled set* problem, one promising approach is to use semi-supervised learning (*SSL*) [van Engelen and Hoos, 2020], which trains models on the union of a small labeled set and a large unlabeled set of only observed time-series of features, without any known labels. While applications of SSL to critical care time series are limited [Ahuja et al., 2022, Lam et al., 2021, Poulain et al., 2022], there are many tasks for which it would be applicable, such as automating preliminary screening of diseases such as sepsis. One line of work has produced effective SSL for end-to-end deep classifiers of *images* [Berthelot et al., 2019b, Miyato et al., 2018, Sohn et al., 2020]. Recently, Goschenhofer et al. [2021] suggest these methods can be effective for multivariate time-series classification as well, assuming no features are missing. A few very recent methods [Shin et al., 2023, Xing et al., 2022] have developed SSL approaches specialized to time series classification, but neither can handle any missingness in the time series features. Unlike such deep SSL classifiers, we show how HMMs as generative models can be trained on both small labeled sets and large unlabeled sets with arbitrary feature missingness.

Contribution 1: HMMs as a feature extractor for time series with missingness. We show how HMMs can produce a learned fixed-dimensional representation vector from any regularly-sampled time series, regardless of missingness or

length. We take advantage of the probabilistic nature of HMMs, estimating the representation conditioning only on the observed features. This effectively *marginalizes* (integrates away) unobserved feature dimensions. In practical terms, this means we only evaluate the each timestep’s likelihood at observed dimensions. There is no direct imputation or approximation of any missing feature values, eliminating a source of bias or noise.

Contribution 2: Effective supervision and semi-supervision for HMMs.

Our experiments find that common unsupervised or supervised approaches to training latent variable models like HMMs for prediction tasks yield sub-par predictions. To make HMMs succeed at clinical time-series prediction, especially with limited labels, our key methodological insight is *prediction constrained* (PC) training [Hope et al., 2021, Hughes et al., 2017]. This optimization objective balances the HMM’s useful generative properties with the essential need for its learned representations to enable effective prediction in the clinical task. We show that PC training outperforms two common alternatives. First, the two-stage pipeline of unsupervised training of an HMM followed by supervised fine-tuning of a classifier on the frozen HMM’s features (to maximize the likelihood of labels given features). Second, supervised training of an HMM to maximize the joint likelihood of features and labels.

To highlight the versatility of the prediction constrained HMM (PC-HMM) for clinical prediction tasks, we perform experiments pursuing a binary classification task (in-ICU mortality prediction) and an ordinal classification task (categorizing length of stay in the ICU) on two large health records datasets. We show that PC training of HMMs of modest size can match or outperform much larger deep neural networks designed specifically to match one of our stated challenges (feature missingness, limited labels). We compare to GRU-D [Che et al., 2018b] and BRITS [Cao et al., 2018b] as strong baselines for missing features. In the limited labels regime, we compare time-series adaptations [Goschenhofer et al., 2021] of two strong semi-supervised methods for deep classifiers: MixMatch [Berthelot et al., 2019b] and FixMatch [Sohn et al., 2020]. We further show how compact HMMs can be *interpreted* to yield useful clinical insights red both at an individual patient

level and over the entire population. Our PC-HMM results suggest that simple, easy-to-train probabilistic approaches deserve consideration in clinical applications, especially when computational resources are limited. From the environmental costs of training deep models [Strubell et al., 2019] to avoiding expensive GPUs in bespoke hospital settings, there are many reasons to try the PC-HMM in clinical time-series applications with missing features or labels.

4.1.1 Benefits Of Our Generative Modeling Approach

Generative models learn the joint probability distribution $P(X, Y)$ of the input data X and the output labels Y . These models attempt to model how the data is generated by capturing the underlying distribution of the data itself. Discriminative models, on the other hand, learn the conditional probability distribution $P(Y|X)$. They focus on the boundary that separates different classes and are directly optimized for classification tasks. In this work, we choose the generative modeling approach for 2 main reasons:

- Generative models offer a probabilistic framework that effectively manages missing features through exact marginalization.
- Generative models are ideal for leveraging both labeled and unlabeled data, maximizing the use of the large unlabeled set and improving predictive performance even when labeled data is scarce.

Later in our experiments, we also show that our generative model is capable of achieving better predictive performance than several discriminative model alternatives.

4.1.2 Is Our Model Implicit or Explicit?

Our proposed Prediction Constrained Hidden Markov Model (PC-HMM) is an explicit generative model.

Implicit Generative Models [Ngiam et al., 2011, Xie et al., 2016] : Implicit generative models do not specify a direct form for the probability distribution

of the data. Instead, they generate samples from the distribution through a process that is not explicitly defined in terms of a probability density function. The training process focuses on improving the generator to produce increasingly realistic samples, but it does not result in an explicit probability distribution that can be easily sampled or evaluated. One example of such models is a Generative Adversarial Network (GAN).

Explicit Generative Models [Arjovsky et al., 2017, Goodfellow et al., 2014]: Explicit generative models define a specific form for the probability distribution of the data, in terms of a joint probability distribution $P(X, Z)$, where X represents the observed data and Z represents the hidden variables or states. These models use parameters to describe the distribution and provide a way to calculate the probability of observed data given the parameters.

HMM as an Explicit Model In the case of HMMs, they explicitly model the joint distribution of the observed sequence and the sequence of hidden states. An HMM consists of:

- **Transition Probabilities:** $P(z_t|z_{t-1})$ - The probability of transitioning from one hidden state z_{t-1} to another hidden state z_t .
- **Emission Probabilities:** $P(x_t|z_t)$ - The probability of observing x_t given the hidden state z_t .
- **Initial State Distribution:** $P(z_1)$ - The probability distribution over initial states.

These components of the joint distribution of observed data and hidden states allow for explicit computation of likelihoods and probabilistic inference, making our proposed model an explicit generative model. Our choice of explicit generative models aligns with our overall goal due to the following reasons :

- **Handling missing data :** Explicit generative models are particularly adept at managing missing data by specifying a model for the underlying data distribution. Implicit models, while they can handle missing data, often do so less

effectively as they might not model the data distribution as comprehensively.

- **Interpretability** : The parameters in explicit models often have meaningful interpretations, such as emissions probabilities in Hidden Markov Models (HMMs) representing the likelihood of generating features given the latent state. This transparency helps clinicians trust and understand the model’s decisions. Implicit models might lack this transparency as they do not explicitly define the data distribution, making it harder to understand the underlying mechanisms.
- **Likelihood estimation** : Explicit generative models allow for direct likelihood estimation, which can be useful for model evaluation and comparison. Likelihood provides a measure of how well the model fits the observed data, facilitating model selection and tuning. Implicit models typically do not provide likelihood estimates, making it more challenging to quantitatively assess model performance.

4.2 Methods

4.2.1 Problem formulation

We represent the measurements available from each patient stay in a hospital as a regularly-spaced time series, unfolding in discrete timesteps indexed by integer $t \in \{1, 2, \dots, T\}$. In later experiments, each timestep represents one hour. In general, the length T of each time series may vary across patient stays.

For each patient stay, we observe a sequence of feature vectors across time: $\mathbf{x}_{1:T} = [\mathbf{x}_1, \dots, \mathbf{x}_T]$. The vector $\mathbf{x}_t$ represents a feature vector of possibly-missing-values of size D . Each index $d \in \{1, 2, \dots, D\}$ represents a specific *measurement* (e.g. vital sign or lab test) of interest. Entry x_{td} represents either (i) the numerical value of measurement d at time t , when available or (ii) a symbol NAN indicating the intended vital or lab was not measured at time t . Let $\mathbf{o}_{1:T} = \mathbf{o}_1, \dots, \mathbf{o}_T$ denote the set of indices of measurements that are observed (not missing) across all timesteps

$1:T$.

Each patient stay can be associated with a classification outcome y taking one of 2 possible values (0 or 1) for binary classification or one of C possible values, denoted $y \in \{1, 2, \dots, C\}$ for ordinal regression. Adaptation of our methods to real-valued y (aka regression) is straightforward conceptually; and in-fact has been shown to perform well in predicting soldier readiness from time-series of gait and stride sensory data [Rath et al., 2023]. Further illustration of evaluating PC-HMM regression models is left to future work.

Our goal is to build a probabilistic classifier that can map any given feature time series $\mathbf{x}_{1:T}$ (which may have arbitrary missingness) to a probability vector over the C -classes of the outcome. The training or learning problem is then how to optimize such a classifier given a finite training set. We consider two distinct learning tasks, differing in assumptions about what training data is available.

Supervised time-series classification with missingness. For this task, we assume for model development we have access to a fully-labeled training dataset $\mathcal{D}^L$ of size N , where each element is a pair $(\mathbf{x}_{1:T}, y)$ representing the time series of measurements and corresponding class outcome. In our target applications, each sequence $\mathbf{x}_{1:T}$ may have arbitrary missingness patterns during training or at prediction time on new patient stays.

Semi-supervised time-series classification with missingness. For this task, we assume for model development we have access to both a (small) labeled training dataset $\mathcal{D}^L$ of size N and a (large) unlabeled training set $\mathcal{D}^U$ of size M . Each element in $\mathcal{D}^L$ is a pair $(\mathbf{x}_{1:T}, y)$ of features and outcome representing one patient stay. Each element in $\mathcal{D}^U$ is just the features $\mathbf{x}_{1:T}$ for a patient stay, without any known label. Again, we assume $\mathbf{x}_{1:T}$ might have arbitrary missingness when training or predicting.

4.2.2 Model : Prediction Constrained HMMs for Semi-supervised Classification and Ordinal Regression

Here, we review the HMM generative model, how its resulting belief-state vector representations can be used for prediction, and how prediction-constrained training can be used to balance discriminative and generative goals in settings with small labeled sets and feature missingness.

HMM generative model. The HMM is defined by a fixed number of states K (set as a hyperparameter), parameters governing *transitions* (outgoing probability vectors for all K states $\{\pi_k\}_{k=1}^K$ and the initial state π_0) and *emission* parameters $\{\phi_k\}_{k=1}^K$. Given (π, ϕ) , the HMM defines a model for a sequence of regularly-spaced times of size T , where the random variables are the observable D -dimensional features over time $X \in \mathbb{R}^{T \times D}$ and the hidden discrete states $Z = [z_1, \dots, z_T]$. The model $p(X, Z|\pi, \phi)$ decomposes as

$$p(Z|\pi) = \text{Cat}(z_1|\pi_0) \prod_{t=2}^T \text{Cat}(z_t|\pi_{z_{t-1}}), \quad p(X|Z, \phi) = \prod_{t,d} \mathcal{N}(x_{td}|\mu_{z_t,d}, \sigma_{z_t,d}^2). \quad (4.1)$$

In words, we draw Z from a first-order Markov model, where each z_t indicates one of the K states. Then, we generate each feature x_{td} given z_t via a state-specific emission model that *factorizes* over feature dimensions d and times t . Elegantly, this factorization assumption allows exact computation of the marginal of observed features under *any missingness pattern* (our specific model falls under the missing completely at random (MCAR) scenario since we do not explicitly assume that missingness depends on observed or unobserved data). By applying the sum rule, the probability of the observed entries of X , denoted $X^\mathcal{O}$, given Z is $\prod_{t,d \in \mathcal{O}} \mathcal{N}(x_{td}|\mu_{z_t,d}, \sigma_{z_t,d}^2)$. Naturally, this factorization is a result of assuming independence between the D input dimensions. We avoided estimating a full covariance matrix because when data is missing, estimating the parameters of a non-diagonal covariance matrix becomes more difficult and less reliable. The presence of missing data complicates the calculation of covariance since not all data points contribute to all parts of the covariance

calculation. This can lead to biased or inaccurate estimates of the covariance parameters. Additionally, estimating the full covariance matrix increases the computational challenges.

Computing beliefs. We can obtain useful representations from the HMM even though we never observe Z by computing for each timestep t and state k the posterior belief of state occupancy: $b_{tk}(X^{\mathcal{O}}) \triangleq p(z_t = k | X^{\mathcal{O}})$. Efficient dynamic programming achieves this in $O(TK^2)$ runtime. From an interpretability perspective, the beliefs provide information about the state of the patient at every time-point. In figure 4.4 and 4.5, we show how the beliefs associated with ‘risk’ states are useful in providing interpretability at a patient-level and at a population level.

Predicting binary outcomes from beliefs. Beliefs represent sufficient statistics for the HMM’s latent state at each timestep, and thus could be useful representations for classification. We condense all T belief vectors into an average occupancy vector $\bar{b} = [\bar{b}_1, \dots, \bar{b}_K]$, where $\bar{b}_k = \frac{1}{T} \sum_{t=1}^T b_{tk}$. We can then build a simple linear classification model:

$$p(y = 1 | X) = \sigma(\eta^T \bar{b}(X)) \quad (4.2)$$

where $\eta \in \mathbb{R}^K$ are the weight parameters and $\sigma(r) = \frac{1}{1+e^{-r}}$ is the logistic sigmoid function for real-valued inputs r . Besides average pooling, several alternative strategies have emerged in recent research as effective means to transform beliefs for downstream prediction. In the work by Sotoodeh and Ho [2019], feature representations are derived by concatenating beliefs from the first and last time-points, enabling the capture of patient stay variability. Additionally, Hope et al. [2021] demonstrate that applying a convolutional transformation to the belief sequence, followed by localized max pooling, leads to improved performance in human activity recognition tasks. While these alternative strategies exhibit promise, our own experiments focusing on mortality and length of stay prediction have conclusively shown that average pooling alone is sufficient to achieve competitive performance when compared to other baselines.

Predicting ordinal outcomes from beliefs We wish to model the label $y \in \{1, \dots, C\}$ given an K -dimensional feature vector $\bar{b}(x)$, by mapping features to a scalar location $g \in (-\infty, +\infty)$ on the real line via a generalized linear model $g(x) = \eta^T \bar{b}(x)$ with weights $\eta \in \mathbb{R}^K$. A *deterministic* model might then map this location to a class via a set of ordered thresholds $\theta_0 < \theta_1 < \dots < \theta_C$:

$$p(y = c|x) = \begin{cases} 1 & \text{if } \theta_{c-1} < g(x) \leq \theta_c \\ 0 & \text{otherwise} \end{cases} \quad (4.3)$$

where we fix boundaries at $\theta_0 = -\infty$ and $\theta_C = +\infty$. But the above would be flat almost everywhere as a function of parameters η, θ , making gradient-based learning difficult.

Cumulative link models assume a smoother likelihood:

$$p(y=c|x) = H\left(\frac{\theta_c - g(x)}{\sigma_{scale}}\right) - H\left(\frac{\theta_{c-1} - g(x)}{\sigma_{scale}}\right) \quad (4.4)$$

where H is a cumulative distribution function (CDF) of a location-scale univariate distribution $\mathcal{H}$ over the reals, and $\sigma_{scale} > 0$ is a scale parameter. Because H as a CDF is a smooth increasing function according to Eq. (4.4), increasing the value of g gradually across any threshold θ_c leads to intuitively gradual changes in output probability (see Fig. 4.1), not the sudden change across boundaries of Eq. (4.3).

Semi-supervised learning of HMMs via prediction-constrained training

Now that our generative and discriminative model components are defined, we wish to *estimate the parameters* of the model from a labeled training set $\mathcal{D}^L$ and an (optional) unlabeled set $\mathcal{D}^U$.

The first approach, simple and commonly used [Sotoodeh and Ho, 2019], is two-phase training that first fits standard HMM parameters π, μ, σ to all feature data (labeled and unlabeled), then fits the prediction parameters η, θ on the labeled set

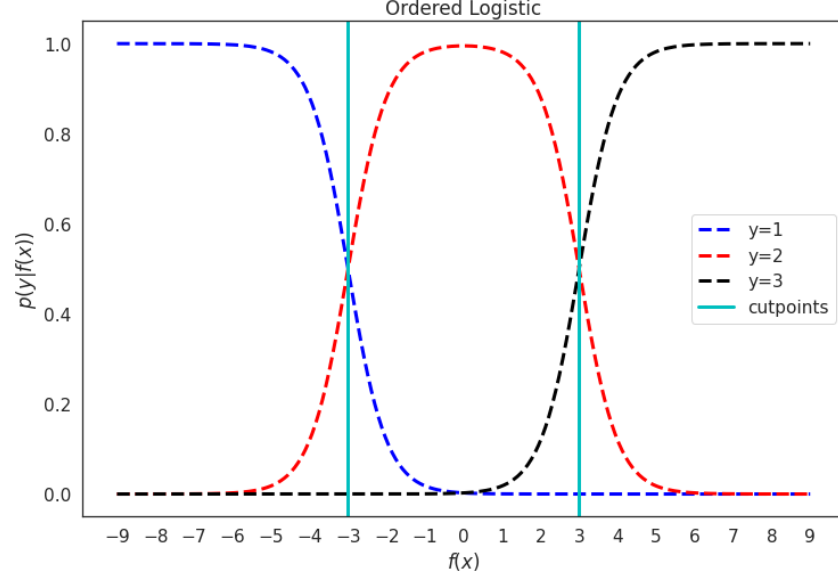


Figure 4.1: Visualization of likelihood function for ordinal regression with 3 classes using the logistic distribution to simulate latent function noise

only.

$$\text{phase 1: } \hat{\pi}, \hat{\mu}, \hat{\sigma} \leftarrow \arg \max_{\pi, \mu, \sigma} \sum_{\mathbf{X}^{\mathcal{O}} \in \mathcal{D}^L \cup \mathcal{D}^U} \log p(\mathbf{X}^{\mathcal{O}} | \pi, \mu, \sigma), \quad (4.5)$$

$$\text{phase 2: } \hat{\eta}, \hat{\theta} \leftarrow \arg \max_{\eta, \theta} \sum_{\mathbf{X}^{\mathcal{O}}, y \in \mathcal{D}^L} \log p(y | \mathbf{X}^{\mathcal{O}}, \eta, \theta, \hat{\pi}, \hat{\mu}, \hat{\sigma}).$$

In phase 2, prediction parameters are estimated to maximize the label-given-feature likelihood, which is Eq. (4.2) for binary classification (where θ is unneeded) or Eq. (4.4) for ordinal regression. Both objectives can be computed via HMM dynamic programming routines and are amenable to modern automatic differentiation, enabling estimating solutions via gradient descent algorithms. For such algorithms, constrained parameters such as initial state probabilities π_0 , state transition probabilities π_k , and Gaussian emission variances σ are reparameterized into the unconstrained real space so that all gradient updates yield valid parameters. Phase 1 can also be solved with expectation-maximization [Ghahramani, 2001].

The potential downside to the above approach is that the HMM is trained in an *unsupervised* fashion. Class labels y do not inform the values of π, μ, σ . Alternatively,

we can maximize the joint supervised likelihood of features and any available labels:

$$\pi, \mu, \sigma, \eta, \theta \leftarrow \arg \max_{\mathbf{X}^{\mathcal{O}}, y \in \mathcal{D}^L} \sum \log p(\mathbf{X}^{\mathcal{O}}, y | \eta, \theta, \pi, \mu, \sigma) + \sum_{\mathbf{X}^{\mathcal{O}} \in \mathcal{D}^U} \log p(\mathbf{X}^{\mathcal{O}} | \pi, \mu, \sigma) \quad (4.6)$$

Again, both terms (and their gradients) can be computed in straightforward fashion, as in the two-stage method above. The main issue with this approach is that while labels can inform the values of the HMM parameters π, μ, σ , the objective does not encode our practical goal to prioritize predicting labels from features well even when we have few labels. This deficiency is reflected in our later results in Figure 4.3, where this “supervised” approach to HMM training is hardly better than the two-stage unsupervised-then-predict method when we have few labels ($N \ll M$).

Recognizing the limitations of these methods, we arrive at our proposed *prediction-constrained* (PC) objective, which directly encodes our goals: first and foremost, achieve label-from-feature prediction better than some threshold ϵ , while also learning a good generative model if possible (to improve handling of missingness). Formally, we wish to estimate parameters by solving the constrained objective

$$\begin{aligned} \pi, \mu, \sigma, \eta, \theta \leftarrow \arg \max_{\mathbf{X}^{\mathcal{O}} \in \mathcal{D}^L \cup \mathcal{D}^U} & \sum \log p(\mathbf{X}^{\mathcal{O}} | \pi, \mu, \sigma), \\ \text{subj. to:} & \sum_{\mathbf{X}^{\mathcal{O}}, y \in \mathcal{D}^L} \log p(y | \mathbf{X}^{\mathcal{O}}, \pi, \mu, \sigma, \eta, \theta) \geq \epsilon. \end{aligned} \quad (4.7)$$

Directly handling the constraint here is difficult, so we transform the problem to be more amenable to modern gradient-based learning. By applying the theory of Langrange multipliers [Bertsekas, 1996], we can form an equivalent unconstrained PC objective

$$\pi, \mu, \sigma, \eta, \theta \leftarrow \arg \max_{\mathbf{X}^{\mathcal{O}} \in \mathcal{D}^L \cup \mathcal{D}^U} \sum \log p(\mathbf{X}^{\mathcal{O}} | \pi, \mu, \sigma) + \lambda \sum_{\mathbf{X}^{\mathcal{O}}, y \in \mathcal{D}^L} \log p(y | \mathbf{X}^{\mathcal{O}}, \pi, \mu, \sigma, \eta, \theta). \quad (4.8)$$

For each tolerance ϵ , there exists a scalar $\lambda > 0$ such that the maximizer of Eq. (4.8) is an optimum of Eq. (4.7). Scalar λ thus plays an analogous role as ϵ in the constrained objective; to achieve a higher prediction quality ϵ , we would solve the above with a larger λ . In the special case of $\lambda = 1$, we recover the earlier supervised objective in Eq. (4.6). We intend usually that $\lambda \gg 1$, which encourages better predictions of y from x , trading off the quality of generative modeling of x when necessary.

PC-HMM Initialization: We initialize the HMM’s emission’s parameters using K-means clustering. For ordinal regression, we initialize the emission’s parameters for each class separately and then aggregate the learned clusters by retaining the states that are maximally distant from each other. We set the HMM’s initial state probabilities to a uniform with probability $1/K$. For the transition distribution, we initialize such that self transitions probabilities (diagonal elements of the transition matrix) are 0.5, while the off-diagonal entries are $0.5/(K - 1)$. We initialize the predictor weights by training a supervised model (logistic regression for binary classification or ordered logistic for ordinal regression) using the initial beliefs as features. We additionally apply L2 regularization on the predictor weights to prevent overfitting.

4.2.3 Justifying Usage of The Prediction Constrained (PC) Framework

The PC framework is built to model tasks containing both labelled and unlabelled examples. In case of time-series health record datasets, unlabelled examples include medical records (including vitals, lab measurements, medications etc.) of patient stays for whom outcome labels may not be available (for example, due to the high cost of obtaining clinician annotated disease labels). In contrast, labelled examples include medical records of patient stays for whom clinician annotated labels exist. The goal of the PC framework is to improve the predictive performance on labelled examples (over a fully supervised approach) by utilizing the unlabelled examples. This is done by maximizing the generative performance on both the labelled and unlabelled examples. While simultaneously adhering to performance constraints on

the labelled set.

To formulate the core principle of the PC framework and highlight it's benefits, we first consider a more generic training approach, *regularized joint maximum likelihood optimization*. We consider a dataset D containing labelled examples $D^L = \{x_n, y_n\}_{n=1:N^L}$ and unlabelled examples $D^U = \{x_n\}_{n=1:N^U}$. Additionally, we assume h_n is a hidden variable, such that the generative process is as follows : (1) Generate h_n from prior distribution parameterized by ξ^h ; (2) Generate x_n given h_n from a distribution parameterized by ξ^x ; (3) Generate y_n given x_n and h_n parameterized by ξ^y . The joint distribution over x_n, y_n, h_n is as follows :

$$p(x_n, y_n, h_n | \xi^h, \xi^x, \xi^y) = p(h_n | \xi^h) p(x_n | h_n, \xi^x) p(y_n | x_n, h_n, \xi^y) \quad (4.9)$$

Regularized maximum likelihood minimizes the following objective and obtains the optimal parameters ξ^{ML} .

$$\xi^{ML} = \min_{\xi=\{\xi^h, \xi^x, \xi^y\}} -\sum_n \log p(x_n, y_n | \xi) + R(\xi) \quad (4.10)$$

where $R(\xi)$ is the regularizer on the model parameters. The main issue with regularized maximum likelihood is that it assumes a symmetric relationship between x_n and y_n . In other words, jointly maximizing the likelihood of x_n and y_n does not clearly highlight the practitioner's goal of predicting y_n given x_n , rather than x_n given y_n . To overcome this issue, the PC framework clearly specifies the need to learn a generative model for data x that simultaneously achieves some threshold performance ϵ while predicting y from x with the following objective.

$$\begin{aligned} \min_{\xi=\{\xi^h, \xi^x, \xi^y\}} & -\sum_{n \in D^L \cup D^U} \log p(x_n | \xi^h, \xi^x) + R(\xi), \\ \text{subj. to:} & -\sum_{n \in D^L} \log p(y_n | \xi^h, \xi^x, \xi^y) \leq \epsilon \end{aligned} \quad (4.11)$$

By applying the theory of Langrange multipliers, the unconstrained PC objective is :

$$\xi^{PC} = \min_{\xi=\{\xi^h, \xi^x, \xi^y\}} -\sum_{n \in D^L \cup D^U} \log p(x_n | \xi^h, \xi^x) - \lambda \sum_{n \in D^L} \log p(y_n | \xi^h, \xi^x, \xi^y) + R(\xi) \quad (4.12)$$

The PC objective clearly prioritizes predictions of y_n from x_n by setting $\lambda > 1$ and eliminates the symmetric assumption between x_n and y_n . This is also justified empirically in Figures 4.3 and 4.7, where we see that models trained with the PC objective ($\lambda \gg 1$) always outperform models trained with maximum likelihood ($\lambda = 1$, shown as supervised HMM), and models where the supervised and unsupervised components are trained separately in a 2 step process ($\lambda = 0$, shown as unsupervised HMM + predictor). In the following sub-sections, we show how the PC framework is applied to time-series data by choosing a HMM as the generative component. We further show how missing data can be handled elegantly without the need for imputation, and how missing labels can be handled via the semi-supervised extension of the PC framework for clinical prediction tasks involving binary classification as well as ordinal regression.

4.2.4 Baseline neural nets for fully-supervised classification with missingness

We now consider alternative deep models intended to produce effective predictions for clinical time series, even with some missingness, but that can only make use of labeled datasets. These approaches cannot easily benefit from an extra large unlabeled dataset $\mathcal{D}^U$. We focus specifically on two methods in the recurrent neural net tradition that are both well-known and competitive: the Gated Recurrent Unit with Decay (GRU-D) [Che et al., 2018b] and Bidirectional Recurrent Imputation Time Series (BRITS) [Cao et al., 2018b], each summarized below.

GRU-D : Gated recurrent Unit with decay model [Che et al., 2018b] developed the GRU-D to adapt the standard GRU (Gated Recurrent Unit) architecture

by [Cho et al., 2014b] in order to specifically handle clinical time series data with missing values and long time delays between non-missing values. GRU-D extends the GRU by producing a complete feature vector at every time step, imputing any missing values between the last observation and a population mean via a learnable, variable-specific decay mechanism informed by the time since last observation. Another similar decay mechanism informs the GRU-D’s continuous-valued hidden state vector. Ultimately, like any recurrent neural net, the GRU-D can be viewed as a deterministic mapping of input time series $\mathbf{X}$ of length T to a representation vector h_T at the final timestep. Following [Che et al., 2018b], a fully-connected output layer maps h_T to probabilities over the C possible classes.

BRITS : Bidirectional Recurrent Imputation Time Series model [Cao et al., 2018b] designed BRITS as a bidirectional RNN that jointly performs imputation and classification/regression. It generates forward and backward hidden state representations, concatenates them into per-timestep representations h_t , and creates a sequence-level representation through mean pooling, followed by a fully-connected layer to produce class probabilities. The BRITS model handles missing data by using bidirectional recurrent neural networks that incorporate both past and future information, leveraging temporal correlations to impute missing values while considering the entire sequences, but its reliance on future data makes it unsuitable for real-time applications.

In our implementation, for both GRU-D and BRITS weight parameters are trained via gradient descent exclusively on the labeled training set $\mathcal{D}^L$ to minimize cross entropy. BRITS also includes a reconstruction and imputation consistency loss. For both methods, key hyperparameters like network size and learning rate settings are tuned on the validation set of each task. See table 4.9 for details.

4.2.5 Baseline neural nets for semi-supervised time series classification without missingness

Many works have aimed to develop deep classifiers for SSL scenarios without abundant labeled data but plenty of unlabeled data. In this work, we focus on two strong

baselines from the deep SSL literature, MixMatch [Berthelot et al., 2019b] and FixMatch [Sohn et al., 2020]. MixMatch recently delivered the most reliable accuracy gains (even over more recent methods) at a benchmark of self-supervised and semi-supervised methods for medical images [Huang et al., 2023]. We adapt both methods to time series following [Goschenhofer et al., 2021]. Importantly, neither method natively handles time series with arbitrary missingness. Thus, all methods here assume missing values have been filled by forward filling imputation (for all times after the first observation of channel d) or mean imputation (for all other times).

Following the unified analysis of many SSL methods in [Zhu et al., 2022], these approaches train a deep network f_w with weights w to minimize a two-term loss:

$$w^* \leftarrow \arg \min \sum_{X, y \in a(\mathcal{D}^L)} \ell(y, f_w(X)) + \lambda \sum_{X \in a(\mathcal{D}^U)} \ell^U(y'(X), f_w(X)) \quad (4.13)$$

Here, $\ell(\cdot, \cdot)$ is a standard classification loss (e.g. binary cross entropy) and ℓ^U a corresponding unlabeled set loss, $\lambda > 0$ is a tradeoff hyperparameter, $y'(X)$ is a labeling transformation, and $a(\cdot)$ represents an (optional) data augmentation transformation. Below, we describe how both MixMatch and FixMatch fit this framework via distinct realizations of augmentations a and loss ℓ^U .

Our implementation of both MixMatch and FixMatch uses a Gated Recurrent Unit (GRU) [Cho et al., 2014b] architecture to obtain a vector representation for the entire time series. We make predictions using representations from the last time step of each sequence. For both methods, weights w are estimated using minibatch gradient descent with the Adam [Kingma and Ba, 2014a] optimizer, starting with the unlabeled loss weight λ at zero and gradually increasing it [Berthelot et al., 2019a, Tarvainen and Valpola, 2017]. Key shared hyperparameters like λ , GRU network size, and learning rate, as well as method-specific hyperparameters, are tuned on the validation set.

MixMatch for time series. MixMatch is a semi-supervised learning algorithm that improves model performance by combining data augmentation, pseudo labels produced by the classifier, and a consistency loss to effectively utilize both labeled

and unlabeled data. To adapt MixMatch [Berthelot et al., 2019b] to time series, we follow Goschenhofer et al. [Goschenhofer et al., 2021]. For each input time series, we have all missing values imputed and then apply z-score transformations to standardize features. Next, we apply two kinds of augmentation: add Gaussian noise $\mathcal{N}(0, \epsilon^2)$ and apply MixUp [Zhang et al., 2017], which interpolates between random pairs of features $\mathbf{X}$ and their labels (or pseudo labels) to create synthetic examples. We sample each pair’s interpolation ratio from the distribution $\text{Beta}(\alpha, \alpha)$. MixMatch defines loss ℓ^U to enforce label consistency between the model’s prediction on an unlabeled instance and the average temperature-sharpened probability vector from several augmentations of that instance. The key hyperparameters for MixMatch are scalars ϵ and α .

FixMatch for time series. FixMatch [Sohn et al., 2020] combines weak and strong augmentations to create diverse versions of unlabeled data, enabling the model to learn robust features by consistently predicting the same labels across these variations. Weak augmentations provide stable pseudo-labels, while strong augmentations introduce robustness by challenging the model with heavily altered data. To adapt FixMatch to time series, we add random Gaussian noise sampled from $\mathcal{N}(0, 0.01)$ for weak augmentations and from $\mathcal{N}(0, 1)$ for strong augmentations. Unlike [Goschenhofer et al., 2021], who use RandAugment [Cubuk et al., 2020], we only add Gaussian noise since time and magnitude warping hurt performance in our experiments. Pseudo-labels for weakly augmented examples are retained if the model produces high-confidence predictions, with thresholds of 0.6 and 0.8 explored. FixMatch’s unlabeled loss ℓ^U uses these pseudo-labels as targets for strongly augmented versions of the same examples, ensuring consistent predictions across augmentations and leveraging high-confidence predictions to enhance model robustness and generalization.

4.2.6 Justification of the $O(TK^2)$ belief computation runtime for HMMs

The complexity largely stems from the number of states and the transitions between these states considered over the length of the observation sequence in an algorithm

termed as Forward Backward Algorithm.

The forward algorithm computes the probability of observing the sequence up to time t and being in state k . This is known as the forward probability, denoted as $\alpha_t(k)$. The recursion formula for $\alpha_t(k)$ is given by:

$$\alpha_t(k) = \left(\sum_{j=1}^K \alpha_{t-1}(j) a_{jk} \right) b_k(o_t)$$

Where:

- $\alpha_{t-1}(j)$ is the forward probability of being in state j at time $t - 1$.
- a_{jk} is the transition probability from state j to state k .
- $b_k(o_t)$ is the probability of observing o_t given state k (the emission probability).

Computational Steps:

- For each state k at each time t , calculate $\alpha_t(k)$.
- For each k , the sum is over all K possible states j , requiring K multiplications and $K - 1$ additions.
- Thus, for each k and each t , the computation is $O(K)$.
- Since there are K states to consider at each time step, the complexity for each t is $O(K^2)$.

Similarly, the backward algorithm computes the probability of the future observations given state k at time t . This is known as the backward probability, denoted as $\beta_t(k)$. The recursion formula for $\beta_t(k)$ is given by:

$$\beta_t(k) = \sum_{j=1}^K a_{kj} b_j(o_{t+1}) \beta_{t+1}(j)$$

Where:

- $\beta_{t+1}(j)$ is the backward probability at state j at time $t + 1$.

- a_{kj} is the transition probability from state k to state j .
- $b_j(o_{t+1})$ is the emission probability at state j for observation o_{t+1} .

Since both the forward and backward algorithms require $O(K^2)$ operations at each of the T time steps, the overall computational complexity for running either algorithm is $O(TK^2)$. This describes the amount of computational work needed to process an entire sequence with respect to the number of states and the length of the sequence, making clear the impact of the state space size and the sequence length on the computational demand.

4.3 Results

4.3.1 Toy Data Experiment

The 2-D toy data (represented as $x = \{x_1, x_2\}$) is generated from a Markov model with 4 states (states 0, 1, 2, 3) with the means of the emission distributions located at $(0, -1)$, $(8, -1)$, $(13, 0)$ and $(24, 0)$. The variance in x_1 is 0.3 and variance in x_2 is 1 for each of the clusters. Each sequence is initialized at state 0 and either remains in the current state or transitions to the next adjacent state with equal probability. Once state 2 is reached, it immediately transitions to state 3 in the next time-step and stays there for the remaining time-steps with probability 1. We simulate 350 sequences, each of length $T = 8$ from this model. A sequence is labeled as 1 if it enters the highlighted black box highlighted in figure 4.2 at any time-point. At each time-point and dimension, data is missing with a fixed probability (ρ). We vary ρ on the right plot of figure 4.2 and report classification performance against several baselines.

Findings : We analyze a toy dataset of many sequences generated by an HMM with $K = 4$ Gaussian clusters in a $D = 2$ feature space. The 2D features are illustrated in Fig. 4.2: Each sequence is labeled $y = 1$ only if any timestep’s feature x_t enters the black square. Fig. 4.2 (left) shows that even at 40% missingness (i.i.d. across features and timesteps) the PC-HMM with $K = 6$ learns separate states that cover

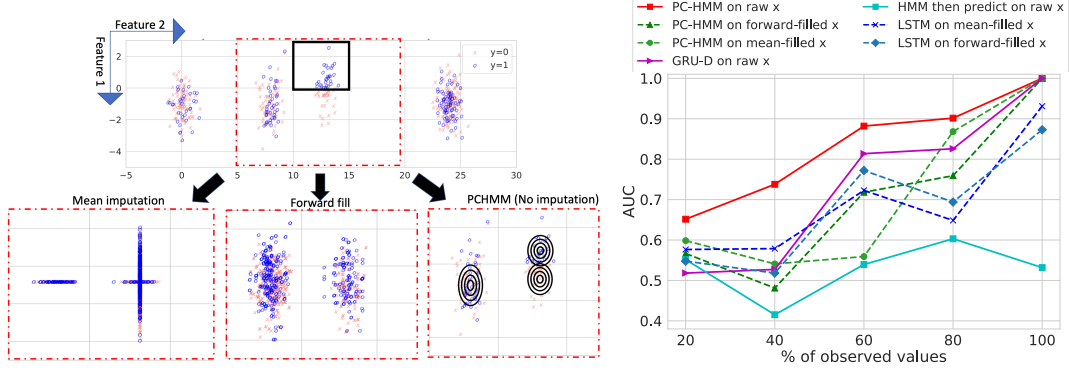


Figure 4.2: Demonstration of proposed PC-HMM on an “easy” binary classifier task where naive imputation fails. **Top left:** Visualization of all observed 2D features from 350 sequences (x-axis: feature 1 value, y-axis: feature 2), colored by labels and collapsed over time. Each sequence ($T=8$) comes from a Markov model that generates features from the 4 clusters shown in this 2D feature space. A sequence’s class label depends on if *any observation* lands in the black box (yes means $y = 1$, no means $y = 0$). Without missingness, the two classes are clearly separable: compare black box to region below it. **Bottom left:** We sample features as missing with 40% probability i.i.d. across timesteps and features. We then apply mean-filling or forward-filling: each panel shows 2D values *after* imputation. Either strategy clearly destroys class separability. Our PC-HMM does not need imputation, and even at 40% missingness can learn to separate the black box from the region below it with separate Gaussian emission densities (black ellipsoids). **Right:** AUROC vs. percentage of feature values observed. Across every tested percentage, our PC-HMM with compact state space (only 6 states, 80 total parameters) can outperform GRU-D or LSTM models even with many more parameters.

the black box and its complement, and thus are *discriminative*. In contrast, both mean-filling and forward-filling imputation destroy the separability of red and blue classes. In Fig. 4.2 (right), at all missingness levels the PCHMM delivers better AUROC than alternatives, including an LSTM and the GRU-D. This dataset is class-balanced, thus AUROC is an appropriate metric. Later mortality prediction tasks are imbalanced, so we use area under the precision recall curve (AUPRC) instead. All PC-HMM results, including “with mean-filled/forward-filled” baselines, assume diagonal covariances.

4.3.2 Real Data Experiments

Below, we define and motivate two concrete prediction tasks that we pursue in our experimental evaluations. Both tasks assume access to the first full 24 hour period

of observation in critical care, and predict a future outcome. To avoid causal leakage and ensure predictions could be actionable, in all training and evaluation data we only examine patient stays that last at least 30 hours. This prevents the classifier from learning a shortcut signal that is already quite obvious to the care team (e.g. lots of palliative morphine before death).

In-ICU Mortality. Given a few dozen vital sign and lab measurements from the first 24 hours of a patient stay, we wish to predict whether that patient will later die in the ICU ($y = 1$) or not ($y = 0$). Classifiers that can effectively predict an increased risk of mortality several hours ahead could be useful as early warning systems [Gao et al., 2020, Harutyunyan et al., 2019b, Purushotham et al., 2018b, Rajkomar et al., 2018]. Several of these works have pursued classifiers for in-ICU mortality. Our work is differentiated because our probabilistic approach does not rely on any form of imputation or iterative refinement of missing values, which are common in health record datasets. We further evaluate performance where labels are limited. While we acknowledge that a clinical outcome like in-ICU mortality outcome would rarely be missing, our goal is to use this outcome as a proxy to demonstrate the utility of our model for disease outcomes that require significant clinical expertise to annotate. To assess in-ICU mortality, we focus on area under the precision-recall curve (AUPRC) because it accounts for several limiting factors in clinical EHR prediction such as severe class imbalance, need for high performance on the positive class at low false alarm rates.

4.3.3 Data

MIMIC-IV We first study data from MIMIC-IV Johnson et al. [2020] to predict length of stay for 52,354 de-identified ICU patient-stays from one hospital. We preprocess the dataset to contain 39 time varying measurements of labs and vitals and 2 demographics (see tables 5.2 and 5.3 for full feature list and missing data information). We selected these labs and vitals after reviewing existing literature pursuing prediction tasks with clinical time-series [Harutyunyan et al., 2019b, Purushotham et al., 2018b]. Each time varying measurement is transformed into hourly buckets

by taking the closest to the nearest earlier hour. If no data existed within an hourly bucket, we leave it as missing. We remove patient stays less than 30 hours to avoid label leakage. Additionally, we ensure that the subjects are split into train/valid and test based on patient-IDs.

Table 4.2: MIMIC-IV Dataset Split and Statistics

Dataset	Split	Stays	Patients	Fraction of Stays			In-ICU mortality
				LOS ≥ 3 days	LOS ≥ 7 days	LOS ≥ 11 days	
MIMIC-IV	Train	33,506	27,084	0.46	0.16	0.08	0.08
	Valid	8,377	7,821	0.46	0.16	0.08	0.08
	Test	10,471	9,673	0.47	0.17	0.08	0.08

eICU We also study the eICU dataset Pollard et al. [2018a] (details in section 4.3.3), containing 111,879 de-identified ICU patient-stays. We follow the same pre-processing steps as MIMIC-IV to aggregate measurements into hourly buckets, and split the data into train/valid and test, and remove stays less than 30 hours. We pre-process the dataset to contain 39 time varying measurements of labs and vitals and 2 demographics.

Table 4.4: eICU Dataset Split and Statistics

Dataset	Split	Stays	Patients	Fraction of Stays			In-ICU mortality
				LOS ≥ 3 days	LOS ≥ 5 days	LOS ≥ 7 days	
eICU	Train	72,947	62,322	0.43	0.21	0.13	0.06
	Valid	18,237	17,429	0.43	0.22	0.12	0.05
	Test	22,797	21,587	0.43	0.21	0.13	0.05

4.3.4 Mortality Prediction Results

Fig. 4.3 shows each method’s test-set AUPRC (where $y = 1$ means death) as label availability p increases. Across all tested percentages p , the PC-HMM is competitive with deep alternatives, including BRITS and GRU-D as well as deep SSL such as

Table 4.1: MIMIC-IV features

	10%	median	90%	missing_rate
Heart Rate	63.00	83.00	109.00	0.00
Respiratory Rate	13.00	18.00	26.00	0.00
O2 saturation pulseoxymetry	93.00	97.00	100.00	0.00
Non Invasive Blood Pressure systolic	93.00	116.00	148.00	0.08
Non Invasive Blood Pressure diastolic	47.00	63.00	85.00	0.08
Temperature Fahrenheit	36.22	36.83	37.61	0.07
Height (cm)	155.00	170.00	183.00	0.98
Respiratory Rate (Total)	14.00	18.00	27.00	0.62
Potassium (serum)	3.40	4.10	5.10	0.04
Sodium (serum)	132.00	138.00	144.00	0.04
Chloride (serum)	96.00	104.00	112.00	0.04
Hematocrit (serum)	23.60	30.30	38.90	0.05
Hemoglobin	7.70	10.10	13.10	0.05
Creatinine (serum)	0.60	1.00	3.10	0.04
Glucose (serum)	88.00	126.00	222.00	0.05
Magnesium	1.60	2.00	2.60	0.08
Phosphorous	2.20	3.50	5.50	0.11
Platelet Count	79.00	175.00	322.00	0.05
Glucose (whole blood)	97.00	136.00	200.00	0.77
Daily Weight	58.40	82.30	114.00	0.66
Absolute Neutrophil Count	4.55	9.30	20.54	1.00
Prothrombin time	11.60	14.20	23.60	0.21
Fibrinogen	127.00	216.00	462.00	0.80
PH (Arterial)	7.25	7.37	7.46	0.60
PH (Venous)	7.23	7.36	7.45	0.78
HCO3 (serum)	17.00	23.00	28.00	0.05
Arterial O2 pressure	75.00	132.00	334.00	0.60
Arterial CO2 Pressure	31.00	40.00	52.00	0.60
Lactic Acid	1.00	2.00	5.60	0.47
Albumin	2.20	3.10	3.90	0.75
Calcium non-ionized	7.30	8.30	9.30	0.12
C Reactive Protein (CRP)	2.00	40.70	213.40	0.98
ALT	11.00	33.00	395.00	0.59
AST	17.00	48.00	568.00	0.59
Direct Bilirubin	0.30	1.60	7.80	0.97
Total Bilirubin	0.30	0.80	5.10	0.60
Troponin-T	0.01	0.07	1.60	0.71
Venous CO2 Pressure	30.00	42.00	63.00	0.82
Age	40.00	65.00	84.00	0.00
is_gender_male	0.00	1.00	1.00	0.00
is_gender_unknown	0.00	0.00	0.00	0.00

Table 4.3: eICU features

Lab Test	10%	Median	90%	Missing Rate
temperature	35.80	37.18	38.21	0.91
sa02	93.25	97.17	100.00	0.04
heartrate	62.42	82.83	108.92	0.03
respiration	13.42	18.67	26.67	0.10
systemlicsystolic	94.50	118.25	150.40	0.78
systemlicdiastolic	44.25	57.45	75.42	0.78
ALT (SGPT)	12.00	29.00	178.00	0.56
AST (SGOT)	14.00	35.00	268.00	0.55
BUN	9.00	21.00	59.00	0.07
Hct	23.40	31.30	40.90	0.08
Hgb	7.70	10.30	13.60	0.09
MCH	26.70	30.00	32.70	0.16
PT	11.60	15.60	27.00	0.61
RBC	2.63	3.57	4.62	0.10
WBC x 1000	5.70	11.00	20.90	0.09
albumin	1.90	2.80	3.70	0.52
anion gap	6.00	11.00	18.00	0.26
calcium	7.20	8.20	9.20	0.09
chloride	96.00	105.00	113.00	0.07
creatinine	0.59	1.08	3.38	0.06
glucose	90.00	132.00	235.00	0.07
platelets x 1000	87.00	181.00	313.00	0.10
potassium	3.30	4.00	5.00	0.05
sodium	132.00	139.00	145.00	0.06
total bilirubin	0.30	0.70	2.50	0.56
total protein	4.60	5.80	7.00	0.55
uric acid	2.70	6.10	11.90	0.98
magnesium	1.50	1.90	2.50	0.42
bedside glucose	91.00	137.00	239.00	0.37
lactate	0.80	1.90	5.70	0.71
HCO3	16.20	22.90	31.00	0.62
pH	7.23	7.36	7.47	0.60
FiO2	21.00	45.00	100.00	0.58
Total CO2	17.20	24.00	35.30	0.85
fibrinogen	144.00	277.00	517.00	0.93
CRP	1.07	11.50	543.00	0.98
phosphate	1.90	3.30	5.50	0.62
direct bilirubin	0.10	0.30	2.40	0.91
troponin - T	0.02	0.14	1.96	0.96
age	38.00	65.00	82.00	0.03
gender_is_male	0.00	1.00	1.00	0.00

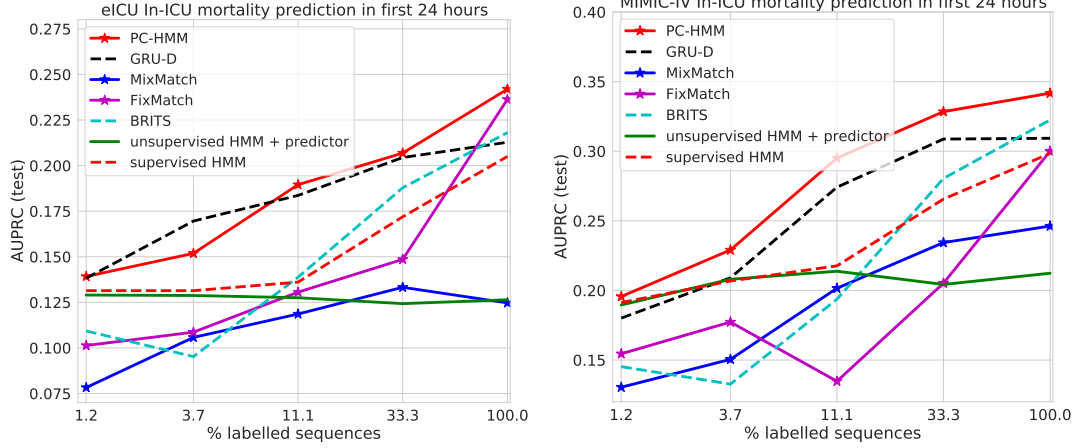


Figure 4.3: AUPRC (higher is better) versus amount of labeled data for in-hospital mortality early warning classifiers on two large EHR datasets (left: eICU, right: MIMIC-IV). X-axis: Percentage of all training sequences available with labels (SSL methods treat remaining sequences as unlabeled; other methods discard them). Y-axis: Area under precision-recall curve (AUPRC, higher is better). SSL methods, including our PC-HMM as well as MixMatch and FixMatch, learn from both labeled and unlabeled data. GRU-D, BRITS, and Random Forest use the labeled set only. The PC-HMM matches or beats the other models across all tested labeled set sizes, despite needing fewer parameters and only $1/10^{th}$ of the training time as the other models.

FixMatch and MixMatch on both MIMIC and eICU. Notably, the PC-HMM outperforms the baselines with the fewest parameters (see table 4.8), the least training time (see table 4.7) and the fewest number of jobs for hyperparameter search (see table 4.8). The full list of hyperparameters are listed in table 4.9

	Confidence Intervals for AUPRC for mortality prediction (100% labelled)				
	PC-HMM	GRU-D	MixMatch	FixMatch	BRITS
MIMIC-IV	0.341 (0.332, 0.351)	0.306 (0.301, 0.315)	0.212 (0.207, 0.218)	0.180 (0.177, 0.189)	0.322 (0.320, 0.325)
eICU	0.243 (0.239, 0.251)	0.211 (0.209, 0.214)	0.126 (0.120, 0.129)	0.124 (0.122, 0.129)	0.218 (0.213, 0.220)

Table 4.5: AUPRC with confidence intervals for mortality prediction with all labels known. The confidence intervals are obtained by bootstrapping on the test set. We resample 10 different subsets of the test set, and compute AUPRC for each subset. The median (5th, 95th) of the AUPRC across subsets are shown in the table. We find that the variance in prediction across subsets is fairly low and thus only show the performance on the full test set in figures 4.3 and 4.7

Beliefs aggregation	AUPRC for MIMIC-IV mortality prediction varying p (% labeled sequences)				
	$p = 1.2\%$	$p = 3.7\%$	$p = 11.1\%$	$p = 33.3\%$	$p = 100\%$
Average	0.19 (0.18, 0.20)	0.23 (0.22, 0.23)	0.29 (0.29, 0.30)	0.32 (0.32, 0.34)	0.34 (0.33, 0.35)
1 Conv layer + Max	0.14 (0.13, 0.14)	0.19 (0.19, 0.20)	0.27 (0.26, 0.28)	0.27 (0.27, 0.28)	0.28 (0.27, 0.29)
First + Last Step concat	0.15 (0.15, 0.16)	0.19 (0.18, 0.19)	0.20 (0.20, 0.21)	0.21 (0.21, 0.22)	0.25 (0.24, 0.25)

Table 4.6: Comparison of aggregation strategies of the HMM beliefs, before being passed as features to the predictor. We compare 3 strategies : (1)averaging beliefs across time (2) convolutional transformation of the belief sequence followed by max-pooling [Hope et al., 2021] and (3)concatenating the beliefs at the first and last time steps. Across all levels of missing labels, we observe that averaging the beliefs performs best [Sotoodeh and Ho, 2019].

4.3.5 Length of Stay Prediction

Figure 4.7 plots AUPRC as a function of available labeled data in an SSL setting for MIMIC-IV. Two major takeaways are apparent. First, ordinal regression CLMs outperform binary classifiers regardless of whether HMM or neural representations are used. Second, PC-HMM implementations outperform all other deep learning baselines. Compared to all other baselines, the ordinal regression PC-HMM not only achieves the best performance but also does so with the fewest trainable parameters and lowest training time.

4.3.6 Interpretation of Learned PCHMM States

Our PC-HMM framework enables us to interpret cohorts of vulnerable populations by visualizing the emission distributions for the states that have the highest predictor coefficients η_k . On eICU data, Fig. 4.4 shows our PC-HMM identifies states representing high blood urea nitrogen (BUN) and high creatinine (common indicators of kidney failure [Baha et al., 2021]) as the most vulnerable to in-ICU mortality. The model transitions to these ‘high-risk’ states as soon as high values of creatinine and BUN are observed in a patient who eventually dies. Additionally, we illustrate the interpretability of the PC-HMM trained to predict length of stay in Figure 4.5, highlighting that longer stays are associated with larger beliefs in the ‘risk state’.

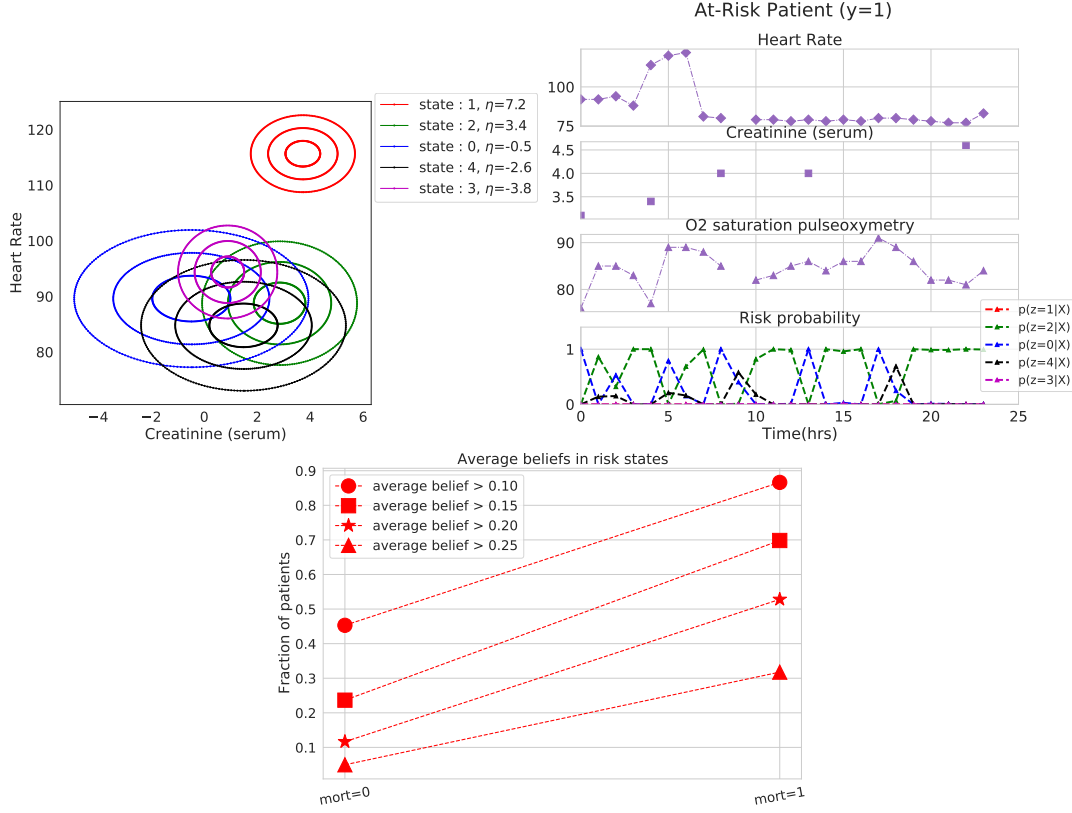


Figure 4.4: Interpretation of learned PC-HMM on eICU data. **Top Left:** Gaussian emission distributions of states and their η_k (predictor weights). The red and green states represent high risk (high η_k). High risk states seem to represent populations with high heart rate and high creatinine levels (above 1.9mg/dL), which could indicate kidney problems [Baha et al., 2021]. **Top Right:** For a patient labeled $y = 1$ (who eventually dies), high heart rate and high creatinine are observed throughout the stay, and thus beliefs b_t give highest probability to high risk states. The probability of state 2 (green) peak during segments of high creatinine and low oxygen saturation **Bottom:** We show the average beliefs (averaged over time) of normal patients and risk patients. Clearly, patients at risk spend more time in the risk state compared to the normal patients.

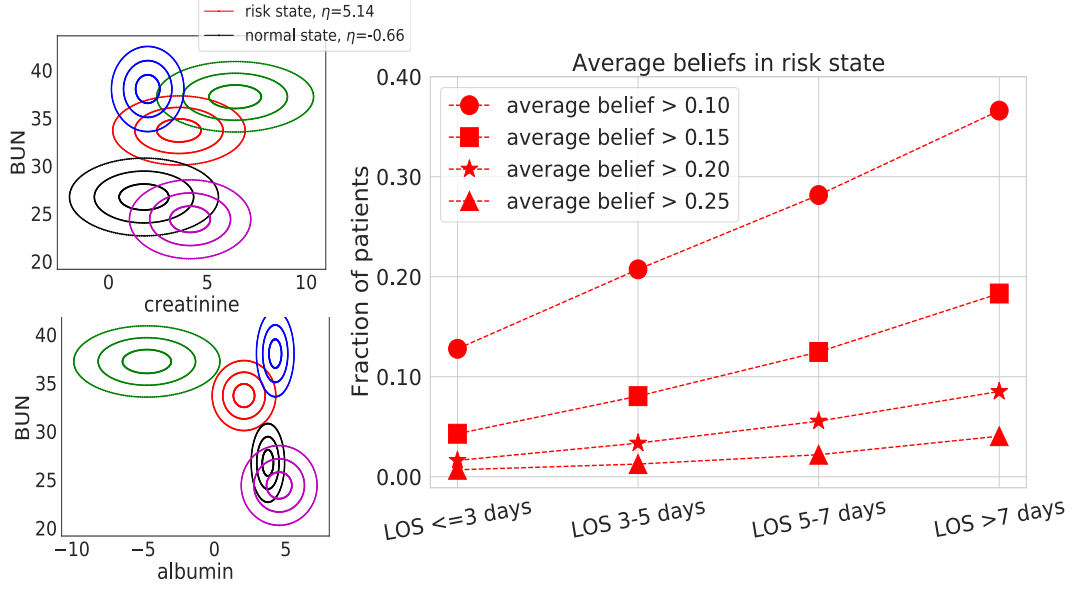


Figure 4.5: **Left** : Our PC-HMM model reveals five states, representing different risk levels. The red state indicates the ‘risk state’ associated with high BUN and low albumin and high creatinine. **Right**: We show the average beliefs, averaged over time, in the ‘risk state’. Notably, as the LOS increases, the average beliefs in the ‘risk state’ increases.

4.3.7 Training times for each model

		<i>seconds/epoch</i> to train each model				
Dataset		PC-HMM	GRU-D	MixMatch	FixMatch	BRITS
MIMIC-IV	$D = 41$ features	9	43	70	55	48
eICU	$D = 41$ features	16	70	115	90	75

Table 4.7: Computation time (seconds/epoch) required by each model. An epoch is completed when every example in the training set is covered at-least once. Although computation time increases when measurements are more frequent, the PC-HMM still requires far lesser time to train due to significantly lesser parameters than the other models.

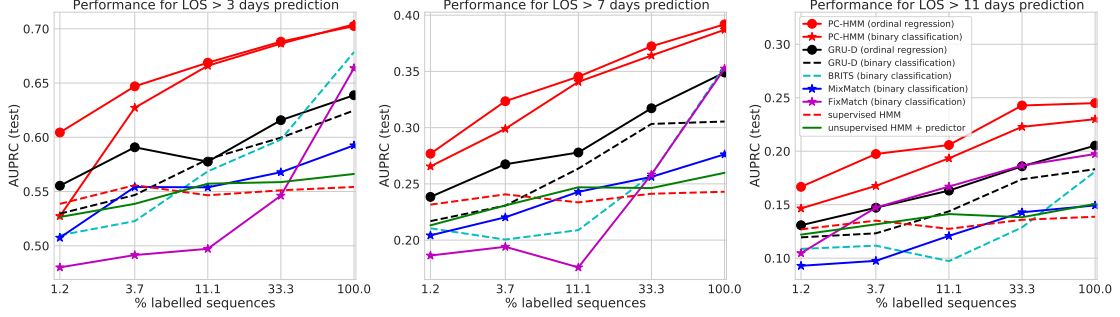


Figure 4.6: AUPRC (higher is better) versus amount of labeled data for length-of-stay prediction on MIMIC-IV at 3 ordinal labels (> 3 , > 7 , and > 11 days). SSL methods (PC-HMM, MixMatch, and FixMatch) learn from both labeled and unlabeled data. GRU-D [Che et al., 2018b] and BRITS [Cao et al., 2018b] use the labeled set only. Binary classification models require 3 separate models (1 for each label), whereas ordinal regression using CLMs require a single model to predict all 3 labels.

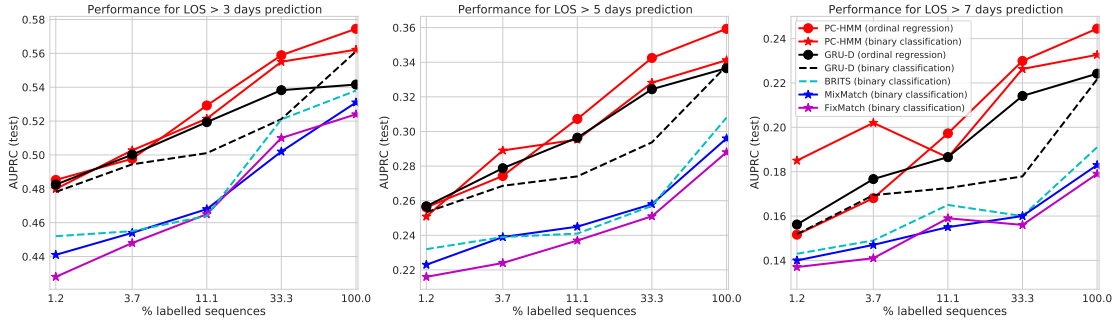


Figure 4.7: AUPRC (higher is better) versus amount of labeled data for length-of-stay prediction on eICU at 3 ordinal labels (> 3 , > 5 , and > 7 days). SSL methods (PC-HMM, MixMatch, and FixMatch) learn from both labeled and unlabeled data. GRU-D [Che et al., 2018b] and BRITS [Cao et al., 2018b] use the labeled set only. Binary classification models require 3 separate models (1 for each label), whereas ordinal regression using CLMs require a single model to predict all 3 labels.

4.3.8 Parameter counts and Hyperparameter search for each model

	PC-HMM	GRU-D	MixMatch	FixMatch	BRITS
Parameters	452-2,102	12,739-87,523	2,866-45,634	2,866-45,634	48,850-252,370
Jobs	900	7,560	10,800	21,600	6,480
Training time(per job)	1-2 hrs	5-6 hrs	6-8 hrs	4-6 hrs	4-5 hrs

Table 4.8: Range of parameter counts and number of jobs for hyperparameter search for each model. The PC-HMM requires the fewest parameters, least number of jobs for hyperparameter search, and the least training time and yet produces the best performance.

Model	Hyperparameters searched	Selection criterion
PC-HMM	• learning rate : 0.001, 0.01, 0.1 • states : 5, 10, 20 • initialization seeds : 5 • l2 penalty : 0, 0.01, 0.1, 1 • λ : 50, 100, 500, 1000, 5000	Best validation AUPRC
GRU-D	• learning rate : 0.001, 0.01, 0.1 • hidden units : 32, 64, 128 • hidden layers : 1, 2 • initialization seeds : 20 • l2 penalty : 7 log-spaced values between 10^{-6} and 100 • batch-size : 128, 256, 512	Best validation AUPRC with early stopping
BRITS	• learning rate : 0.001, 0.01, 0.1 • hidden units : 32, 64, 128 • initialization seeds : 20 • hidden layers : 1, 2 • dropout : 0, 0.1, 0.25 • batch-size : 128, 256, 512 • imputation-weight : 0.3, 0.6 • label weight : 1.0	Best validation AUPRC with early stopping
FixMatch	• learning rate : 0.001, 0.01, 0.1 • hidden units : 16, 32, 64 • initialization seeds : 5 • hidden layers : 1, 2 • batch-size : 128, 256, 512 • l2 penalty : 5 log-spaced values between 10^{-8} and 10 • pseudo-label thresholds : 0.6, 0.8 • augmentation noise variance : 0.1, 0.01 • unlabeled strength : 50, 75 • Temperature scaling : 0.5 • α(for MixUp): 0.75	Best validation AUPRC with early stopping
MixMatch	Same hyperparameter grid as FixMatch except no psuedo-label thresholds	Best validation AUPRC with early stopping

Table 4.9: Hyperparameters for all the models

4.4 Conclusion

In order to address the growing need to predict reliable LOS outcomes from patient data, we have presented a model that not only outperforms alternatives but also does so with minimal parameters. By taking advantage of the smooth likelihood for ordinal data, we show how the cumulative link framework may be incorporated into any model trainable with gradient descent. When incorporated into the PC-HMM, we observe how generative and discriminative goals are balanced to effectively

predict LOS in both fully- and semi-supervised cases. Unlike other candidates for ordinal regression, this established framework maintains a monotonically increasing probability for subsequent ordinal classes that are grouped together. This, along with their loss convexity, make CLMs an ideal candidate to address the novel problem of predicting hospital stays. Promising results on patient time-series datasets exhibit the utility of this model. We plan to explore further use cases such as characterizing disease severity with medical data.

Chapter 5

Probabilistic Modeling of Data-Driven Missingness in Irregular Time Series Medical Records

5.1 Introduction

In this chapter, we consider the problem of modeling a time series of several variables gathered at irregularly-spaced times, especially when only a few variables are typically measured at one time. Our motivating application is to model the sequence of vital signs and laboratory values recorded over time in the EHR of a hospitalized patient. Effective models of such data could enable many improvements to both patient care and hospital logistics, such as extrapolating the future state of the patient, facilitating diagnosis of underlying illness, or crafting personalized treatment strategies [Bica et al., 2021, Hyland et al., 2020a].

Realizing these benefits requires a model that can handle the two key aspects of our chosen scenario. First, the model must accommodate the *irregular timing of individual measurements*. Second, the model must capture the *data-dependent patterns of missingness*. To clarify this latter point, whether a specific variable is measured or not in the present moment is often influenced by current or recent measurements of other variables. Intuitively, doctors rarely order lab tests unrelated to the patient’s present condition or medical history.

We focus on a specific technical formulation of the generative mechanism for missingness, known as *missing at random* (MAR) [Rubin, 1976]. In MAR settings, the presence or absence of a measurement variable (binary indicator of missingness) depends on the values of all observed (non-missing) variables. As one example of MAR,

the current value of Troponin, an indicator of heart failure measured via lab processing of a blood draw, could be missing due to other observed vitals indicating normal heart function. As another example, the absence of cholesterol measurements could be attributed to the lack of evident risk factors for heart disease, alongside a normal weight and healthy Body Mass Index (BMI). In contrast to MAR, the commonly assumed *missing completely at random* (MCAR) setting assumes the missingness indicator arises independently of any observations.

Unfortunately, many common models for time-varying labs and vitals are not capable of handling both irregular timing and data-driven (MAR) missingness assumptions. Well-known recurrent neural nets for clinical time-series, such as GRU-D [Che et al., 2018a] or BRITS [Cao et al., 2018b], do allow for arbitrary missingness at each timestep at both training and prediction. However, these methods assume regular time intervals, necessitating alignment to a uniformly-spaced time grid as a pre-processing step. Another growing line of work directly models irregularly timed measurements via continuous formulation of recurrent networks [Pearlmutter, 1989, Pineda, 1987] and more recently via neural networks coupled with differential equation solvers [De Brouwer et al., 2019a, Rubanova et al., 2019b]. However, in these works (as well as works like GRU-D and BRITS) there is no focus on a data-dependent (MAR) missingness mechanism. The current landscape thus lacks a method that can simultaneously handle irregular-spacing and data-dependent missingness.

In this work, our **primary contribution** is a method that fills this gap. We present a probabilistic generative model for an irregular time series of clinical measurements with an data-driven mechanism that allows arbitrary missingness at any time. Irregularly-spaced measurements are handled via an encoder-decoder stochastic differential equation architecture known as the continuous recurrent unit (CRU) [Schirmer et al., 2022]. Data driven missingness is handled via regression on time-specific representation vectors produced by the CRU. Our approach preserves the efficient closed-form updates inside the CRU.

We demonstrate the effectiveness of our approach by examining prediction error

when *extrapolating* the next few hours of a patient’s multivariate time series. Across 3 real health records datasets, each containing thousands of patients, we show our approach delivers lower extrapolation error (averaged across 12+ labs and vitals) than several state-of-the-art alternative methods designed for irregularity but not data-dependent missingness. Further sensitivity analyses on synthetic data confirm that our advantage occurs specifically in settings where the MCAR assumptions of baselines are not appropriate. While our evaluations focus on EHRs for hospitalized patients, our method design is general purpose and useful for any time-series modeling that needs both irregular time and data-dependent missingness.

As a **contribution to best practices in evaluation**, our experiments report two kinds of error. First, we report average error across all channels after normalization (as many past works do). Second, we report channel-specific error without normalization, using the raw units of the original physiological measurement. This latter approach is not routinely done in other works, but we find it is valuable way to contextualize our method’s gains.

Generalizable Insights about Machine Learning in the Context of Healthcare

Our work highlights the importance of careful joint modeling of missingness and measured values, using a data-driven (MAR) mechanism, while also accounting for the irregularly-spaced nature of EHR records over time. The choice of what to measure and when carries critical information that can be used to improve understanding of the patient’s condition. Hopefully this improved understanding and ability to forecast future patient state could help lead to improved treatment decisions and better outcomes.

5.2 Prior Work

Below, we review several lines of work addressing either missingness or irregular timing.

Types of missingness. Foundational work in defining the 3 different missing data scenarios MAR (missing at random), MCAR (missing completely at random) and MNAR (missing not-at random) was done by Rubin [1976]. We situate our proposed model in the MAR category.

Missingness and deep generative models. Mattei and Frellsen [2019] introduced an approach for training deep generative models using importance weighted autoencoders (IWAE) under MAR assumptions. Ipsen et al. [2020] extended that work to cover MNAR assumptions with their not-MIWAE method, which was later specialized to supervised tasks [Ipsen et al., 2022]. Another model called GINA, introduced by Ma and Zhang [2021] provides identifiability guarantees for a wide range of MNAR mechanisms. All these works assume *static* data and are not designed to handle time series.

RNNs for regular time series with missing data. Several prior works tackle missing values with variants of recurrent neural networks (RNNs). Lipton et al. [2016] feed missing value indicators as additional inputs to the model together with forward-fill imputation to enable the model to learn 'real' vs 'filled-in' values. Che et al. [2018a] proposed the GRU-D, which extends the well-known GRU architecture [Cho et al., 2014a] by incorporating a trainable decay that imputes missing features in a way that interpolates over time between two strategies: forward-fill and the population mean. GRU-D allows the time since the last observation to impact representations in a way that usual regular-grid RNNs do not. However, their experiments required an input time series aligned to a regularly spaced grid. Cao et al. [2018b] designed a bidirectional RNN called BRITS that performs imputation and classification/regression jointly in one neural graph. Futoma et al. [2017] proposed a hybrid approach which uses a Gaussian process to impute values which are later fed into an RNN. More recently, Kim et al. [2023] proposed a probabilistic model for regularly spaced time-series with a missing not at random assumption. They extend

the not-MIWAE deep generative model [Ipsen et al., 2022] to time-series with a GRU backbone and introduce a novel regularization strategy to enable learning missing values for better downstream classification. Unlike all of these methods, our model does not rely on aligning data to a regularly spaced grid, thereby enabling learning from the raw irregularly spaced time-points.

Models for irregular time-series. Models for irregular time series belong to several types: (1) interpolation-based models and (2) ODE/SDE-based models, (3) Graph based models, (4) Diffusion based models, (5) Large language models (LLM) based models discussed below and (6) Tensor factorization based models

Interpolation-based models. Shukla and Marlin [2019] propose a semi-parametric interpolation network where irregularly spaced observations are interpolated to a grid of regularly spaced timepoints before prediction with any standard downstream predictor. Li and Marlin [2020a] propose an encoder-decoder architecture where the irregularly sampled time-series is assumed to be sampled from a continuous unobserved function. The encoder is a continuous convolution layer to accommodate irregularly sampled inputs, while their decoder uses a kernel smoother to learn a function over a reference set of uniformly spaced times. Shukla and Marlin [2021] propose an attention-based model that converts the irregular time series to a regular grid before prediction. For all these models, missing values are zero-encoded before training. There is no emphasis on data-dependent missingness.

ODE/SDE-based models. Rubanova et al. [2019b] propose a generative model where the latent state is driven by a latent ordinary differential equation (ODE). Kidger et al. [2020] extends that work via a neural-controlled differential equation instead of an ordinary differential equation. This allows the model to update the latent state trajectory with incoming observed data. De Brouwer et al. [2019a] extends the GRU architecture to incorporate input from an ODE, also in a way that allows observations to influence the latent trajectory. A similar ODE based model has also been proposed by Jia and Benson [2019]. Chauhan et al. [2022] extend the continuous state perceiver network [Jaegle et al., 2021] to learn the continuous

dynamics of a patient, requiring 2 ODE solvers. All these works require expensive numerical solvers for training and prediction.

Graph-based models. Zhang et al. [2021] introduced RAINDROP, a graph neural network that handles asynchronous and irregularly recorded observations from multiple sensors. It considers sensor dependencies for message passing and generates multi-scale embeddings using hierarchical attention. However, the model estimates missing values using observational embeddings from neighboring vertices without learning a probabilistic model for missingness. Further, it is hard to assess the quality of imputations by their model, since only classification performance on downstream tasks are presented.

Diffusion based models. Tashiro et al. [2021] propose a score-based diffusion model trained via self-supervision and is conditioned on observed data for imputation. While their model claims to reduce absolute errors by a 5 – 20% margin compared to other models, their evaluation depends on intricate adjustments to the denoising function and a self-supervised learning strategy that separates observed values into conditional information and imputation targets for training.

Large language models. Recently, Gruver et al. [2023] demonstrated that zero-shot extrapolation with Large Language Models (LLMs) outperform numerous time-series forecasting baselines in performance. However, their approach forecasts each dimension independently and ignores the actual timestamps of measurements. This limitation renders their model ineffective for datasets containing irregular timestamps and correlations between channels. We add the LLMTime baseline to our toy data experiment in Figure 5.3 and highlight the poor extrapolation performance on synthetic data.

Tensor factorization based models. [Yin et al., 2020] introduced Logistic PARAFAC2, a model designed to handle binary data with missing values by using Bernoulli distributions and a positive-unlabeled learning loss function. This approach improves tensor completion and predictive performance by incorporating uniqueness and temporal smoothness regularization. However, their MIMIC-III dataset is constructed

on an eight-hour basis by accumulating clinical features in consecutive time windows. This may not capture all relevant temporal dynamics and could limit the applicability to clinical features that vary within shorter time frames.

Our work directly builds upon the continuous recurrent unit (CRU) by Schirmer et al. [2022]. As detailed in Sec. 5.3.1, the CRU is an SDE-based method for representation learning on irregular time series. Hidden representations evolve via a linear time-invariant stochastic differential equation (SDE), with encoder/decoder mappings for added flexibility. However, the off-the-shelf CRU model does not model *missingness* at all. Our proposed method extends the CRU to handle data-dependent missingness scenarios. Recent work by Ansari et al. [2023] propose a neural continuous discrete state space model (NCDSSM). This is similar to the CRU but it avoids the CRU’s restriction to linear time-invariant SDEs. Again, that work does not address learning missing mechanisms.

5.3 Background

Here, we review the technical problem we are trying to solve as well as the details of Schirmer et al. [2022]’s continuous recurrent unit (CRU). This lays a conceptual foundation for our later innovations that incorporate data-dependent missingness mechanisms in Sec. 5.4.2.

5.3.0.0.1 Problem: Irregular time series with missingness. We consider building a model for many multivariate sequences. For each sequence, we have observations at a set Γ of L distinct irregularly-spaced observation times: $\Gamma = \{t_1, \dots, t_L\}$, which we assume are sorted in ascending order and are all greater than a reference start time t_0 .

We also have a sequence $\mathbf{x}$ of observed feature vectors, one for each observed time: $\mathbf{x} = \{x_t : t \in \Gamma\}$. Each feature vector $x_t \in \mathbb{R}^D$ represents measurements of a fixed set of D variables (sometimes called channels). There may be arbitrary missingness in each x_t . Let $s_t \in \{0, 1\}^D$ be a binary indicator vector of size D indicating whether a

variable is “seen” (observed) or not (missing). That is, $s_{td} = 1$ if x_{td} is observed and 0 otherwise.

5.3.1 Continuous Recurrent Unit (CRU)

A core building block of our approach is the continuous-time recurrent unit (CRU) by Schirmer et al. [2022]. The purpose of the CRU is to provide a mapping from a *non-missing* irregular time series of length L to a output sequence of representation vectors of the same length L . We write the CRU as

$$[h_{t_1} \dots h_{t_L}], [\tau_{t_1} \dots \tau_{t_L}] \leftarrow \text{CRU}(x_{t_1} \dots x_{t_L}, \Gamma). \quad (5.1)$$

At each observation time t_ℓ , the CRU produces two D -dimensional vectors $h_{t_\ell}, \tau_{t_\ell}$. That is, $|h_t| = |\tau_t| = |x_t| = D$ for all $t \in \Gamma$. Vector h_{t_ℓ} is can be thought of as a decoded representation for data x_{t_ℓ} . Vector τ_{t_L} is meant to encode uncertainty about each data dimension.

We stress that the CRU as presented by Schirmer et al. [2022] assumes that each x_t is fully observed, without any missingness. While they offer a zero-filling heuristic to accommodate missingness, our later experiments show this is inadequate for downstream predictions when data is missing-not-at-random. Instead, our methods in Sec. 5.4.2 provide an explicit mechanism to handle the data-dependent missingness scenarios.

5.3.1.0.1 CRU overview. The CRU produces representations via continuous-discrete Kalman-filter forward inference [Jazwinski, 1970] for a latent variable time-series model. Internally, the CRU assumes two time-varying continuous random variables: a M -dimensional latent vector z_t , and a C -dimensional encoded feature vector y_t . Latent z_t has a *prior* distribution specified by a stochastic differential equation (SDE) model for z_t as a function of t . When this SDE assumes a Gaussian noise process, this ultimately yields a Gaussian prior with time-varying parameters. We then define a Gaussian *likelihood* that produces encoded features y_t given

latent z_t . Due to conjugacy between the solution to the prior SDE and the likelihood, Jazwinski [1970] shows that the posterior over z_t given previously observations, $p(z_t | \{y_u : u < t, u \in \Gamma\})$, remains Gaussian with closed-form mean and covariance.

The CRU proceeds by mapping observed time series x to encoded features y and then employing forward Kalman-filter recursions to update the time-varying mean and covariances that summarize the posterior on latent z . Later, a decoder neural network maps the posterior mean and uncertainty from the latent space to the D -dimensional data space, producing the sequence of h and τ values in Eq. (5.1).

We emphasize that although there is an underlying probabilistic model for z and y , the actual CRU computations are all *deterministic*, like any other recurrent neural net. Given an input sequence of data values x , the CRU will always produce the same latent means, latent covariances, and decoded data representations h, τ . Our later model thus treats the CRU as a flexible non-linear transformation function, like any other recurrent neural net.

5.3.1.0.2 CRU prior on z . The CRU model assumes a continuous latent vector $z \in \mathbb{R}^M$, which as a function of time $z(t)$ is governed by a linear time invariant (LTI) SDE

$$dz = Azdt + Gd\beta, \quad p(z_{t_0}) = \mathcal{N}(m_0, P_0), \quad (5.2)$$

where $A \in \mathbb{R}^{M \times M}$ is the time-invariant transition matrix, $G \in \mathbb{R}^{M \times B}$ is the diffusion coefficient and $\beta \in \mathbb{R}^B$ is a Brownian motion process (aka Wiener process) with zero mean and diffusion matrix $Q \in \mathbb{R}^{B \times B}$. The initial density condition (at time t_0) for the LTI-SDE is Gaussian, with m_0 and P_0 as the mean and covariance.

Following Särkkä and Solin [2019], the solution to Eq. (5.2) has the form

$$z(t) = \text{MEXP}(A(t - t_0)) z(t_0) + \int_{t_0}^t \text{MEXP}(A(t - \gamma)) G d\beta(\gamma)$$

Here MEXP refers to the matrix exponential, which maps a square matrix to another square matrix of the same size (details in the appendix).

5.3.1.0.3 CRU likelihood of y given z . The CRU model assumes that time-varying vector $y \in \mathbb{R}^C$ is generated by a Gaussian whose mean is a linear function of z

$$y_t | z_t \sim \mathcal{N}(Rz_t, \sigma_t^2 I_C),$$

where matrix $R \in \mathbb{R}^{C \times M}$ and scalar $\sigma_t > 0$ are learnable parameters.

5.3.1.0.4 Posterior of z given y . First, we can provide a formula for $p(z_t | y_{<t})$, our belief about z_t using only previous observations of y values:

$$p(z_t | y_{<t}) = \mathcal{N}(\mu_t^-, \Sigma_t^-) \quad (5.3)$$

where μ_t^- and Σ_t^- at time t have analytical solutions [Särkkä and Solin, 2019] that depend also on the value t' indicating the most recent previous observation time before t in Γ :

$$\mu_t^- = \begin{cases} \text{MEXP}(A(t - t'))m_0 & \text{if } t' = t_0, \\ \text{MEXP}(A(t - t'))\mu_{t'}^+ & \text{if } t' > t_0 \end{cases} \quad (5.4)$$

$$\Sigma_t^- = \text{MEXP}(A(t - t'))\Sigma_{t'}^+ \text{MEXP}(A(t - t'))^T + \int_{t_0}^t \text{MEXP}(A(t - \gamma))GQG^T \text{MEXP}(A(t - \gamma))^T d\gamma$$

In other words, the LTI SDE is propagated forward from either *initial conditions* (at the beginning of the sequence, when $t' = t_0$) or *updated conditions* (from the most recent previous observation at t').

Next, we can ask about the distribution of z_t if we condition on additionally knowing the current observation y_t as well as all previous y values. By Bayes theorem, we have

$$p(z_t | y_{\leq t}) \propto p(y_t | z_t) p(z_t | y_{<t}), \quad p(z_t | y_{\leq t}) = \mathcal{N}(\mu_t^+, \Sigma_t^+)$$

We arrive at this closed-form Gaussian because both densities on the right side

of the first line are Gaussian. Vector μ_t^+ and matrix Σ_t^+ represent the updated mean and covariance of z_t , using all observed y values at or before the current time t . These are computed using linear-Gaussian formulas of the continuous-discrete Kalman filter [Jazwinski, 1970]:

$$\mu_t^+ = \mu_t^- + K_t(y_t - R\mu_t^-), \quad \Sigma_t^+ = (I_M - K_t R)\Sigma_t^-, \quad K_t = \Sigma_t^- R^T (R\Sigma_t^- R^T + \sigma_t^2 I_C)^{-1}. \quad (5.5)$$

5.3.1.0.5 CRU summary. The CRU has three components: (i) an encoder network f with weights ψ , (ii) a decoder network g with weights ϕ , and (iii) fixed values m_0, P_0 and learnable parameters A, G, Q, R that define the linear-Gaussian SDE model. The main motivation for a non-linear encoder-decoder mapping (where the dimension of the encoded space is lower than the original observation space) is to ensure a gating mechanism that can optimally integrate noisy and sparse observations at arbitrary observation times. This noise modeling is crucial for adjusting the confidence in the observations, which directly affects the update mechanisms in the CRU’s state estimation. Moreover, ignoring the encoder-decoder structure would necessitate an extra step of data imputation to accommodate the continuous-discrete Kalman updates, thereby complicating the modeling process.

To process an input sequence with the CRU, we first initialize $\mu_{t_0}^+ = m_0$ and $\Sigma_{t_0}^+ = P_0$. Then, iteratively perform the following deterministic algorithm:

for each time $t \in \Gamma$ with adjacent previous time t' :

1. $y_t, \sigma_t \leftarrow f_\psi(x_t)$ (5.6)
2. $\mu_t^-, \Sigma_t^- \leftarrow \text{PREDICT}(\mu_{t'}^+, \Sigma_{t'}^+, t, t')$ via Eq. (5.4)
3. $\mu_t^+, \Sigma_t^+ \leftarrow \text{UPDATE}(\mu_t^-, \Sigma_t^-, y_t, \sigma_t)$ via Eq. (5.5)
4. $h_t, \tau_t \leftarrow g_\phi(\mu_t^+, \Sigma_t^+)$

The decoded data representations at each timestep, $h_t \in \mathbb{R}^D$ and $\tau_t \in \mathbb{R}^D$, as well as the latent representation $\mu_t^+ \in \mathbb{R}^M$, can then be used in downstream modeling similar

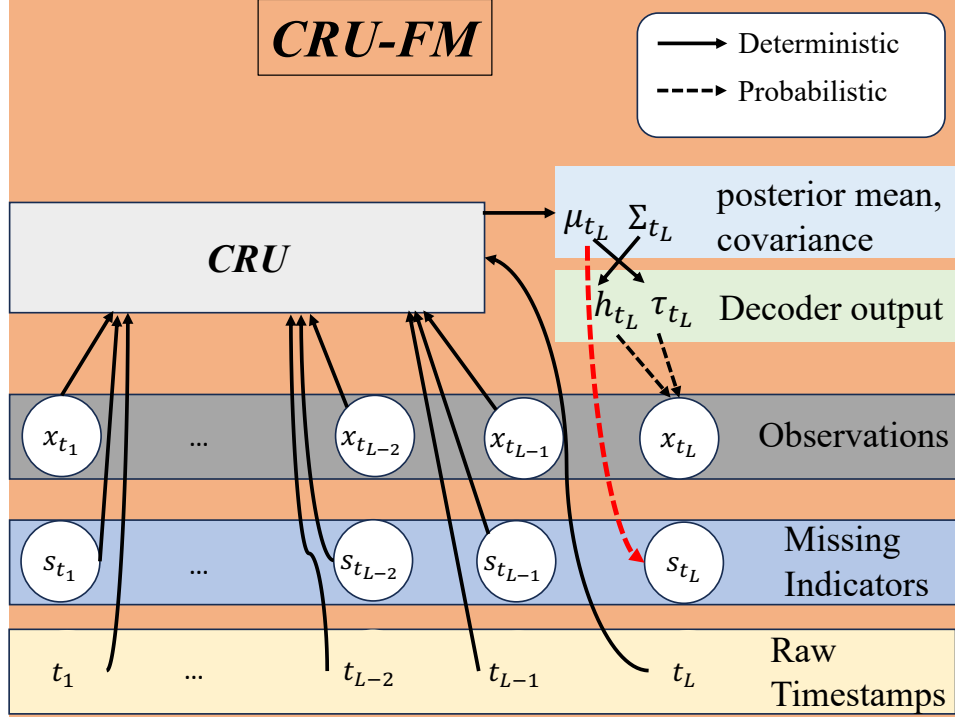


Figure 5.1: Diagram of our CRU-FM model (Sec. 5.4.2). We show the extrapolation generative process for data vector x_{t_L} and binary seen/missing indicator s_{t_L} at time t_L , conditioning on previous $x_{t_1:t_{L-1}}, s_{t_1:t_{L-1}}$. The red line highlights our contribution: a distribution on s_{t_L} that is data-dependent and thus suitable for MAR settings. Dashed lines point to values that are sampled from a distribution, not deterministically computed from predecessors.

to the outputs of any recurrent neural net. Because the entire iterative algorithm is amenable to automatic differentiation, the computation of gradients of the outputs with respect to all CRU parameters is tractable.

5.4 Methods

In order to develop probabilistic models for irregular time series with arbitrary missingness, let us define some essential notation. First, for a given time t , let x_t^S denote the subset of all entries of $x_t \in \mathbb{R}^D$ that are *seen* or present, not missing. x_t^S may vary in size depending on how many entries are missing at time t . Second, let $\bar{x}_t$ be a vector of size D constructed from x_t by filling any missing entry with a numerical zero, but otherwise leaving present entries alone. Recalling that s_t is a binary vector indicating the seen entries of x_t , we can write $\bar{x}_t \leftarrow x_t \odot s_t$ to represent the

element-wise product (denoted $\odot$) which fills any missing entry with zero.

5.4.1 CRU model for seen features only

Schirmer et al. [2022] provide the following generative model for $\mathbf{x}^S$, the *seen* (non-missing) entries of the D -dimensional time series $\mathbf{x}$:

$$\begin{aligned} p(\mathbf{x}^S) &= \prod_{t \in \Gamma} p(x_t^S | \{x_u^S : u < t, u \in \Gamma\}), \\ &= \prod_{t \in \Gamma} \prod_{d: s_{td}=1} \mathcal{N}(x_{td} | h_{td}, \tau_{td}^2), \end{aligned} \quad (5.7)$$

where h_t, τ_t are the decoded data representations for time t produced by a CRU given zero-filled input: $[h_{t_1}, \dots], [\tau_{t_1}, \dots] \leftarrow \text{CRU}(\bar{x}_{t_1}, \dots, \bar{x}_{t_L}, \Gamma)$. Symbol $\mathcal{N}$ is the Gaussian pdf.

While this model can be fit on irregular time series with arbitrary missingness, we stress that it does not model the missingness indicator s_t as a random variable and thus cannot account for data-dependent missingness mechanisms. Our later experiments show that in both synthetic and real-world missingness scenarios, this leads to poor recovery of the underlying model and high error in extrapolation.

5.4.2 CRU-FM generative model

To better capture data-dependent missingness, we propose a generative model of both seen features $\mathbf{x}^S$ and binary seen/missing indicators $\mathbf{s}$ over time. We call this the CRU-FM, emphasizing the joint modeling of both Features and Missingness, and illustrate this model in Fig. 5.1. The joint distribution is defined as:

$$\begin{aligned} p(\mathbf{x}^S, \mathbf{s}) &= \prod_{t \in \Gamma} p(x_t^S, s_t | \{x_u^S, s_u : u < t, u \in \Gamma\}) \\ &= \prod_{t \in \Gamma} \left[\prod_{d: s_{td}=1} \mathcal{N}(x_{td} | h_{td}, \tau_{td}^2) \cdot \prod_{d=1}^D \text{Bern}(s_{td} | r(\eta_d^T \mu_t^+)) \right] \end{aligned} \quad (5.8)$$

Here, we again provide zero-filled features $\bar{\mathbf{x}}$ as input to the CRU to obtain the representations: $[h_{t_1}, \dots], [\tau_{t_1}, \dots], [\mu_{t_1}^+, \dots] \leftarrow \text{CRU}(\bar{x}_{t_1}, \dots, \bar{x}_{t_L}, \Gamma)$. Note that latent

mean μ_t^+ can easily be accessed from step 3 of the CRU algorithm defined in Eq. 5.6. For each data dimension d , vector $\eta_d \in \mathbb{R}^M$ represents learnable weights for a regression that estimates the probability of observing variable d given the M -length vector μ_t^+ from the CRU. Symbol $r(\cdot)$ denotes the logistic sigmoid function, which maps a real value to a probability in $[0,1]$.

The chosen form of Eq. (5.8) balances two desiderata. First, we want the generative process for indicator vector s_t to be data-dependent to address MAR scenarios. Second, we want to maintain the efficient computational properties of the CRU. Our construction ensures both. The joint over s_t and x_t is clearly dependent on all previous observations $x_{1:t-1}^S$. However, we can still benefit from the CRU's efficient algorithm in Eq. (5.6).

Extension. A more general extension of our proposed CRU-FM is easily possible, which keeps the same model for x^S but models each entry of s_t via

$$s_{td} \sim \text{Bern}(r(\eta_d^T \rho(\mu_t^+, h_t, \tau_t, t))), \quad (5.9)$$

where function ρ produces a fixed-size vector representation via any deterministic and differentiable mapping of provided representations at time t , with a weight vector η_d of suitable size. For example, in a hospital care setting, allowing non-linear dependent on the time of day via timestamp t could capture the fact that typically fewer lab tests are done at night while patients sleep. One example of such a formulation could be :

$$s_{td} \sim \text{Bern}(r(\eta_{d1}^T \mu_t^+ + \eta_2^T t)), \quad (5.10)$$

where $\eta_d = \{\eta_{d1}, \eta_2\}$. Similarly, by providing decoded data vector $h_t \in \mathbb{R}^D$ and uncertainty $\tau_t \in \mathbb{R}_+^D$ as input, the mean and variance values of specific measurements related to feature d could modulate its missingness. Training this extended CRU-FM would be straightforward.

Our later experiments examine only the simpler model in Eq. (5.8), so that

comparisons directly assess how the addition of data-dependent missingness improves performance.

5.4.3 Training

To train our model, we assume we are given a collection of N irregularly-spaced sequences of the same D possible measurement variables. Each sequence, indexed by n , has its own set of observation times Γ_n , its own seen feature time series $\mathbf{x}_n^S$, and its own binary indicator vector time series $\mathbf{s}_n$. We assume each sequence is independent and identically-distributed according to the CRU-FM model in Eq. (5.8).

We thus frame the goal of fitting the CRU-FM model to N patient records as finding the CRU parameters θ (defined in Sec. 5.3.1, encompassing the SDE parameters as well as encoder and decoder weights) and data-dependent regression weight η (defined in Eq. (5.8)) that maximize the log likelihood of all N observed sequences under our chosen model. Equivalently, we solve the following optimization problem

$$\theta^*, \eta^* \leftarrow \underset{\theta, \eta}{\operatorname{argmin}} \lambda \eta^T \eta - \sum_{n=1}^N \log p_{\theta, \eta}(\mathbf{x}_n^S, \mathbf{s}_n) \quad (5.11)$$

where scalar hyperparameter $\lambda > 0$ governs the strength of regularization on regression weights η that govern the data-dependent generation of each s_t . Recall that our model’s density function $p_{\theta, \eta}(\mathbf{x}_n^S, \mathbf{s}_n)$ is defined in Eq. (5.8). The notation $p_{\theta, \eta}$ reminds us to view this density as a function of parameters θ, η . Evaluation of this log likelihood objective and its gradients with respect to θ, η is straightforward using modern automatic differentiation.

5.4.4 Implementation Details

We solve parameter estimation using the Adam optimizer [Kingma and Ba, 2014a] training for 100 epochs. We perform hyperparameter search across 2 different batch sizes (64 and 128), 3 different learning rates (0.001, 0.005, and 0.01), and 6 possible regularization strengths λ (between 10^{-6} and 0.1) and then choose the configuration

with lowest RMSE on the validation set. Details for reproducibility are in the Supplement. For all experiments involving both the basic zero-fill CRU model as well as our CRU-FM model, we use the efficient implementation of CRU proposed by Schirmer et al. [2022] as *f-CRU*. This method simplifies the dynamics of $z(t)$ by representing matrix A as a weighted average of K basis matrices with shared orthogonal eigenvectors, reducing the matrix exponential’s complexity from $O(M^3)$ to $O(KM)$ in Eq. (5.2).

5.4.5 Evaluating Extrapolations

In later experiments, we wish to evaluate how well our model can extrapolate beyond the provided observation times Γ . For each sequence of interest, we’ll suppose we have additional future times of interest Γ^* (all larger than the times in Γ).

Given parameters θ defining our CRU-FM model, we make extrapolative *predictions* $\hat{x}_t^*$ of future feature values for all $t \in \Gamma^*$. The procedure is a straightforward sequence of steps, all of which condition *only* on the observed x_t for $t \in \Gamma$. First, we obtain representations h_t, τ_t for each t in the union of $\Gamma \cup \Gamma^*$ by running the CRU algorithm forward as in Eq. (5.6). For the observed times in Γ , the algorithm runs as written. For the heldout times in Γ^* , we modify the algorithm to just do forward prediction and decoding. No observation in x^* impacts the extrapolation. We skip step 1, for step 2 we define t' as the last time in Γ , we skip step 3, and for step 4 we compute $h_t, \tau_t \leftarrow g_\phi(\mu_t^-, \Sigma_t^-)$. Ultimately, we predict the most likely feature value at each dimension and time, via

$$\hat{x}_{td}^* \leftarrow h_{td}, \text{ for all } t \in \Gamma^*, \ d : s_{td}^* = 1 \quad (5.12)$$

Across all future times, we then compute and report error between the prediction $\hat{x}_t^*$ and the seen ground-truth features x_t^{*S} , using only dimensions d where we have ground-truth (satisfying $s_{td}^* = 1$) and ignoring the others.

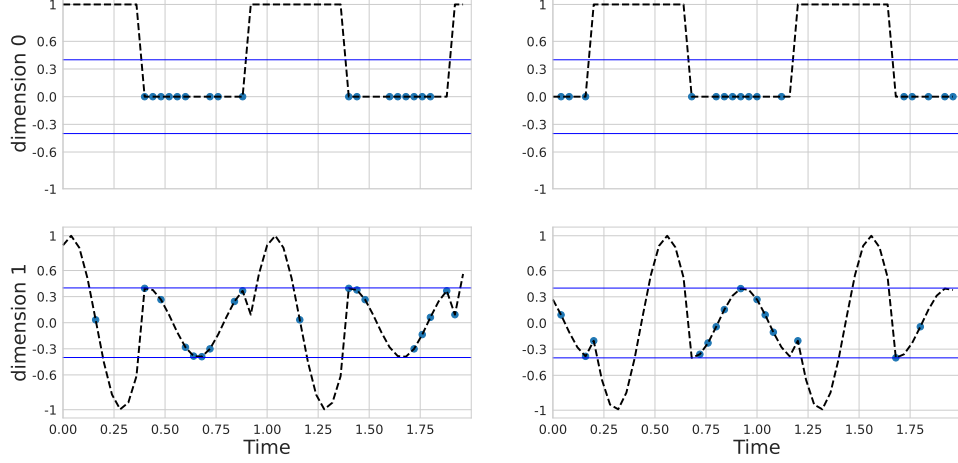


Figure 5.2: Two examples of generated sequences for the synthetic task. The dashed black lines represent the true data-generating process. The circular dots represent the observations at irregular timepoints. To incorporate data-dependent missingness, the probability of observing a value within the solid blue lines is 0.6; outside the lines is $p_S = 0.0006$.

5.5 Experimental Results

We now assess our CRU-FM model experimentally. First, we study a synthetic task where we can control the missingness mechanism precisely (varying between data-dependent and data-independent). Second, we study extrapolation performance on 3 real open-access datasets of critical care electronic health records (EHR).

We compare to 6 total competitive baselines for irregular time series modeling. First, Schirmer et al. [2022]’s CRU model (Sec. 5.4.1) is a natural baseline, as it represents removing the data-dependent missingness mechanism of our CRU-FM. Additionally, we compare to mTAND [Shukla and Marlin, 2021], pVAE [Li and Marlin, 2020a], Latent ODE [Rubanova et al., 2019b], and Neural CDE [Kidger et al., 2020]. On the synthetic task, we also compare to an additional recent LLM baseline for time series forecasting called LLMTime [Gruver et al., 2023]. Due to its poor performance, we did not apply it to any EHRs.

5.5.1 Synthetic Experiments

Purpose. We designed a synthetic extrapolation task with $D=2$ data channels to assess how well each model can learn from hundreds of sequences drawn from a

known data-generating process that includes data-dependent missingness (defined in equations 5.5.1 and 5.14). Figure 5.2 illustrates two example sequences. The top channel is a square wave with a sequence-specific phase offset. The bottom channel is a sinusoid whose amplitude and phase depend on the current value of the top channel. As detailed below, our data-generation process introduces missingness in a channel at different rates depending on whether the channel’s current value is inside or outside the illustrated blue horizontal bounds.

This process results in significant missingness in the top (first) channel. Thus, in order to deliver good extrapolations for both channels, a model must accurately model dependencies in feature values and missingness jointly across channels.

Setup. To generate each sequence (indexed by n), we draw a horizontal offset $b_n \sim \text{Unif}([0, 2])$. This value informs a deterministic “true” time-varying data-generation function $x_{n,d}(t)$ for each dimension d : we have a square wave for $d = 1$ and a sine wave for $d = 2$

$$\begin{aligned} x_{n,1}(t) &= \text{sgn}(\sin(4\pi t + b_n)) \\ x_{n,2}(t) &= \sin(4\pi t + b_n) \cdot x_{n,1}(t) + 0.4 \cdot \cos(4\pi t + b_n) \cdot (1 - x_{n,1}(t)) \end{aligned} \quad (5.13)$$

For each sequence n , a set Γ_n of observation times is drawn with 50 elements, each one i.i.d. uniform within $(0, 2)$. At each time $t \in \Gamma_n$, the indicator controlling whether dimension d is seen or missing is then sampled so that inside the blue lines of Fig. 5.2, there is a good chance of observation (60%), but outside the lines the chance is lower:

$$p(s_{ntd} = 1) = \begin{cases} 0.6 & \text{if } -0.4 < x_{n,d}(t) < 0.4 \\ p_S & \text{otherwise} \end{cases} \quad (5.14)$$

Here, scalar $p_S > 0$ determines the chance of seeing an outside-the-lines value. By default, we fix $p_S = 0.0006$, yielding a data-dependent missingness mechanism for both channels. Later sensitivity analyses study what happens as p_S increases toward 0.6

(MCAR scenario).

Extrapolation task. We generated 1000 sequences (keeping only the x_{nt}, s_{nt} values at times in Γ_n), divided into train/valid/test splits of size 700/150/150. After training and tuning any model, we measure extrapolation on each test sequence by providing data up to time $t = 1.5$, then obtaining from each model the *most likely* predicted value for each channel at each heldout time in $(1.5, 2.0)$.

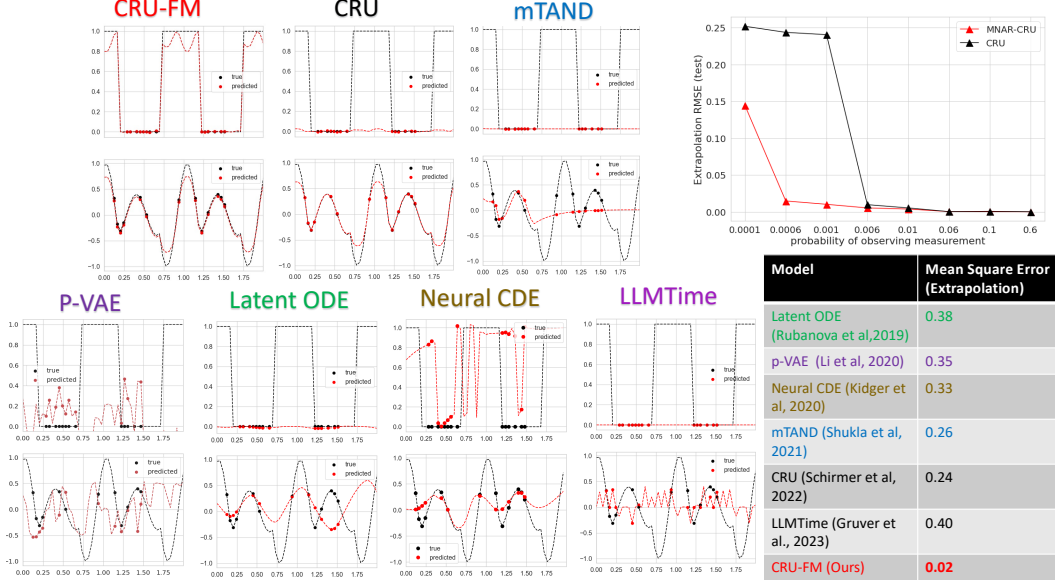


Figure 5.3: Synthetic task results comparing CRU-FM to competitors. *Left:* Visuals of reconstruction ($t \in (0, 1.5)$) and extrapolation ($t \in (1.5, 2)$) for the *same* 2-channel test sequence when missingness is data-dependent ($p_S=0.0006$). Black (red) lines indicate true (reconstructed) values. To fairly compare models, we evaluate each one’s most likely prediction $\hat{x}_t$ (mode of the predictive distribution). For example, for our CRU-FM we set $\hat{x}_t = h_t$ (via Eq. (5.8)). This choice eliminates extraneous randomness needed to sample from a predictive distribution. *Bottom Right:* Extrapolation results on the synthetic task sequences comparing MNAR-CRU, CRU, MTAND and p-VAE. The black/red dots are the observed/predicted values at each irregularly sampled time point. The black/red line is the true/reconstructed sequence. Each model is trained until $t = 1.5$ and the extrapolated from $t = 1.5$ to $t = 2$. MNAR-CRU generates sequences that are closer to the ground-truth and also achieves the least mean squared error on the extrapolation task. We attribute this improved performance of the MNAR-CRU to the model’s ability to learn the true missingness probability in each sequence. *Top right :* We examine how extrapolation error changes as the probability of observing a measurement (denoted as p_{obs}) outside the $(-0.4, 0.4)$ bounds changes. As p_{obs} increases (MNAR assumption relaxes), as expected the CRU and MNAR-CRU perform similarly. For low values $p_{obs} \leq 0.006$, MNAR-CRU’s better-specified missingness model notably improves extrapolation (via a joint model of x_t and s_t).

Results: Head-to-head comparison with competitor models : Fixing $p_S = 0.0006$ to ensure data-dependent missingness, we compare the extrapolation error of our CRU-FM with all competitors. In the lower right panel of Fig. 5.3, CRU-FM delivers by far the lowest overall mean squared error on the test set of 150 sequences (0.02 vs. 0.24 for the next best competitor, the CRU).

To understand possible reasons for CRU-FM’s improved performance, we show plots of sequence-specific reconstructions/extrapolations from each model (left panels of Fig. 5.3). These plots confirm CRU-FM recovers the underlying “true” model structure best, especially looking at the more difficult first (top) channel. All models are trained on the same 700 sequences, few of which do have rare cases of observations from the top of the square wave. We suggest our model handles this top channel better than alternatives because it is the only tested model capable of data-dependent missingness. CRU-FM’s latent representations μ_t are informed by both features x_t and seen/missing indicators s_t , enabling learning of dependencies between channels (e.g. steep slope in channel 2 is indicative of a crest in channel 1 even when channel 1 is often missing).

As illustrated in Fig. 5.3, the second (bottom) channel is extrapolated similarly well by both the CRU and the CRU-FM, both much better than other models. We suggest two possible reasons CRU is similar to CRU-FM on this channel. First, channel 2 spends less time outside the $(-0.4, 0.4)$ bounds and is thus typically observed. Second, unlike the sharp jump of the square wave in channel 1, the true function of channel 2 transitions *smoothly* across the $(-0.4, 0.4)$ bounds, so sane extrapolations are possible even for simpler models.

Results: Sensitivity analysis to missingness mechanism. To further verify that capturing the data-dependent missingness of our model is the primary reason for success, we sweep p_S from 0.0006 to 0.6 (MCAR). At each value, we resample all train/valid/test data and retrain the CRU-FM and CRU. As expected, the top-right panel in Fig. 5.3 shows that our CRU-FM yields far lower error in the data-dependent missingness setting, but becomes indistinguishable from the CRU in the MCAR setting. This helps rule out other confounding explanations for which model

is superior, like better tuning or lucky initialization.

Results : Parameter Learning on Toy Data We show the capability of our model to learn the missingness parameter for each channel in our toy example in figure 5.4

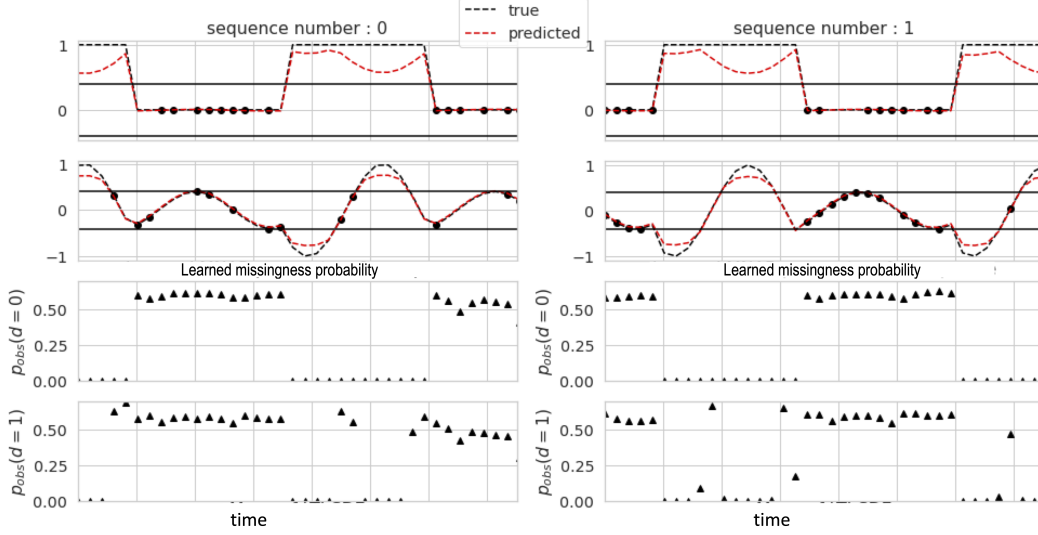


Figure 5.4: Demonstration of our CRU-FM model’s ability to learn the data-dependent missingness mechanism. *Top 2 rows* : We show the 2 channels of data from 2 representative sequences from our toy experiment (details in section 5.5.1). The black dashed line and the red dashed line represent the true and reconstructed sequences respectively. The black dots represent the observed values at irregularly spaced timepoints. *Bottom 2 rows* : Learned probabilities of observing a measurement in each of the channels above. Remember, we set the data-dependent missingness mechanism as follows : the probability of observing a value outside the dashed black lines for each channel is 0.0006 and the probability of observing a measurement within the dashed black lines is 0.6. This mechanism is learned by our model (the black upward arrows show the learned probabilities of observing a value). The ability of our model to learn these data-dependent missingness mechanisms leads to improved downstream extrapolation performance.

5.5.2 Results on Real EHR Datasets

Next, we examine extrapolation error when trained models from different competitor methods get to observe the first 24 hours of patient health-records (labs and vitals) then must extrapolate the same measurements over the next 6-24 hours. We evaluate on 3 separate large EHR datasets that are all de-identified and publicly available.

Physionet-2012 setup. The Physionet-2012 challenge dataset [Silva et al., 2012] is available at <https://physionet.org/content/challenge-2012/1.0.0/>. We only include 8000 ICU stays lasting at-least 48 hours and exclude static measurements, following earlier analyses by Rubanova et al. [2019b]. For each stay, we model all 37 labs and vitals available in released data (see Tab. 5.1 for feature details and missingness rates). Our train/valid/test splits have 4800/1600/1600 patient stays.

MIMIC-IV setup. Out of single-hospital MIMIC-IV dataset [Johnson et al., 2020], we select 43,883 ICU stays that last at least 30 hours. For each stay, we track 37 time-varying labs and vitals used in prior works [Gong et al., 2017, Nestor et al., 2018] (see Tab. 5.2 for feature missingness details). Our train/valid/test splits have 28084/7022/8777 patient-stays.

eICU setup. Out of the over 200,000 ICU-stays in the multi-hospital eICU dataset [Johnson et al., 2020], we select all patient stays that last at least 30 hours. For each stay, we track 12 labs (see Tab. 5.3). We exclude channels like respiratory rate, temperature and arterial blood pressure as these measurements are collected more regularly. Our train/valid/test splits have 63555/15889/19862 patient-stays.

Measuring uncertainty. For each method on each dataset, we record its RMSE when extrapolating on the test set. To construct uncertainty intervals, we follow Raschka [2018] and perform bootstrap sampling (with replacement) from the entire test set 100 times. For each of the 100 sets, we evaluate the RMSE and report the median, 5th and 95th percentiles.

5.5.2.0.1 Results: Overall extrapolation error. For each dataset in Fig. 5.5, a panel labeled “All channels (normalized)” reports overall performance across all examined vitals and labs (even those not depicted in other panels). This requires normalizing actual and predicted values from each channel to a 0-1 scale using the min and max estimates from the training set, then computing the average RMSE. Across all 3 datasets, our CRU-FM consistently scores better than any competitor on this overall error.

In addition to our CRU-FM and the 5 competitor methods for irregular time

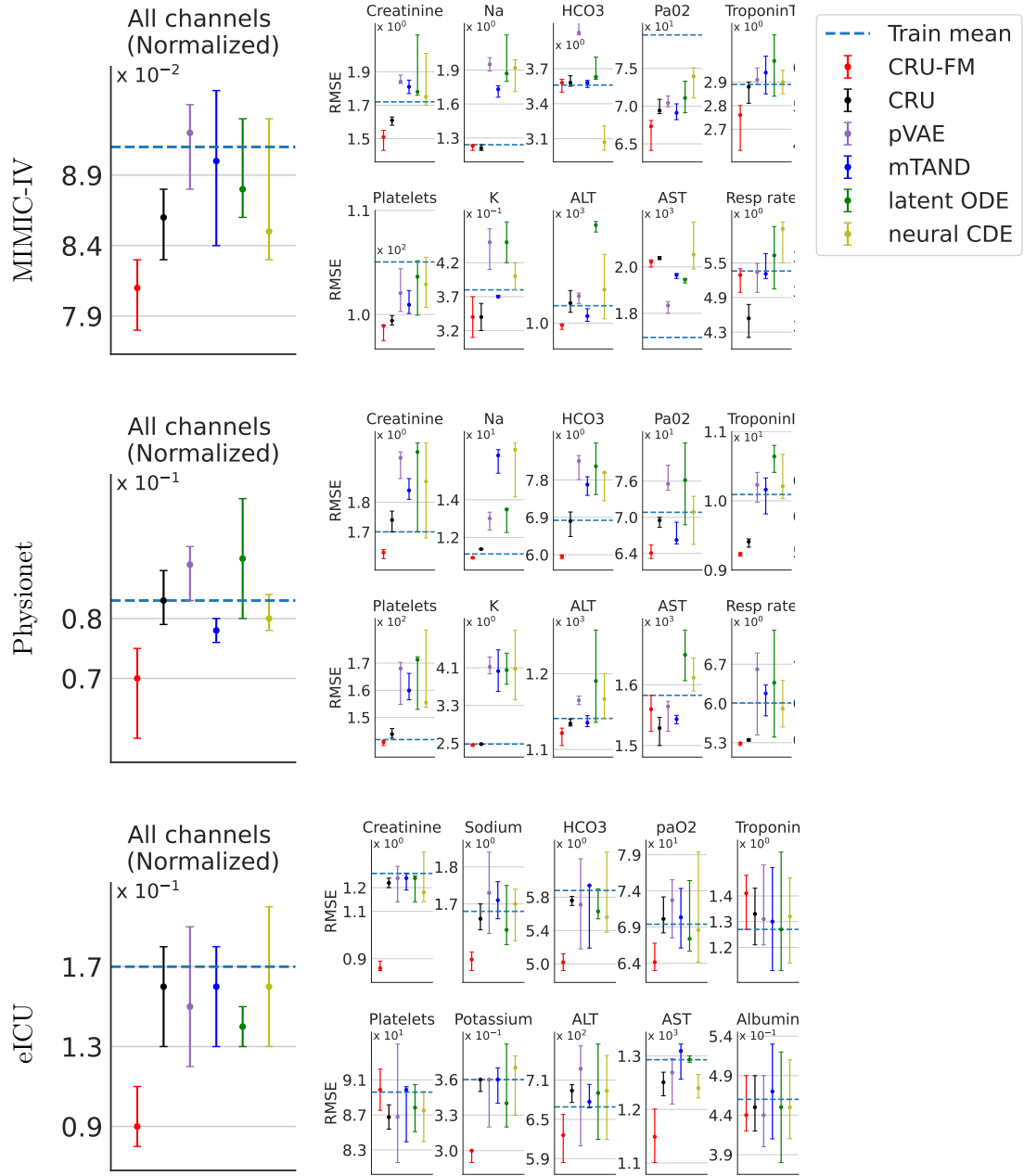


Figure 5.5: Comparison of our CRU-FM with 5 competitor methods, in terms of extrapolation error (RMSE, lower is better) on MIMIC-IV, Physionet, and eICU test sets. For each test patient-stay, the first 24 hours of data are provided as input, and methods must extrapolate the next 6-24 hours of available measurements. Confidence intervals show the 5th and 95th percentiles of the mean RMSE across 100 bootstrapped samples of the test set. Our CRU-FM clearly outperforms competitors in overall error across all channels (left column). Per-channel error comparisons (right columns) for select channels also show clear wins in a majority of cases (CRU-FM’s confidence interval is lower and does not overlap with others). Expanded per-channel results for all datasets in App. Fig. 5.6

Feature	Min	Max	Observation rate per hour (averaged over first 48 hours)
Weight	0.0	300	0.640
Albumin	0.00	4.70	0.012
ALP	0	1024	0.016
ALT	0	11030	0.016
AST	0	16200	0.016
Bilirubin	0	45.8	0.016
BUN	0	170	0.071
Cholesterol	0	240	0.002
Creatinine	0	16	0.072
DiasABP	0	268	0.768
FiO2	0	1	0.170
GCS	0	15	0.327
Glucose	0	1075	0.067
HCO3	0	42	0.070
HCT	0	61.8	0.095
HR	0	211	1.170
K	0	9.4	0.075
Lactate	0	31	0.043
Mg	0	7.7	0.070
MAP	0	300	0.757
Mechvent	0	1	0.162
Na	0	165	0.070
NiDiasABP	0	161	0.497
NIMAP	0	165.3	0.493
NiSysABP	0	296	0.500
PaC02	0	98	0.122
Pa02	0	500	0.121
pH	0	187	0.127
platelets	0	1047	0.074
RespRate	0	95	0.277
Sa02	0	100	0.041
SysABP	0	278	0.768
Temp	-17.8	42.2	0.454
TroponinI	0	46.2	0.002
TroponinT	0	24.9	0.010
Urine	0	11000	0.713
WBC	0	528	0.067

Table 5.1: List of features for Physionet

series, we include a simpler baseline that always predicts the mean of the training set values for each channel. This baseline predicts the same future for all patients and thus is not clinically useful. Our all-channel results suggest surprisingly that

Feature	Min	Max	Observation rate per hour (averaged over first 48 hours)
CRP	0.4	286	0.0005
Bilirubin (direct)	0.1	19.7	0.0015
Venous CO2 pressure	20	97	0.008
Fibrinogen	71.39	792.22	0.008
Glucose (blood)	68	313	0.024
pH (venous)	7.09	7.53	0.011
Height	135	193	0.012
Albumin	1.7	4.5	0.008
TroponinT	0.01	10.40	0.011
Resp rate	8	35	0.014
PaO2	51	478	0.037
PaCO2	22	78	0.037
pH (Arterial)	7.09	7.53	0.039
Bilirubin (total)	0.2	25.28	0.016
AST	10	6958.04	0.016
ALT	5	5200	0.016
Weight	43.4	158.9	0.014
Lactic acid	0.6	15	0.033
PT	10.2	52.7	0.043
Calcium	6.5	10.4	0.056
Phosphorus	1.2	9.0	0.056
Magnesium	1.3	3.2	0.060
ABP (systolic)	78	178	0.700
ABP (diastolic)	34	110	0.700
Temperature	35.4	38.6	0.220
Glucose (serum)	63	399	0.061
Platelets	23	570	0.060
Hemoglobin	6.3	15.2	0.070
Creatinine	0.4	8.7	0.060
Hematocrit	19.4	45.2	0.070
Potassium	2.9	6.4	0.070
HCO3	10	37	0.060
Chloride	85	121	0.070
Sodium	120	154	0.070
Resp Rate	8	35	0.950
O2 sat	88	100	0.940
Heart rate	49	133	0.960

Table 5.2: List of features for MIMIC-IV

some competitor methods do not always surpass even this baseline. Previous studies [Rubanova et al., 2019b, Shukla and Marlin, 2021] have overlooked such simple baselines, focusing solely on comparing neural network methods. We include it to

Feature	Min	Max	Observation rate per hour (averaged over first 48 hours)
TroponinT	0.01	11.8	0.002
Bilirubin	0.01	12.8	0.002
ALT	6.00	2565.8	0.017
AST	8.00	3773.2	0.017
Albumin	1.30	4.30	0.019
HCO3	9.00	44.09	0.022
PaO2	37.0	447.0	0.023
Platelets	27.0	504.0	0.047
Creatinine	0.36	9.07	0.054
Glucose	64.0	442.0	0.056
Sodium	119.0	156.0	0.059
Potassium	2.7	6.2	0.065

Table 5.3: List of features for eICU

demonstrate that existing complex models frequently do not outperform such basic approaches, indicating a key need for improvement in the present landscape of irregular time series methods we hope our method fills.

5.5.2.0.2 Results: Individual feature extrapolation. Other panels in Fig. 5.5 each show RMSE for a specific vital sign or lab common across datasets. We report such RMSE values in the corresponding variable’s original units (un-normalized) to help contextualize any gains shown on a physiological scale more familiar to clinicians. Not all vitals or labs are shown here due to space constraints; see Figure 5.6 for results on an extended list of channels beyond those in Figure 5.5. Across the extended channels shown in Fig. 5.6, we find the CRU-FM model achieves the clearly lowest RMSE (the red interval lies below the interval for any other method by visual inspection) in 11/16 features from physionet, 8/16 from MIMIC-IV, and 7/12 from eICU. In several other cases, the CRU-FM is essentially “tied” with one or more other methods. These per-channel results, together with the CRU-FM’s overall win, suggest the value of our approach.

CRU-FM does not seem clearly superior to alternatives on channels such as respiratory rate that are collected more frequently, perhaps because they are less likely to be missing and there is less need for a data-dependent mechanism. However, for lab measurements such as Creatinine, Troponin, ALT that are measured less

frequently, the CRU-FM outperforms other baselines in at least 2/3 datasets.

5.5.2.0.3 Qualitative visuals of extrapolations. We illustrate examples of extrapolation on 3 randomly picked patient stays in Figure 5.6. These plots illustrate improved extrapolations (and interpolations) for our CRU-FM compared to the CRU.

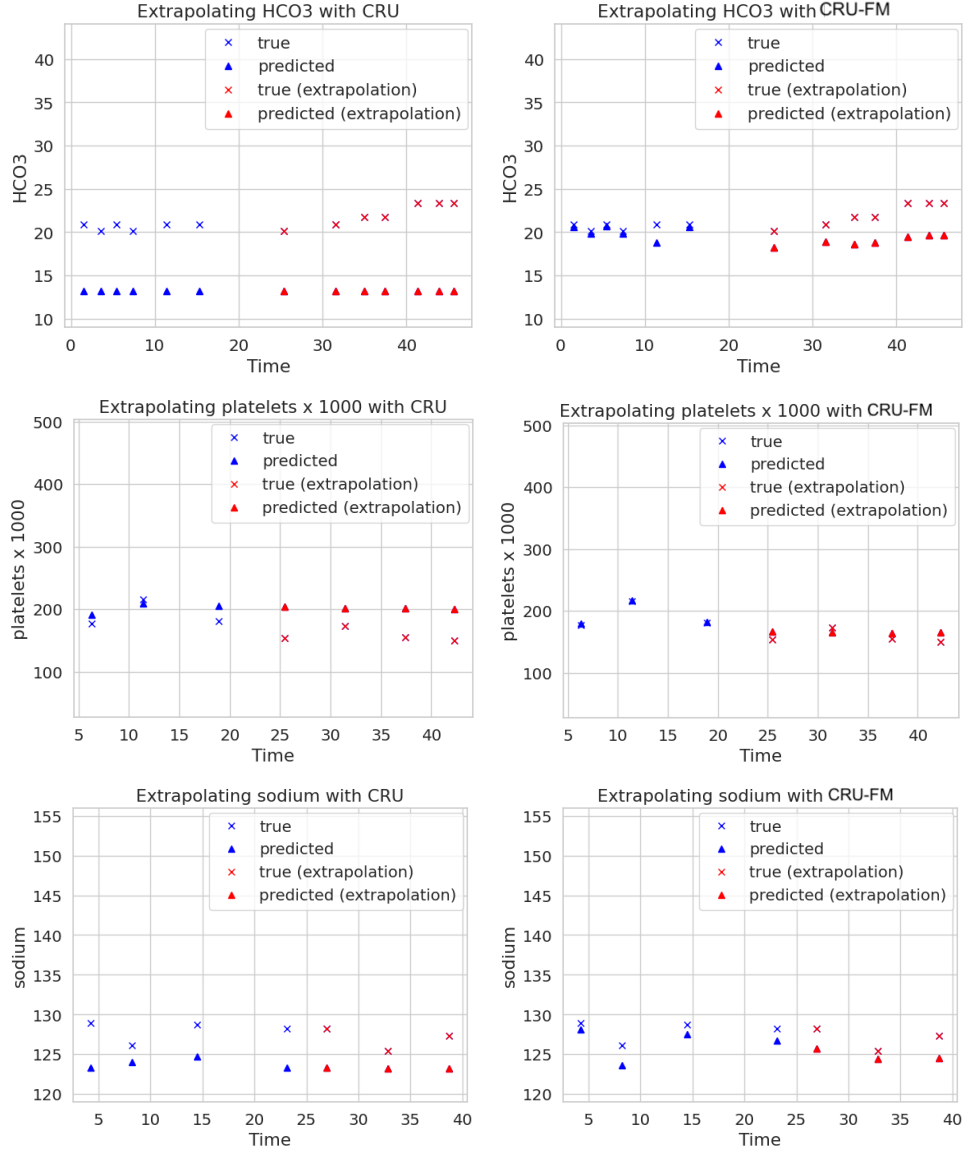


Figure 5.6: Illustrative examples of predicted trajectories from the eICU dataset, demonstrating that our CRU-FM outperforms the CRU in both extrapolation and interpolation of sequences.

5.5.2.0.4 Alternative method: CRU for concatenation of features and missingness mask.

To confirm that improvements in our model stem from its ability to learn the mechanism of missing data, we introduced another baseline in which the missingness indicator mask s_t is incorporated into the CRU as extra input feature channels. Below, we report results comparing our CRU-FM with the vanilla CRU with both features and missingness masks as inputs.

Lab Measurement	CRU-FM (test RMSE)	CRU w/ missing mask (test RMSE)
creatinine	0.86 (0.85, 0.89)	1.25 (1.22, 1.29)
potassium	0.30 (0.29, 0.30)	0.38 (0.36, 0.41)
Sodium	1.55 (1.52, 1.57)	1.77 (1.70, 1.82)
Platelets x 1000	89.86 (87.53, 92.22)	91.75 (89.66, 93.29)
Glucose	46.81 (45.00, 48.63)	53.66 (53.10, 55.15)
HCO ₃	5.02 (4.92, 5.12)	6.01 (5.88, 6.18)
paO ₂	64.16 (63.00, 66.78)	73.22 (69.04, 77.21)
albumin	0.44 (0.42, 0.49)	0.51 (0.48, 0.53)
ALT	625.80 (584.00, 657.63)	696.52 (688.74, 705.81)
AST	1148.96 (1100.80, 1200.86)	1248.19 (1236.01, 1271.91)
Bilirubin	3.03 (2.75, 3.19)	3.80 (3.65, 3.84)
Troponin	1.41 (1.27, 1.48)	1.31 (1.20, 1.44)

Table 5.4: Comparing Test RMSE of CRU (w/missing mask as input) and our CRU-FM on the eICU dataset. In 11 out of 12 channels, our proposed method does better in terms of test RMSE.

5.5.2.0.5 Predicting future missingness of a channel.

We additionally conduct an experiment on the eICU and MIMIC-IV datasets assessing how jointly modeling features and missingness can predict future missingness of one particular channel (creatinine), chosen because it is infrequently observed and as a blood draw is likely to depend on data-driven decision-making.

For each sequence in the test set : In interval $24h < t'' < 24h + w$ (window length)
: Predict $s_{t'', creatinine} \in \{0, 1\}$ (predict if a measurement will be observed)

We trained a linear classifier (L2-penalized logistic regression) for 4 distinct scenarios that vary the input features fed into the classifier listed below:

- MCAR: input feature is just p , where p is the patient-specific rate of observing a creatinine measurement in the first 24 hours (before the prediction interval)

- Last observation only (MAR simple): classify $s_{t'', creatinine}|X_{obs}, t'$, where $t' \in \mathbb{R}^D$ and $X_{obs}, t' \in \mathbb{R}^D$ is the last observed time point and measurement for each channel before 24 hours.
- Modeling features only (MAR CRU) : classify $s_{t'', creatinine}|\mu_{t'_{24}}$, where $\mu_{t'_{24}}$ refers to the representation vector at time t=24 produced by the CRU model.
- Modeling features + missingness (MAR CRU-FM) : classify $s_{t'', creatinine}|\mu_{t_{24}}$, where $\mu_{t_{24}}$ refers to the representation vector at time t=24 produced by our model

Below for each model and possible window length, we report AUROC intervals computed from 100 bootstrapped samples on test set.

Dataset	Window Len.	MCAR	MAR Simple	MAR CRU	MAR CRU-FM
eICU	2 hours	0.61 (0.57, 0.62)	0.61 (0.59, 0.61)	0.64 (0.63, 0.66)	0.70 (0.69, 0.72)
	4 hours	0.62 (0.58, 0.63)	0.62 (0.60, 0.62)	0.65 (0.63, 0.67)	0.72 (0.70, 0.75)
	6 hours	0.64 (0.60, 0.64)	0.64 (0.62, 0.65)	0.68 (0.66, 0.69)	0.73 (0.71, 0.74)
MIMIC-IV	2 hours	0.64 (0.63, 0.66)	0.65 (0.63, 0.66)	0.67 (0.66, 0.69)	0.72 (0.69, 0.73)
	4 hours	0.67 (0.64, 0.69)	0.67 (0.66, 0.67)	0.69 (0.68, 0.70)	0.74 (0.70, 0.74)
	6 hours	0.70 (0.69, 0.72)	0.68 (0.66, 0.69)	0.69 (0.68, 0.71)	0.75 (0.72, 0.76)

Table 5.5: The table presented above clearly illustrates the superiority of our model’s approach—utilizing our CRU-FM’s representation vector for predicting the likelihood of a measurement occurring within a specified window, yields better AUROC compared to the alternative methods of relying solely on past observations or the simplistic strategy of utilizing a patient-specific observation rate (MCAR). Additionally, the improvement in AUROC as the window length increases can be attributed to the increase in ground truth positives.

5.5.2.0.6 Extrapolation performance on extended list of channels In figure 5.6, we report the extrapolation performance on an extended list of channels beyond those reported in figure 5.5.

5.6 Conclusion

We present a novel approach to modeling irregular time series data that can learn mechanisms for *data-dependent missingness*. On all 3 large EHR datasets studied, our improved handling of missingness yields noticeably lower overall extrapolation

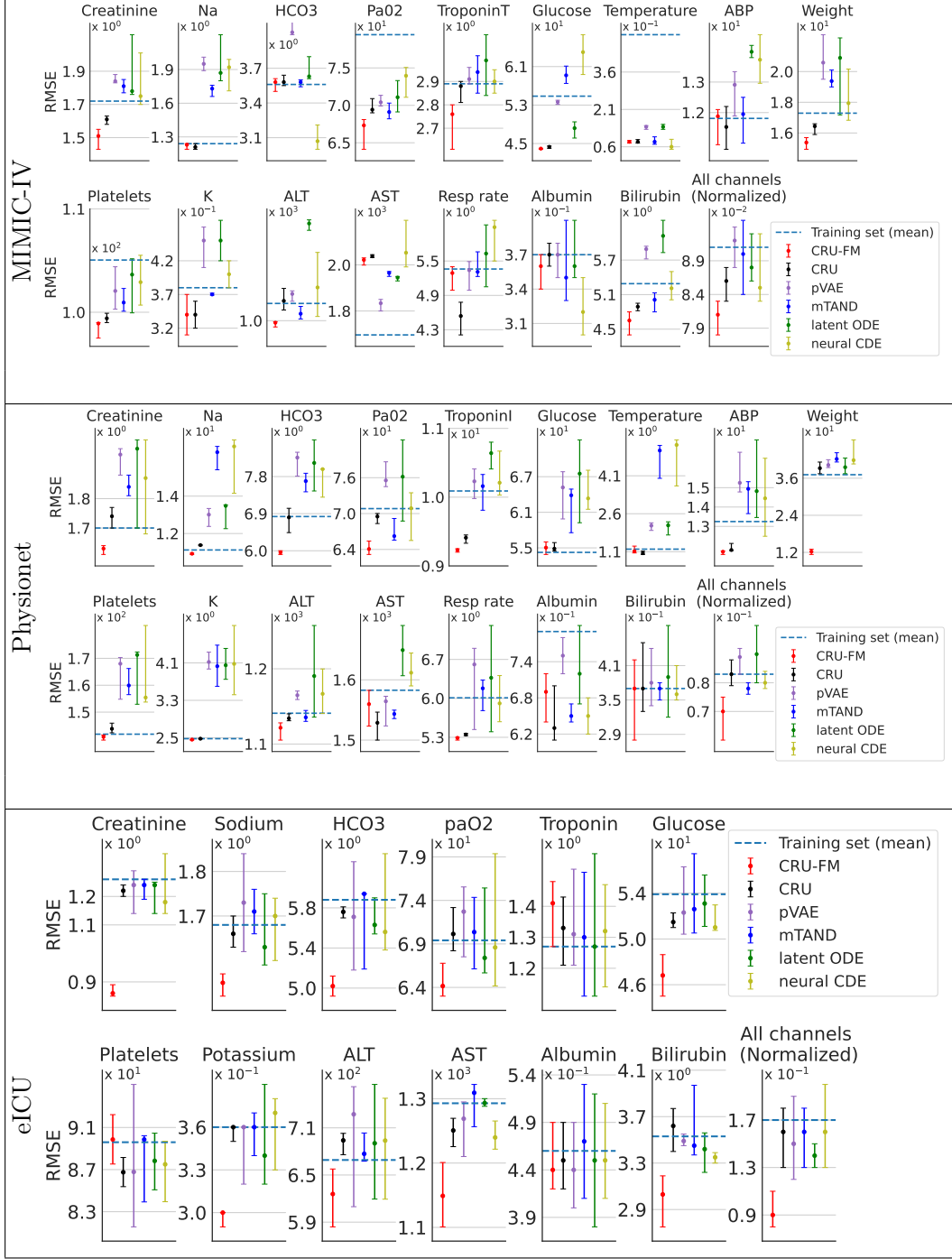


Table 5.6: RMSE comparison of our CRU-FM with 5 baselines after training on the first 24 hours of data and extrapolating the next 24 hours of measurements on MIMIC-4, Physionet, and eICU. Confidence intervals represent the 5th and 95th percentiles from 100 bootstrapped samples from the test set. Our CRU-FM outperforms baselines in 8/16 channels in MIMIC-4, 11/16 channels on Physionet and 7/12 channels in eICU (RMSE confidence intervals are lower and do not overlap with baselines). The training set mean baseline is added to uncover a gap in existing studies [Rubanova et al., 2019b, Shukla and Marlin, 2021], which overlooked a simple yet effective benchmark that, while showing decent RMSE performance, falls short in complex clinical decision-making settings.

error than several recent competitors. While joint modeling of missingness indicators and observed features could be accomplished with many model architectures, by building upon Schirmer et al. [2022]’s CRU backbone our implementation benefits from efficient computations and avoids the need for expensive ODE/SDE solvers. Future work could investigate the even more flexible missing-not-at-random (MNAR) setting for irregular time series; though empirically every MNAR model has a MAR counterpart with equal fit [Molenberghs et al., 2008] and thus the choice of MNAR requires belief in the underlying model assumptions not easily confirmed via empirical error analysis alone. More research is also necessary to ensure that our model’s superior extrapolation capabilities translate effectively into enhanced clinical understanding and improved patient care. Additionally, while our comparisons are currently limited to other irregular time series models, examining how our approach stacks up against different types of models could provide broader insights. Lastly, identifying which specific vitals or lab results our methods most effectively predict remains an open question.

Chapter 6

Conclusion and Outlook

In this dissertation, we developed robust machine learning models aimed at improving the monitoring for hospitalized patients. This exploration is particularly crucial given the typical challenges faced in clinical settings, such as data missingness, lack of detailed labels, and high rates of false alarms. These factors often hinder the effective deployment of machine learning technologies in healthcare.

Throughout the thesis, we introduced methodologies and models designed to address these pervasive issues. Chapter 3 begins with the optimization of early warning classifiers to control false alarms through a precision constraint, enhancing their ability to mitigate alarm fatigue [Sendelbach and Funk, 2013] and reduce distrust on early warning systems amongst clinicians. Following this, in chapter 4 we introduce a semi-supervised Hidden Markov Model to effectively handle time series data with missing values, proving its efficacy in realistic hospital scenarios. Further, this thesis explores the management of missingness in irregular time series, a common occurrence in EHRs, by proposing a generative model that learns the data-dependent missingness mechanisms.

The empirical evaluations presented demonstrate the significant improvements these models provide over traditional approaches, particularly in their ability to handle large-scale health records datasets effectively. We also leverage some of our models to not only deliver improved performance in predicting patient outcomes but also provide interpretable insights, which are crucial for clinical decision-making.

The methodologies presented in this thesis enhance the accuracy and reliability of patient monitoring systems, potentially leading to better patient monitoring through timely and precise medical interventions. These innovations pave the way for future research to explore further integrations of machine learning in healthcare, focusing on transparency, scalability, and fairness across diverse patient cohorts.

6.0.1 Limitations and Future Work

While the methodologies developed in this thesis represent significant advancements in healthcare monitoring using machine learning, they come with inherent limitations that may impact their deployment and effectiveness in clinical settings.

Interpretability to enhance adoption by clinicians: While we make efforts to incorporate interpretability via the PC-HMM’s learned states, it remains challenging to interpret the underlying dynamics learned by the continuous time models like the CRU-FM. While extrapolative performance improves with the CRU-FM, it remains unclear as to which specific channels would benefit from the data-dependent missingness assumption. Additionally, while models adhere to pre-specified false alarm rate on the entire population via our sigmoid objective, the performance on individual subgroups remains under-explored. Further questions remain unanswered such as what specific channel at which specific time point triggers an alert? Incorporating methodologies from the XAI literature [Enguehard, 2023, Lakkaraju et al., 2017, Tonekaboni et al., 2020] would enable adoption of these technologies, as healthcare providers may relate the model’s interpretation to their clinical knowledge.

Missing at random assumption for missing data in irregular time series: Although the CRU-FM address missing data and irregular time series, the assumptions made (such as Missing At Random (MAR)) may not always hold true in real-world clinical environments. This discrepancy can lead to biases in predictions and affect the models’ effectiveness in live clinical settings. Moreover, missingness may sometimes depend on unobserved data, termed as missing-not at random (MNAR). Future work could extend probabilistic models such as those proposed by Kim et al. [2023] and Fouladgar and Främling [2020] to handle irregular timing of measurements.

Generalizability and fairness across diverse population sub-groups: To enhance the robustness and applicability of the models across varied healthcare environments, future research should prioritize cross-validation studies that span diverse medical settings. This approach will help to ensure that the models perform consistently well across different populations and clinical practices. Additionally, it is crucial to

integrate fairness directly into the model development process. Techniques such as model desensitization [Boughorbel et al., 2021] and the incorporation of fairness-specific constraints [Agarwal et al., 2018] are essential to ensure that these models serve all patient sub-groups equitably, promoting widespread clinical adoption and trust in machine learning-driven healthcare solutions.

Optimizing Compute Utilization in Model Training: The methodologies developed in this research can sometimes converge to local optima, necessitating the use of robust computational infrastructure capable of parallel processing across multiple cores for training with various random initializations. This dependency could be mitigated through smarter initialization strategies that reduce the reliance on extensive hardware resources. For instance, our experiments demonstrated that initializing models using binary cross-entropy before transitioning to a sigmoid-based objective significantly enhanced performance compared to starting with random initializations. Furthermore, adopting programming frameworks like JAX [Bradbury et al., 2018] instead of TensorFlow for models like the PC-HMM can substantially improve model training in resource-constrained environments. JAX offers a lower memory footprint and superior compilation efficiency, which are crucial in settings with limited hardware capabilities. By streamlining the computational demands through these approaches, future research can focus on creating more accessible and efficient modeling techniques that maintain high performance without requiring extensive computational infrastructure. This shift not only optimizes resource usage but also extends the feasibility of deploying advanced machine learning models to a broader range of environments.

Upgrading models to incorporate various modalities of data: The focus of our current models is predominantly on processing standard measurements like laboratory results and vital signs. However, there is substantial potential for future enhancements that could integrate a wider range of data modalities. Models like RAIM Xu et al. [2018] offer promising frameworks that could be adapted to include high-frequency data streams such as electrocardiograms and pulse plethysmographs. Additionally, expanding these models to incorporate other essential Electronic Health

Record (EHR) components like medications, clinical notes, and imaging data—such as X-rays and CT scans—could dramatically enrich the model’s utility and accuracy.

Through addressing these limitations and exploring these avenues for future research, the field can move towards more robust, interpretable, and equitable health-care solutions that leverage the full potential of machine learning in hospitalized patient monitoring.

Bibliography

- Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. A reductions approach to fair classification. In *International conference on machine learning*, pages 60–69. PMLR, 2018.
- Yuri Ahuja, Jun Wen, Chuan Hong, Zongqi Xia, Sicong Huang, and Tianxi Cai. A semi-supervised adaptive markov gaussian embedding process (samgep) for prediction of phenotype event times using the electronic health record. *Scientific reports*, 12(1):17737, 2022.
- Md Hasan Al Banna, Kazi Abu Taher, M Shamim Kaiser, Mufti Mahmud, Md Sazadur Rahman, ASM Sanwar Hosen, and Gi Hwan Cho. Application of artificial intelligence in predicting earthquakes: state-of-the-art and future challenges. *IEEE Access*, 8:192880–192923, 2020.
- Abdul Fatir Ansari, Alvin Heng, Andre Lim, and Harold Soh. Neural continuous-discrete state space models for irregularly-sampled time series, 2023.
- Christoph Hoog Antink, Steffen Leonhardt, and Marian Walter. Reducing false alarms in the icu by quantifying self-similarity of multimodal biosignals. *Physiological measurement*, 37(8):1233, 2016.
- Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.
- Wan-Tai M Au-Yeung, Ashish K Sahani, Eric M Isselbacher, and Antonis A Armoundas. Reduction of false alarms in the intensive care unit using an optimized machine learning based approach. *NPJ digital medicine*, 2(1):1–5, 2019.
- Ayşe Demir Baha, Türkan Fendoğlu, Zeynep Işıkdogan, Nurdan Köktürk, Hatice Kılıç, Nalan Ogan, Evrim Eylem Akpınar, and Aslıhan Alhan. The effect of blood urea nitrogen/albumin ratio in the short-term prognosis of chronic obstructive pulmonary disease. *Erciyes Medical Journal*, 2021.
- David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. *arXiv preprint arXiv:1905.02249*, 2019a.
- David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. MixMatch: A Holistic Approach to Semi-Supervised Learning. In *Advances in Neural Information Processing Systems*, 2019b. URL <http://arxiv.org/abs/1905.02249>.
- Dimitri P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1 edition, 1996. ISBN 1886529043.
- Ioana Bica, Ahmed M Alaa, Craig Lambert, and Mihaela Van Der Schaar. From real-world patient data to individualized treatment effects using machine learning: current and future methods to address underlying challenges. *Clinical Pharmacology & Therapeutics*, 109(1):87–100, 2021.

- Sabri Boughorbel, Fethi Jarray, and Abdou Kadri. Fairness in tabnet model by disentangled representation for the prediction of hospital no-show. *arXiv preprint arXiv:2103.04048*, 2021.
- James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL <http://github.com/google/jax>.
- Niels Bruun Ipsen, Pierre-Alexandre Mattei, and Jes Frellsen. not-miwa: Deep generative modelling with missing not at random data. *arXiv e-prints*, pages arXiv–2006, 2020.
- Christopher Burges, Robert Ragno, and Quoc Le. Learning to rank with nonsmooth cost functions. *Advances in neural information processing systems*, 19:193–200, 2006.
- Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. Brits: Bidirectional recurrent imputation for time series. *Advances in neural information processing systems*, 31, 2018a.
- Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. BRITS: Bidirectional Recurrent Imputation for Time Series. In *Advances in Neural Information Processing Systems*, 2018b. URL <https://papers.nips.cc/paper/2018/file/734e6bfcd358e25ac1db0a4241b95651-Paper.pdf>.
- Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 161–168, 2006.
- Marie-Christine Chambrin. Alarms in the intensive care unit: how can the number of false alarms be reduced? *Critical Care*, 5(4):1–5, 2001.
- Vinod Kumar Chauhan, Anshul Thakur, Odhran O’Donoghue, and David Andrew Clifton. Coper: Continuous patient state perceiver. In *2022 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*, pages 1–4. IEEE, 2022.
- 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, 2018a.
- 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), 2018b.
- Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014a.
- Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations

- using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014b.
- Edwin Kah Pin Chong and Stanislaw H. Żak. Chapter 23: Algorithms for Constrained Optimization. In *An Introduction to Optimization*, Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley, New York, fourth edition edition, 2013.
- Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.
- Maria Cvach. Monitor alarm fatigue: an integrative review. *Biomedical instrumentation & technology*, 46(4):268–277, 2012.
- Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. Gru-ode-bayes: Continuous modeling of sporadically-observed time series. *Advances in neural information processing systems*, 32, 2019a.
- Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019b.
- Shuchisnigdha Deb and David Claudio. Alarm fatigue and its influence on staff performance. *IIE Transactions on Healthcare Systems Engineering*, 5(3):183–196, 2015.
- Elad Eban, Mariano Schain, Alan Mackey, Ariel Gordon, Ryan Rifkin, and Gal Elidan. Scalable Learning of Non-Decomposable Objectives. In *Artificial Intelligence and Statistics*, pages 832–840, 2017. URL <http://proceedings.mlr.press/v54/eban17a.html>.
- Linda M Eerikäinen, Joaquin Vanschoren, Michael J Rooijakkers, Rik Vullings, and Ronald M Aarts. Reduction of false arrhythmia alarms using signal selection and machine learning. *Physiological measurement*, 37(8):1204, 2016.
- Joseph Enguehard. Learning perturbations to explain time series predictions. In *International Conference on Machine Learning*, pages 9329–9342. PMLR, 2023.
- Gabriel J Escobar, Vincent X Liu, Alejandro Schuler, Brian Lawson, John D Greene, and Patricia Kipnis. Automated identification of adults at risk for in-hospital clinical deterioration. *New England Journal of Medicine*, 383(20):1951–1960, 2020.
- Rizal Fathony and Zico Kolter. AP-perf: Incorporating generic performance metrics in differentiable learning. In *International Conference on Artificial Intelligence and Statistics*, pages 4130–4140. PMLR, 2020.
- Chrystinne Fernandes, Simon Miles, Carlos José Pereira Lucena, et al. Detecting false alarms by analyzing alarm-context information: Algorithm development and validation. *JMIR Medical Informatics*, 8(5):e15407, 2020.

- Nazanin Fouladgar and Kary Främling. A novel lstm for multivariate time series with massive missingness. *Sensors*, 20(10):2832, 2020.
- Joseph Futoma, Sanjay Hariharan, and Katherine Heller. Learning to Detect Sepsis with a Multitask Gaussian Process RNN Classifier. In *International Conference on Machine Learning*, 2017. URL <http://arxiv.org/abs/1706.04152>.
- Yue Gao, Guang-Yao Cai, Wei Fang, Hua-Yi Li, Si-Yuan Wang, Lingxi Chen, Yang Yu, Dan Liu, Sen Xu, Peng-Fei Cui, et al. Machine learning based early warning system enables accurate mortality risk prediction for covid-19. *Nature communications*, 11(1):5033, 2020.
- Zoubin Ghahramani. An introduction to hidden Markov models and Bayesian networks. *International Journal of Pattern Recognition and Machine Intelligence*, 15(01):9–42, 2001.
- Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.
- Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.
- Jen J. Gong, Tristan Naumann, Peter Szolovits, and John V. Guttag. Predicting clinical outcomes across changing electronic health record systems. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 1497–1505, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450348874. doi: 10.1145/3097983.3098064. URL <https://doi.org/10.1145/3097983.3098064>.
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.
- Jann Goschenhofer, Rasmus Hvingelby, David Rügamer, Janek Thomas, Moritz Wagner, and Bernd Bischl. Deep semi-supervised learning for time series classification. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 422–428. IEEE, 2021.
- Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. *arXiv preprint arXiv:2310.07820*, 2023.
- Hayr 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, 2019a.

- 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, 2019b.
- Ethan Harvey, Junzi Dong, Erina Ghosh, and Ali Samadani. A comparative analysis of machine learning models for early detection of hospital-acquired infections. *arXiv preprint arXiv:2311.09329*, 2023.
- Gal Hever, Liel Cohen, Michael F O’Connor, Idit Matot, Boaz Lerner, and Yuval Bitan. Machine learning applied to multi-sensor information to reduce false alarm rate in the icu. *Journal of clinical monitoring and computing*, pages 1–14, 2019.
- Gabriel Hope, Michael C. Hughes, Finale Doshi-Velez, and Erik B. Sudderth. Prediction-Constrained Hidden Markov Models for Semi-Supervised Classification. In *Time Series Workshop at ICML 2021*, 2021. URL https://www.michaelchughes.com/papers/HopeEtAl_TimeSeriesWorkshopAtICML_2021.pdf.
- Zhe Huang, Ruijie Jiang, Shuchin Aeron, and Michael C Hughes. Accuracy versus time frontiers of semi-supervised and self-supervised learning on medical images. *arXiv preprint arXiv:2307.08919*, 2023.
- Michael C. Hughes, Leah Weiner, Gabriel Hope, Thomas H. McCoy Jr., Roy H. Perlis, Erik B. Sudderth, and Finale Doshi-Velez. Prediction-Constrained Training for Semi-Supervised Mixture and Topic Models. *arXiv:1707.07341 [cs, stat]*, July 2017. URL <http://arxiv.org/abs/1707.07341>. arXiv: 1707.07341.
- Michael C Hughes, Gabriel Hope, Leah Weiner, Thomas H McCoy Jr, Roy H Perlis, Erik B Sudderth, and Finale Doshi-Velez. Semi-supervised prediction-constrained topic models. In *AISTATS*, pages 1067–1076, 2018.
- Stephanie L Hyland, Martin Faltys, Matthias Hüser, Xinrui Lyu, Thomas Gumbsch, Cristóbal Esteban, Christian Bock, Max Horn, Michael Moor, Bastian Rieck, et al. Early prediction of circulatory failure in the intensive care unit using machine learning. *Nature medicine*, 26(3):364–373, 2020a.
- Stephanie L. Hyland, Martin Faltys, Matthias Hüser, Xinrui Lyu, Thomas Gumb-sch, Cristóbal Esteban, Christian Bock, Max Horn, Michael Moor, et al. Early prediction of circulatory failure in the intensive care unit using machine learning. *Nature Medicine*, 26(3):364–373, 2020b.
- Niels Bruun Ipsen, Pierre-Alexandre Mattei, and Jes Frellsen. not-miwae: Deep generative modelling with missing not at random data. *arXiv preprint arXiv:2006.12871*, 2020.
- Niels Bruun Ipsen, Pierre-Alexandre Mattei, and Jes Frellsen. How to deal with missing data in supervised deep learning? In *ICLR 2022-10th International Conference on Learning Representations*, 2022.
- Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021.

- Daniel Jarrett, Jinsung Yoon, Ioana Bica, Zhaozhi Qian, Ari Ercole, and Mihaela van der Schaar. Clairvoyance: A pipeline toolkit for medical time series. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=xnC8YwKUE3k>.
- AH Jazwinski. *Stochastic processes and filtering theory*. ACADEMIC PRESS, INC., 1970.
- Junteng Jia and Austin R Benson. Neural jump stochastic differential equations. *Advances in Neural Information Processing Systems*, 32, 2019.
- Thorsten Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 377–384, 2005.
- Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. Mimic-iv. *PhysioNet*. Available online at: [https://physionet.org/content/mimiciv/1.0/\(accessed August 23, 2021\)](https://physionet.org/content/mimiciv/1.0/(accessed%20August%2023,%202021)), 2020.
- Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, et al. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3, 2016.
- Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. *Advances in Neural Information Processing Systems*, 33:6696–6707, 2020.
- SeungHyun Kim, Hyunsu Kim, Eunggu Yun, Hwangrae Lee, Jaehun Lee, and Juho Lee. Probabilistic imputation for time-series classification with missing data. In *International Conference on Machine Learning (ICML)*, 2023.
- Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014a.
- Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, 2014b. URL <http://arxiv.org/abs/1412.6980>.
- Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. Interpretable & explorable approximations of black box models. *arXiv preprint arXiv:1707.01154*, 2017.
- Carson Lam, Chak Foon Tso, Abigail Green-Saxena, Emily Pellegrini, Zohora Iqbal, Daniel Evans, Jana Hoffman, Jacob Calvert, Qingqing Mao, Ritankar Das, et al. Semisupervised deep learning techniques for predicting acute respiratory distress syndrome from time-series clinical data: Model development and validation study. *JMIR Formative Research*, 5(9):e28028, 2021.
- Steven Cheng-Xian Li and Benjamin Marlin. Learning from irregularly-sampled time series: A missing data perspective. In *International Conference on Machine Learning*, pages 5937–5946. PMLR, 2020a.

- Steven Cheng-Xian Li and Benjamin M Marlin. Learning from Irregularly-Sampled Time Series: A Missing Data Perspective. In *International Conference on Machine Learning*, 2020b.
- Zachary C Lipton, Charles Elkan, and Balakrishnan Naryanaswamy. Optimal thresholding of classifiers to maximize F1 measure. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 225–239. Springer, 2014.
- Zachary C Lipton, David C Kale, and Randall Wetzel. Modeling Missing Data in Clinical Time Series with RNNs. In *Proceedings of Machine Learning for Healthcare*, page 17, 2016. URL <https://proceedings.mlr.press/v56/Lipton16.pdf>.
- Alexander Arjun Lobo, Preetish Rath, and Michael C Hughes. Semi-supervised ordinal regression via cumulative link models for predicting in-hospital length-of-stay. In *ICML 3rd Workshop on Interpretable Machine Learning in Healthcare (IMLH)*, 2023.
- Chao Ma and Cheng Zhang. Identifiable generative models for missing not at random data imputation. *Advances in Neural Information Processing Systems*, 34:27645–27658, 2021.
- Pierre-Alexandre Mattei and Jes Frelsen. Miwae: Deep generative modelling and imputation of incomplete data sets. In *International conference on machine learning*, pages 4413–4423. PMLR, 2019.
- Hongyuan Mei and Jason Eisner. The Neural Hawkes Process: A Neurally Self-Modulating Multivariate Point Process, 2017. URL <http://arxiv.org/abs/1612.09328>.
- Donald Metzler and W Bruce Croft. A markov random field model for term dependencies. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 472–479, 2005.
- Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.
- Geert Molenberghs, Caroline Beunckens, Cristina Sotito, and Michael G. Kenward. Every missingness not at random model has a missingness at random counterpart with equal fit. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 70(2), 2008. URL <http://www.jstor.org/stable/20203830>.
- S Mostafa Mousavi, Weiqiang Zhu, Yixiao Sheng, and Gregory C Beroza. Cred: A deep residual network of convolutional and recurrent units for earthquake signal detection. *Scientific reports*, 9(1):1–14, 2019.
- Harikrishna Narasimhan, Purushottam Kar, and Prateek Jain. Optimizing non-decomposable performance measures: A tale of two classes. In *International Conference on Machine Learning*, pages 199–208. PMLR, 2015.

- Bret Nestor, Matthew B. A. McDermott, Geeticka Chauhan, Tristan Naumann, Michael C. Hughes, Anna Goldenberg, and Marzyeh Ghassemi. Rethinking clinical prediction: Why machine learning must consider year of care and feature aggregation, 2018.
- Jiquan Ngiam, Zhenghao Chen, Pang W Koh, and Andrew Y Ng. Learning deep energy models. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 1105–1112, 2011.
- Pearlmutter. Learning state space trajectories in recurrent neural networks. In *International 1989 Joint Conference on Neural Networks*, pages 365–372. IEEE, 1989.
- Fernando Pineda. Generalization of back propagation to recurrent and higher order neural networks. In *Neural information processing systems*, 1987.
- Tom J Pollard and AEW Johnson III. The mimic iii clinical database, version 1.4. *The MIMIC-III Clinical Database. PhysioNet*, 2016.
- 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, 2018a. doi: 10.1038/sdata.2018.178. URL <https://doi.org/10.1038/sdata.2018.178>.
- Tom J Pollard, Alistair EW 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):1–13, 2018b.
- Raphael Poulain, Mehak Gupta, and Rahmatollah Beheshti. Few-shot learning with semi-supervised transformers for electronic health records. In *Machine Learning for Healthcare Conference*, pages 853–873. PMLR, 2022.
- Sanjay Purushotham, Chuizheng Meng, Zhengping Che, and Yan Liu. Benchmarking deep learning models on large healthcare datasets. *Journal of biomedical informatics*, 83:112–134, 2018a.
- Sanjay Purushotham, Chuizheng Meng, Zhengping Che, and Yan Liu. Benchmarking deep learning models on large healthcare datasets. *Journal of Biomedical Informatics*, 83:112–134, 2018b.
- Shameem Puthiya Parambath, Nicolas Usunier, and Yves Grandvalet. Optimizing f-measures by cost-sensitive classification. *Advances in Neural Information Processing Systems*, 27:2123–2131, 2014.
- Qi Qi, Youzhi Luo, Zhao Xu, Shuiwang Ji, and Tianbao Yang. Stochastic optimization of areas under precision-recall curves with provable convergence. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=Q_64PF6XNut.
- Lawrence Rabiner and Biinghwang Juang. An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16, 1986a.

- Lawrence R Rabiner and Biing-Hwang Juang. An introduction to hidden Markov models. *ASSP Magazine, IEEE*, 3(1):4–16, 1986b.
- Alvin Rajkomar, Eyal Oren, Kai Chen, Andrew M Dai, Nissan Hajaj, Michaela Hardt, Peter J Liu, Xiaobing Liu, Jake Marcus, Mimi Sun, et al. Scalable and accurate deep learning with electronic health records. *NPJ digital medicine*, 1(1):18, 2018.
- Alain Rakotomamonjy. Optimizing area under roc curve with SVMs. In *ROCAI*, pages 71–80, 2004.
- Elias Ramzi, Nicolas THOME, Clément Rambour, Nicolas Audebert, and Xavier Bitot. Robust and decomposable average precision for image retrieval. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL <https://openreview.net/forum?id=VjQw3v3FpJx>.
- Sebastian Raschka. Model evaluation, model selection, and algorithm selection in machine learning. *arXiv preprint arXiv:1811.12808*, 2018.
- Preetish Rath and Michael Hughes. Optimizing early warning classifiers to control false alarms via a minimum precision constraint. In *International Conference on Artificial Intelligence and Statistics*, pages 4895–4914. PMLR, 2022.
- Preetish Rath and Michael C. Hughes. Optimizing clinical early warning models to meet false alarm constraints. In *1st Workshop on Interpretable Machine Learning in Healthcare (IMLH)*, 2021.
- Preetish Rath, Gabriel Hope, Kyle Heuton, Erik B Sudderth, and Michael C Hughes. Prediction-constrained markov models for medical time series with missing data and few labels. In *NeurIPS 2022 Workshop on Learning from Time Series for Health*, 2022.
- Preetish Rath, Seth Elkin-Frankston, Amy Silder, Josiah Steckenrider, Victoria Bode, and Michael Hughes. Predicting warfighter physical performance and readiness metrics from sensor data collected during ruck march. *Americal Society of Biomechanics*, 2023.
- Santiago Romero-Brufau, Jeanne M. Huddleston, Gabriel J. Escobar, and Mark Liebow. Why the C-statistic is not informative to evaluate early warning scores and what metrics to use. *Critical Care*, 19(1), 2015.
- Yulia Rubanova, Ricky T Q Chen, and David Duvenaud. Latent ODEs for Irregularly-Sampled Time Series. In *Advances in Neural Information Processing Systems (NeurIPS)*, page 11, 2019a. URL <https://arxiv.org/abs/1907.03907>.
- Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019b.
- Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

- Simo Särkkä and Arno Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.
- Mona Schirmer, Mazin Eltayeb, Stefan Lessmann, and Maja Rudolph. Modeling irregular time series with continuous recurrent units. In *International Conference on Machine Learning*, pages 19388–19405. PMLR, 2022.
- Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112, 2017.
- Mark P Sendak, William Ratliff, Dina Sarro, Elizabeth Alderton, Joseph Futoma, Michael Gao, Marshall Nichols, Mike Revoir, Faraz Yashar, et al. Real-World Integration of a Sepsis Deep Learning Technology Into Routine Clinical Care: Implementation Study. *JMIR Medical Informatics*, 8(7), 2020.
- Sue Sendelbach and Marjorie Funk. Alarm fatigue: a patient safety concern. *AACN advanced critical care*, 24(4):378–386, 2013.
- Yooju Shin, Susik Yoon, Hwanjun Song, Dongmin Park, Byunghyun Kim, Jae-Gil Lee, and Byung Suk Lee. Context consistency regularization for label sparsity in time series. 2023.
- Satya Narayan Shukla and Benjamin Marlin. Multi-time attention networks for irregularly sampled time series. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=4c0J6lwQ4_.
- Satya Narayan Shukla and Benjamin M Marlin. Interpolation-prediction networks for irregularly sampled time series. *arXiv preprint arXiv:1909.07782*, 2019.
- Ikaro Silva, George Moody, Daniel J Scott, Leo A Celi, and Roger G Mark. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. In *2012 Computing in Cardiology*, pages 245–248. IEEE, 2012.
- Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, Chun-Liang Li, et al. FixMatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems*, 2020. URL <https://proceedings.neurips.cc/paper/2020/file/06964dce9addb1c5cb5d6e3d9838f733-Paper.pdf>.
- Mani Sotoodeh and Joyce C Ho. Improving length of stay prediction using a hidden markov model. *AMIA Summits on Translational Science Proceedings*, 2019:425, 2019.
- Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and Policy Considerations for Deep Learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy, 2019. Association for Computational Linguistics.
- Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.

- Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csd: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021.
- Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.
- Sana Tonekaboni, Shalmali Joshi, Kieran Campbell, David K Duvenaud, and Anna Goldenberg. What went wrong and when? instance-wise feature importance for time-series black-box models. *Advances in Neural Information Processing Systems*, 33:799–809, 2020.
- Xin Tong, Lucy Xia, Jiacheng Wang, and Yang Feng. Neyman-pearson classification: parametrics and sample size requirement. *Journal of Machine Learning Research*, 21(12):1–48, 2020.
- Nathan Tsoi, Yofti Milkessa, and Marynel Vázquez. A heaviside function approximation for neural network binary classification. *CoRR*, abs/2009.01367, 2021. URL <https://arxiv.org/abs/2009.01367>.
- Berk Ustun and Cynthia Rudin. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391, 2016.
- Jesper E. van Engelen and Holger H. Hoos. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440, 2020.
- Shirly Wang, Matthew B. A. McDermott, Geeticka Chauhan, Marzyeh Ghassemi, Michael C. Hughes, and Tristan Naumann. MIMIC-Extract: A data extraction, preprocessing, and representation pipeline for MIMIC-III. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, CHIL ’20, pages 222–235, New York, NY, USA, 2020a. Association for Computing Machinery.
- Shirly Wang, Matthew B. A. McDermott, Geeticka Chauhan, Michael C. Hughes, Tristan Naumann, and Marzyeh Ghassemi. MIMIC-Extract: A Data Extraction, Preprocessing, and Representation Pipeline for MIMIC-III. *Proceedings of the ACM Conference on Health, Inference, and Learning*, pages 222–235, April 2020b. doi: 10.1145/3368555.3384469. URL <http://arxiv.org/abs/1907.08322>. arXiv: 1907.08322.
- Shirly Wang, Matthew BA McDermott, Geeticka Chauhan, Marzyeh Ghassemi, Michael C Hughes, and Tristan Naumann. Code for eicu extract. https://github.com/MLforHealth/MIMIC_Extract/tree/eICU_Extract, 2020c.
- Layne T Watson. Globally convergent homotopy methods: a tutorial. *Applied Mathematics and Computation*, 31:369–396, 1989.
- Ben Wellner, Joan Grand, Elizabeth Canzone, Matt Coarr, Patrick W Brady, Jeffrey Simmons, Eric Kirkendall, Nathan Dean, Monica Kleinman, et al. Predicting Unplanned Transfers to the Intensive Care Unit: A Machine Learning Approach Leveraging Diverse Clinical Elements. *JMIR Medical Informatics*, 5(4), 2017.

- Benjamin S Wessler, Zhe Huang, Gary M Long Jr, Stefano Pacifici, Nishant Prashar, Samuel Karmiy, Roman A Sandler, Joseph Z Sokol, Daniel B Sokol, Monica M Dehn, et al. Automated detection of aortic stenosis using machine learning. *Journal of the American Society of Echocardiography*, 36(4):411–420, 2023.
- Jianwen Xie, Yang Lu, Song-Chun Zhu, and Yingnian Wu. A theory of generative convnet. In *International conference on machine learning*, pages 2635–2644. PMLR, 2016.
- Huanlai Xing, Zhiwen Xiao, Dawei Zhan, Shouxi Luo, Penglin Dai, and Ke Li. Selfmatch: Robust semisupervised time-series classification with self-distillation. *International Journal of Intelligent Systems*, 37(11):8583–8610, 2022.
- Yanbo Xu, Siddharth Biswal, Shriprasad R Deshpande, Kevin O Maher, and Jimeng Sun. Raim: Recurrent attentive and intensive model of multimodal patient monitoring data. In *Proceedings of the 24th ACM SIGKDD international conference on Knowledge Discovery & Data Mining*, pages 2565–2573, 2018.
- Kejing Yin, Ardavan Afshar, Joyce C Ho, William K Cheung, Chao Zhang, and Jimeng Sun. Logpar: Logistic parafac2 factorization for temporal binary data with missing values. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1625–1635, 2020.
- Jinsung Yoon, James Jordon, and Mihaela van der Schaar. GAIN: Missing Data Imputation using Generative Adversarial Nets. In *International Conference on Machine Learning*, 2018. URL <http://proceedings.mlr.press/v80/yoon18a.html>.
- Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 271–278, 2007.
- Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. Mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.
- Xiang Zhang, Marko Zeman, Theodoros Tsiligkaridis, and Marinka Zitnik. Graph-guided network for irregularly sampled multivariate time series. In *International Conference on Learning Representations*, 2021.
- Zhaowei Zhu, Tianyi Luo, and Yang Liu. The Rich Get Richer: Disparate Impact of Semi-Supervised Learning. *International Conference on Learning Representations (ICLR)*, 2022.

