Knowledge Distillation for Interpretable Clinical Time Series Outcome Prediction

by

Anna Wong

S.B., Computer Science and Engineering and Mathematics,
Massachusetts Institute of Technology (2022)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2023

Authored by:	Anna Wong Department of Electrical Engineering and Computer Science May 19, 2023
Certified by:	Roger G. Mark Distinguished Professor Thesis Supervisor
Certified by:	Li-wei Lehman Research Scientist Thesis Supervisor
Accepted by:	Katrina LaCurts Chair, Master of Engineering Thesis Committee

Knowledge Distillation for Interpretable Clinical Time Series Outcome Prediction

by

Anna Wong

Submitted to the Department of Electrical Engineering and Computer Science
on May 19, 2023, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

A common machine learning task in healthcare is to predict a patient's final outcome given their history of vitals and treatments. For example, sepsis is a life-threatening condition that happens when the body has an extreme response to an infection. Treating sepsis is a complicated process, and we are interested in being able to predict a sepsis patient's final outcome. Neural networks are a powerful model to make accurate predictions on such outcomes, but a major drawback of these models is that they are not interpretable. Being able to accurately predict treatment outcomes while also being able to understand the model's predictions is necessary for these models and algorithms to be used in the real world.

In this thesis, we use knowledge distillation, which is a technique for taking a model with high predictive power (known as the "teacher model"), and using it to train a model that has other desirable traits such as interpretability (known as the "student model"). For our teacher model, we use an LSTM, which is a type of neural network, to predict mortality for sepsis patients, given information about their recent history of vital signs and treatments. For our student model, we use an autoregressive hidden Markov model to learn interpretable hidden states. To incorporate the knowledge from the teacher model into the student model, we use a similarity-based constraint. We evaluate a method from a previous work that uses variational inference to learn the hidden states, and also develop and evaluate an alternative approach that uses the expectation-maximization algorithm. We analyze the interpretability of the learned states. Our results show that, although there is room for improvement in maintaining the generative performance of the model after adding the similarity constraint, the expectation-maximization algorithm is successful in incorporating the constraint to achieve high predictive power similar to the teacher model, along with better interpretability when compared to the teacher model.

Thesis Supervisor: Roger G. Mark
Title: Distinguished Professor

Thesis Supervisor: Li-wei Lehman
Title: Research Scientist

Acknowledgments

First, I'd like to thank my direct supervisor, Li-wei Lehman for all of her support. She provided so much guidance on the direction of this project, and always had useful suggestions for next steps, ideas for new approaches to try when I was stuck, and insights into the results we saw. She was always there to answer any questions I had, and I appreciate how much time she dedicated to helping me grow as a student and researcher. I'm very grateful for her patience while working with me, her clear explanations of difficult concepts, and her steady encouragement.

I would also like to thank my faculty supervisor, Dr. Roger Mark, for his suggestions and support throughout the year, and for letting me be a part of this lab. Thank you also to Ardavan Saeedi for his insights into various technical aspects of this project that helped me debug and figure out what to try next, and Adam Dejl who helped me set-up and understand the codebase.

Finally, thank you to my friends for all the memorable adventures and moral support, and my family for always supporting me and being there to listen and offer advice.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A common task for machine learning in a healthcare setting is to predict what a patient's final outcome will be. Information we might consider in the prediction process include the patient's history, their recent vital signs, their response to previous treatments, and the outcomes of similar patients. Given this information, we would like to be able to accurately predict the final outcome of a patient.

One real-world example where such a model could be useful is in sepsis outcome prediction. Sepsis is a life-threatening condition that occurs when the body has an extreme response to an infection. It's estimated that over 1.7 million people in the United States are affected by sepsis each year, and between one-third and one-sixth of those affected will die [26] [4]. One-third of patients who die in a hospital are affected by sepsis during their stay [26]. It is challenging to predict what outcome a patient will experience beccause there are many factors to consider, and in addition to considering the current information about a patient, we also need to factor in their recent history to predict how they will fare in the future.

1.1 Technical Challenges

In order to accurately predict a patient's outcome in challenging settings such as during sepsis treatment, we might be interested in training a neural network to make these predictions, and previous approaches have used these models [1], [18]. However,

although neural networks are powerful predictive models, they are not interpretable. While the models can predict outcomes with high accuracy, they do not explain which factors are most predictive of a patient's outcome. Interpretability is important in healthcare so that doctors can understand where the model's predictions are coming from, which will lead to more trust in the model. Furthermore, interpretable models can help us learn new things that we previously have not observed, such as if we discover that one of the most predictive covariates for the model is a biomarker that we previously thought was unrelated.

Latent variable models, such as hidden Markov models (HMM) and other variants of HMMs like the autoregressive HMM (AR-HMM), can be trained on clinical time series data to learn an interpretable latent structure that represents a patient's state of health over time. The limitation of these models is that the state representations are often not optimized for predicting downstream outcomes, and thus the predictions may not be as accurate as with a neural network.

1.2 Thesis Overview

In this thesis, we train a model that attempts to combine both the neural network approach and the latent variable model approach, with the goal of achieving high predictive power while having an interpretable model. This project builds on work done by Saeedi et al. [25]. In that previous work, the authors use a technique called knowledge distillation, which is used to distill the knowledge learned in a powerful "teacher" model to train a "student" model that has more desirable characteristics such as interpretability. In the case of Saeedi et al., the teacher model is an LSTM with high predictive power, and the student model is an AR-HMM that learns latent states. The authors use a technique known as automatic differentiation variational inference (ADVI) for approximate inference to learn the latent states. They then incorporate the teacher model's knowledge through the use of a similarity constraint. We denote the models created using this method as AR-HMM-ADVI.

In this thesis, we first evaluated the models developed in Saeedi et al. on a real-

world dataset (MIMIC-IV) [12]. We then developed a similar model with knowledge distillation, where we use an LSTM for our teacher model and an AR-HMM for the student model, and use a similar similarity constraint. The difference is that in our model, instead of using ADVI, we investigate an alternative approach by instead using the expectation-maximization (EM) algorithm to learn the latent states, and we also use a similarity constraint to incorporate the teacher model's knowledge so that we can make accurate clinical time series outcome predictions. We evaluate the predictive power and the interpretability of the models produced by this new approach. We denote the models created using this approach as AR-HMM-EM.

1.2.1 Contributions

The contributions of this thesis are as follows:

1. **Evaluation of the AR-HMM-ADVI models on MIMIC-IV dataset.** We evaluated and analyzed the performance of the AR-HMM-ADVI models presented in [25] on a new dataset, namely the MIMIC-IV dataset.

2. **Development of an alternative approach for incorporating similarity-based constraints.** We developed and evaluated an alternative approach for incorporating similarity-based constraints to learn AR-HMM models for clinical time series outcome prediction. Our approach is similar to the previous work in Saeedi et al., but we use the EM algorithm as an alternative to ADVI.

3. **Analysis of tradeoffs between predictive and generative performance.** Using our new models, we examined the tradeoff between the predictive and generative performance of our approach by weighting the similarity constraint, which is designed to improve predictive performance, at different amounts to evaluate its impact on the generative performance.

4. **Analysis of predictive performance and interpretability of models.** We evaluated how well our models were able to predict hospital mortality, and also examined the interpretability of the model by looking at the learned latent

states. In addition to evaluating the models' ability to predict mortality, we evaluated the performance of the model in predicting other downstream outcomes such as the development of pulmonary edema, or the need for dialysis, diuretics, or mechanical ventilation.

While the work presented in this thesis is focused on outcome prediction, we hope that our method for predicting patient outcomes can also be extended to help predict a patient's response to a specific treatment. This would be useful for physicians who are often faced with a variety of treatment options, and must choose the treatment that will be most beneficial for each patient and result in the best downstream outcome.

Chapter 2

Related Works

2.1 Outcome Prediction

Predicting clinical outcomes is a common target of machine learning research. There is often a large amount of data that we want to consider, and it would be difficult for a human to process all of that information to make an accurate prediction. Furthermore, many clinical outcomes depend on not only the patient's current state, but also their recent history. Previous work has been done for time series outcome prediction to identify patients who are at risk of sepsis shock [8].

The predictive power of neural networks makes these models a frequent choice for researchers who are interested in predicting clinical outcomes. Neural networks are able to take in a lot of data to make accurate predictions, and there are many types of neural networks that are able to evaluate time series data to make predictions, such as a long short term memory (LSTM) model, which is a type of recurrent neural network.

For example, LSTMs have been used successfully to predict the onset of congestive heart failure, and convolutional neural networks (CNNs) have been used to predict the mortality of patients with heart failure [21], [17]. Recurrent neural networks have also been used to predict various COVID outcomes including in-hospital mortality, needing mechanical ventilation, and staying the hospital for more than 7 days [23]. Both CNNs and LSTMs have been used to predict mortality outcomes for sepsis patients [3].

Park et al. evaluated different ML methods for predicting sepsis mortaliity, including random forest, gradient-boosted decision tree, and deep neural network, and while all of them were pretty accurate, the neural network had the best predictive performance [22].

However, while neural networks have been shown to be effective at a variety of outcome prediction tasks, they have the drawback of not being very interpretable. Interpretability is an important trait of models and algorithms if they are to be used in a healthcare setting, as it gives medical experts confidence in the model's predictions.

2.2 Supervised Learning with Probabilistic Graphical Models

Another common approach to learning models to predict final outcomes is a 2-stage approach. In the first stage, a graphical model is used to infer latent features for a dataset. These features are then passed into the second stage of the model, which is a discriminative model that predicts an outcome.

Examples of this method include using a graphical model to derive features that are then used as input for a linear regression to predict neuroticism and depression [24], using a switching vector autoregressive (SVAR) model to learn states that are used to predict hospital mortality [14], and deriving interpretable features from time series vital signs to detect severe sepsis with a logistic regression on these features [16].

The latent features from the graphical model make the model more interpretable than a neural network, but since the inferred features are not optimized for the prediction task, these types of models often do not have very high predictive power. Previous work has tried to resolve this issue by incorporating the labels as part of the model [19], [24]. Saeedi et al. used a knowledge distillation approach to improve performance by using a pre-trained discriminative model with high predictive ability

to help learn useful features in a more interpretable model [25].

2.3 Constrained Inference

Another approach that is used to improve model performance is constrained inference. In this approach, the posterior distribution of learned states is constrained in some way to enforce a performance constraint. For example, we can add a discriminative constraint to improve prediction while still learning interpretable states, by incorporating supervised loss as part of the loss function [10]. Instead of constraining by performance, another constraint could be to constrain the feature space of one model so that it is similar to the feature space of a pre-trained model [25]. We use this feature space constraint in this thesis.

Chapter 3

Methods

Our approach was to use knowledge distillation to develop models that are both predictive and interpretable. Knowledge distillation is a technique that uses a model with high predictive power, known as the teacher model, and distills its information into another model known as the student model, which ideally has some preferable characteristics such as interpretability. For our work, we first trained the teacher model, which was an LSTM, to learn patient hospital mortality. For the student model, we chose to use an autoregressive HMM (AR-HMM) to learn hidden states. We tried two different approaches for learning this student model. The first approach was to use a variational inference technique known as ADVI to learn the latent states of the model. The second approach used the expectation-maximization (EM) algorithm to learn the hidden states. In both approaches, while learning the AR-HMM, we trained a recognition network to learn latent states of the model while also incorporating a similarity-based constraint between the teacher model and the student model. The similarity-based constraint was used to distill knowledge from the teacher to the student.

3.1 Background

3.1.1 Knowledge Distillation

Knowledge distillation is a technique used to take the knowledge of a teacher model and distill it so it can be used in a student model. Typically, the teacher model is a more powerful model such as a neural network, but the student model has some preferable traits such as being more efficient, more generalizable, or in our case, more interpretable. The teacher model may have additional features that can also improve its predictive power. Since the feature sets may be different between teacher and student models, we cannot simply use the student model to copy the teacher model. In Saeedi et al, the authors use the observation that similar inputs should generate similar feature representations in a neural network to propose a method for knowledge distillation. Their proposed method, which we adapted for our work, is to ensure that if a pair of inputs to the teacher model produces dis(similar) feature representations, then that pair will also produce dis(similar) feature representations when given to the student model. More specifically, they create pairwise similarity matrices for the student and teacher models, where each matrix contains the pairwise distances between feature representations of the inputs to a model [25].

3.1.2 AR-HMM

In many problems such as ours, we have a time series of data and wish to be able to segment this data in terms of similar dynamics. In order to represent this, for each observation, we introduce a latent variable z. If we then use the latent variables to form a Markov chain, this forms a state space model (SSM).

A hidden Markov model (HMM) is a type of SSM where the variables are discrete (although the observed variables can be discrete or continuous). HMMs are commonly used in speech recognition and natural language modelling. However, an HMM assumes that given the latent state sequence, the observations are conditionally independent. Such a model would fail to capture important dependencies between

timesteps. In order to further account for dynamics, we can consider each latent state in the HMM to model one dynamic mode or behavior, and we can allow the model to switch between the various dynamic modes by switching between the latent states. In particular, we can use a switching vector autoregressive (SVAR) process, which differentiates between the discrete part of the state (the "mode") and the continuous part of the state, which capture the dynamics for a mode. Using this SVAR process with our HMM gives us an autoregressive HMM (AR-HMM). By switching between these linear dynamic states and behaviors, we can model more complicated dynamics [5].

In an AR-HMM, each state is parameterized by a set of AR coefficients (which define how much the previous observations affect the current observation), a covariance matrix Σ, and a bias term b, so we can define the k-th state as $\theta_k = \{A_k, \Sigma_k, b_k\}$. In a SVAR process with order r, each observation depends on the r observations before it [6]. Let $y_t^{(i)}$ be the observation vector of the i-th patient at time t, and let $z_t^{(i)}$ be the state of the corresponding Markov chain for that patient at time t. Let π_k be the transition probabilities for state k. Then, since this is a Markov chain, we know that $z_t^{(i)} \sim \pi_{z_{t-1}^{(i)}}$, for all $t > 1$. An order r SVAR process, denoted by VAR(r), is defined as follows

$$z_t^{(i)} \sim \pi_{z_{t-1}^{(i)}} \tag{3.1}$$

$$y_t^{(i)} = \sum_{l=1}^{r} A_l^{z_t^{(i)}} y_{t-l}^{(i)} + e_t^{(i)}(z_t^{(i)}) + b_{z_t^{(i)}} \triangleq A_{z_t^{(i)}} \tilde{y}_t^{(i)} + e_t^{(i)}(z_t^{(i)}) + b_{z_t^{(i)}} \tag{3.2}$$

where $e_t^{(i)}(z_t^{(i)}) \sim \mathcal{N}(0, \Sigma_{(Z_t)})$ is the state-specific noise, $A_k = [A_1^k ... A_r^k]$ are the lag matrices that indicate how much to weight previous observations, and $\tilde{y}_t^{(i)} = [y_{t-1}^{(i)\top} ... y_{t-r}^{(i)\top}]^\top$ are the previous observations [15]. The models we discuss in this thesis use an AR-HMM with AR order 1.

3.1.3 Automatic Differentiation Variational Inference

The mechanism behind many models that identify patterns in a model or make predictions is typically a computation of the posterior distribution of latent variables, given the observation. However, it is often difficult to calculate the posterior, which leads to the use of approximate inference techniques. Automatic Differentiation Variational Inference (ADVI) is an example of one such technique. It is a flexible black-box variational inference method that can be used for many different probabilistic models. In ADVI, the goal is to maximize the evidence lower bound (ELBO), which is a lower bound on the log-likelihood. In Saeedi et al., ADVI is used as part of the model to approximate the posterior of the global variables [25].

3.1.4 Expectation Maximization Algorithm

The expectation-maximization (EM) algorithm is a technique used to learn latent variable models, such as an HMM. It does this by maximizing the likelihood function of the model.

We start with an initial selection of model parameters. Each iteration of the model involves two steps - the expectation step and the maximization step. In the expectation step (E step), we use the parameter values to get the posterior distribution of the latent variables, and use this distribution to calculate the expected value of the log-likelihood. In the maximization step (M step), we find the parameters that maximize the expected log likelihood. On each iteration of E step and M step, the log-likelihood should hopefully increase and gives us new parameter values that result in the increased log-likelihood [2].

3.1.5 Recognition Network

A recognition network is used to map from observations to a posterior distribution of local latent variables. They can be used to efficiently approximate parameters or latent variables [13]. The output from a recognition network is typically used as the features that are inputted to a predictive task. In Saeedi et al., the authors use

Autoencoding Variational Bayes (AEVB) to train a recognition network that is used to infer local latent variables of the HMM [25]. In our approach, between the E- and M- steps of the EM algorithm, we train an LSTM as our recognition network to learn latent states of our model while incorporating the knowledge distillation constraint. We do so by calculating a similarity loss between the student and teacher models, and adding it to the loss function for the recognition network.

3.2 Data

In this work, we used data from the Medical Information Mart for Intensive Care IV (MIMIC-IV) database, which contains medical records from hospital admissions and ICU stays at the Beth Israel Deaconess Medical Center (BIDMC) [12]. Our dataset came from previous work done by Hu [9]. For our patient cohort, we selected patients meeting the sepsis-3 criteria. Under this criteria, a patient is defined to have sepsis if they have both an episode of suspected infection and a Sequential Organ Failure Assessment (SOFA) score of 2 or more points. An episode of suspected infection is defined as either (a) an antibiotic was given and a culture was sampled within 24 hours or (b) a culture was sampled and an antibiotic was administered within 72 hours.

The dataset excludes patients whose time of suspected infection was more than 24 hours after ICU admission. It also excludes patients who were admitted after cardiac, vascular, or trauma surgery since those surgeries pose risks that could lead to different mortality outcomes. Additionally, if a patient had more than one ICU stay, only the first stay was used. The dataset also excludes patients who did not have documented pre-ICU fluids. This was because we expect sepsis patients to get some fluid therapy before being admitted to the ICU, so not having pre-ICU fluids documented probably means it just was not properly recorded in the dataset. Patients were also excluded if they had certain outliers in their covariates, as determined by medical experts [9]. Finally, we removed patients who died within 24 hours of entering the ICU, and patients who did not have all 24 hours of data.

Table 3.1: Final Sepsis Cohort Characteristics.

Characteristic	***n***
Number of Patients	7,296
Mean age	65.10
Median age	67.0
Male	4,135
Hospital mortality	13.4%

After ensuring patients met all of these criteria, we were left with 7663 patients. We put 70% of patients in the training set, and 15% in each of the validation and testing sets. Since we used batches for our training with a batch size of 128, we removed any patients that were left out of a full batch. This left us with 5248 patients in the training set, and 1024 in each of the validation and testing sets, for a total of 7296 patients. There were 694 patients who died in the training set, 139 in the validation set, and 148 in the testing set, for a total of 981 patients who died, and a mortality rate of 13-15% within each set. More statistics about the final cohort are provided in Table 3.1.

For each patient, we have 24 hours of hourly data, with covariates such as heart rate, blood pressure, SOFA score, and other clinical variables such as glucose, creatinine, potassium, and more. We also have information about the actual treatment given to these patients at each hour, such as if patients are on mechanical ventilation or dialysis and the amount and dosage (if any) of fluids, vasopressors, and diuretics given. Finally, we have information about outcomes such as if the patient has pulmonary edema, is on diuretics, dialysis, or mechanical ventilation, or if they die in the hospital. We fill in missing covariate values by either extending the previous covariate measurement for that patient if it exists, or filling it in with the population average value for that covariate.

The presence of pulmonary edema was not included as a variable in the original MIMIC-IV dataset, but it was derived in Hu's thesis using NLP techniques on radiology reports of patient x-ray images. More specifically, chest x-ray images are frequently used to diagnose pulmonary edema, and these images are interpreted by

medical experts in radiology reports. The radiology reports in MIMIC-IV were first de-identified using an NLP technique known as bidirectional encoder representations for transformers (BERT) and a Python module called pydeid. These reports were then fed into a state-of-the-art labeler called CheXpert, which outputs 1, 0, or -1 to indicate the presence, absence, or possible presence of pulmonary edema [11]. Our dataset includes both 1 and -1 as indicating the presence of edema [9].

The dataset provided in Hu's thesis additionally measures O2 requirements, which represent levels of oxygen requirement of the patient. The value is an integer from 0 to 6, where 0 means there is no oxygen requirement, and 6 means the patient is on an invasive mechanical ventilator [9].

For our models, we used covariates similar to the ones used in Hu's thesis [9]. A full list of covariates is provided in Tables A.1 and A.2.

3.3 Teacher Model

For our teacher model, we trained an LSTM on the patient data to predict mortality. The teacher model included a total of 47 features, which is more features than the student model. The extra features included in the LSTM but not the student model are shown in Table A.2. Our hope was that the knowledge learned by the teacher model could be transferred to the student model, without the student model needing as many covariates. We tried various seeds and hyperparameter settings for this LSTM, and used the model with the best validation AUROC for our teacher model.

3.4 Student Models: AR-HMMs

For both of our student models, we used an autoregressive hidden Markov Model (AR-HMM) of order 1, with D-dimensional Gaussian distribution, where D is the number of covariates used for each patient's input to the model. In our models, $D = 34$. For each patient, we have a $D \times T$ vector input, where $T = 24$ is the number of timesteps. There are K possible latent states learned by the AR-HMM.

3.4.1 Knowledge Distillation Constraint

We incorporated the knowledge distillation constraint by using a similarity-based constraint between the teacher and student models. We used a constraint similar to the one used in Saeedi et al. [25]. Let C_t be the feature dimensionality of the teacher model, and C_s be the feature dimensionality of the student model. For a dataset of size N, we denote the feature representations of the teacher and student models as $F^t \in \mathbb{R}^{N \times C_t}$ and $F^s \in \mathbb{R}^{N \times C_s}$, respectively. For the student model, we assume that every row of the feature representation is a function of the inferred latent variables. The knowledge distillation constraint is designed to make sure that the differences between the two feature representations is less than some tolerance level.

More specifically, we compute the similarity of feature representations across patients by taking the dot product, which results in $N \times N$ matrices:

$$\tilde{F}^s = F^s \cdot F^{s\top} \text{ and } \tilde{F}^t = F^t \cdot F^{t\top} \tag{3.3}$$

We also apply a normalization to the matrices. In Saeedi et al, and for the KD-AR-HMM-ADVI models we trained, the similarity matrices were derived by first taking the dot product of the feature matrix, then normalizing by dividing each column by the ℓ^2 norm of the row. We modified this normalization for the KD-AR-HMM-EM models so that we first normalized across the columns, each of which is a covariate, and then took the dot product of the normalized matrices. Let us denote the final normalized similarity matrices as $\bar{F}^s$ and $\bar{F}^t$ for the student and teacher models, respectively. We calculate the similarity loss as

$$\text{similarity loss} = \gamma \frac{1}{N^2} ||\bar{F}^s - \bar{F}^t||^2 \tag{3.4}$$

where γ is a hyperparameter that specifies how much to weight the loss from this similarity constraint in the overall loss function.

3.5 AR-HMM-ADVI

For our first attempt at using knowledge distillation, we used the AR-HMM-ADVI developed in [25]. This model uses constrained variational inference to incorporate the knowledge distillation constraint. The baseline model was a basic AR-HMM without knowledge distillation or supervision. The model that incorporates knowledge distillation via a similarity constraint is denoted as KD-AR-HMM-ADVI. We also used a model with a discriminator constraint (DISC-AR-HMM-ADVI), and tried combining both the discriminator and similarity constraints in the KD-DISC-AR-HMM-ADVI model.

3.6 AR-HMM-EM

The majority of the work presented in this paper involves an AR-HMM fitted using the EM algorithm, denoted as AR-HMM-EM.

3.6.1 Baseline

We used the SSM package [20] to develop our HMM models with a custom EM fitting method. For the baseline, we used the package's provided EM fitting algorithm, and trained the model for 50 iterations of the expectation and maximization steps. Although we had an early stopping mechanism in place that was based on changes in log-likelihood, this mechanism was never triggered, so all 50 iterations were run. We used various seeds for the baseline model and picked the one with the highest training and validation log-likelihood to build our knowledge distillation model on top of.

3.6.2 Modified E-step

In order to incorporate the teacher model into the AR-HMM-EM, we modified the EM algorithm to factor in the similarity constraint between the AR-HMM-EM and the teacher LSTM model. We denote the resulting model as KD-AR-HMM-EM. In the modified algorithm, we trained a recognition network to output expected states

that factored in the teacher model's knowledge. The recognition network is described in more detail below. We used the expected states outputted by the recognition network as input to the M-step for that EM iteration. This modified E-step was used for up to 20 iterations of the EM algorithm (after the initial baseline was trained). However, due to an early stopping condition based on the validation AUCs of the recognition network, only 11 iterations were actually used.

Recognition Network

The recognition network was an LSTM that took the same input as the overall model, and was trained to predict the state marginals for each patient at each time step. The labels provided to the recognition network were the true state marginals, as calculated by a function provided in the SSM package on that iteration of the algorithm. Thus, the output of the recognition network has dimensions $N \times T \times K$. For this recognition network, the objective was to minimize a combination of the difference between the true and predicted state marginals (as calculated using mean squared error (MSE) loss), as well as the similarity loss between the features generated by the teacher model and the features generated by the recognition network.

When training the recognition network, we used 5-fold cross validation on the provided training data. Each of the five recognition networks was trained for 20 epochs. The recognition network with the highest validation AUC was selected as our recognition network for that step, and was used to calculate the predicted state marginals that were passed into the M-step of the model.

Algorithm 1 summarizes the training process for the KD-AR-HMM-EM. Lines 3 - 8 correspond to the modified E-step that trains the recognition network.

3.6.3 Final Outcome Prediction

Our ultimate goal behind the AR-HMM-EM models was to be able to predict final patient outcomes such as mortality. To do this, we used a logistic regression model that took as input the features outputted by the AR-HMM-EM model, and outputted

Algorithm 1: KD-AR-HMM-EM Training

```
Train baseline AR-HMM-EM for 50 iterations
for iter = 1, ..., 20 do
    for fold = 1, ..., 5 do
        for epoch = 1, ..., 20 do
            Train Recognition LSTM
        end
    end
    Select best recognition network from the 5 folds
    Run M-step on recognition network output
    Check for early stopping
end
```

the probability of an outcome such as mortality.

For the baseline EM model, the inputs to the logistic regression were the state marginals as calculated by the SSM package's built-in function. For our custom EM models, after fully training the EM with a combination of the base EM iterations and the iterations using a recognition network, we saved the recognition network from the final iteration that had the highest validation AUROC. The output from this final recognition network was then used as the input features to the logistic regression.

In addition to predicting mortality, we also used the same set of features to predict other patient outcomes such as the development of pulmonary edema, or the need for dialysis, mechanical ventilation, or diuretics.

3.6.4 Evaluation

The primary metrics we used to evaluate our models were Area under the ROC Curve (AUROC) and log-likelihood. We also have several ways of evaluating the interpretability of our models.

AUROC (AUC)

A Receiver Operating Characteristic (ROC) curve graphs the true-positive rate versus the false-positive rate for a classification model. Each point on the curve shows the true- and false- positive rates for a classification threshold. A higher area under the

curve (AUROC) means that the model does a better job of distinguishing between the two classes - an AUROC of 0.5 means the classifier is essentially randomly guessing the class, whereas an AUROC of 1.0 means the model can perfectly distinguish between the two classes. We calculated the AUROC of the final logistic regression that predicts mortality or other final outcomes.

Log-Likelihood

Log-likelihood is a metric for how well a model fits the data. A higher log-likelihood means the model is a better fit for the dataset. Unlike AUROC, there is no absolute cutoff for log-likelihood that determines if the model is "good" or "bad". Instead, we can use the log-likelihood to compare two models against each other, or to see how a model is improving over training.

Interpretability

While the previous two metrics help us determine how good our model is for predicting values, we are also interested in the interpretability of our model. To this end, we can look at the states predicted by our trained AR-HMM-EM models and compare the state distributions of patients with one outcome to patients with another outcome (e.g. comparing patients who lived versus patients who died).

Another metric we can use is the odds ratio of our final logistic regression coefficients. The odds ratio for a coefficient measures the change in the odds of one event occurring when you change the feature associated with that coefficient. We can also calculate a confidence interval for the odds ratio. An odds ratio where the upper and lower bounds are either both greater than 1 or both less than 1 indicates that the feature is significant. For example, if we are predicting mortality and the coefficient for state 1 is greater than 1, that means that being in state 1 is correlated with dying. However, it is important to note that the odds ratio can only show correlation, and does not imply causation.

Table 3.2: Hyperparameter Settings. This table shows the options for various hyperparameters that we tried for our KD-AR-HMM-EM models.

Hyperparameter	Settings
Similarity coefficient	1e1, 1e3, 1e5, 1e10, 1e30
Learning rate	0.1, 0.01, 0.001
Batch size	64, 128, 256
Number of hidden dimensions	4, 8, 16, 32
Number of hidden layers	2, 4

3.6.5 Experiment Settings

In order to tune the model, we considered adjusting various hyperparameters. One of the main hyperparameters we focused on was the similarity coefficient used to scale the similarity loss. If the coefficient was too low, then the similarity loss would have little effect on the model's predictions and essentially ignore the teacher model. However, if the coefficient was too high, the similarity loss term would dominate the overall loss of the recognition network, and the network might fail to learn the true state marginals.

In addition to the similarity coefficient, we examined different settings for hyperparameters involved in the tuning of the recognition network LSTM, including the learning rate, batch size, number of hidden dimensions, and number of hidden layers. The options we considered for each of these hyperparameters is shown in Table 3.2.

A grid search through all of these combinations was too computationally expensive. Thus, when training the KD-AR-HMM-EM model with 5 states, we first fixed the seed to be 123 and chose a starting configuration for the remaining hyperparameters, then varied each hyperparameter to one of its extreme values to see the effect of that hyperparameter individually. This allowed us to slightly narrow down the hyperparameter search space.

For subsequent runs of the model (e.g. for different numbers of states), we varied the seed and for each seed, we randomly selected values from the narrowed down search space.

Chapter 4

Results

Using knowledge distillation, we were able to significantly improve the model from the baseline to more accurately predict mortality. We adapted the existing AR-HMM-ADVI models to work on the MIMIC-IV dataset and found that the similarity constraint helped the model achieve high predictive power for various numbers of hidden states. However, the model exhibited stability issues when incorporating a discriminator constraint. We found that the alternative method we developed for incorporating the similarity constraint, which uses the EM algorithm, was similarly successful to the ADVI method in learning states that could be used to successfully predict the final mortality outcome. We also evaluated the model's performance on other outcomes, including presence of pulmonary edema, and need for diuretics, dialysis, or mechanical ventilation. This chapter also conducts exploratory analyses on the interpretability of the states learned by our models at both the population level and the individual level. We find that the model learned some states that were significantly associated with the hospital mortality, which suggests that the states are potentially clinically meaningful.

4.1 AR-HMM-ADVI

We adapted the existing AR-HMM-ADVI models from [25] to apply them to the MIMIC-IV dataset, and we ran the models with different numbers of states to see

Table 4.1: DISC-AR-HMM-ADVI Test AUC Results

Model	2 states	3 states	5 states	10 states
LSTM (Teacher)	0.833			
AR-HMM-ADVI (Baseline)	0.500	0.579	0.516	0.653
DISC-AR-HMM-ADVI	0.799	0.693	0.596	0.488
KD-AR-HMM-ADVI	0.783	0.795	0.791	0.792
KD-DISC-AR-HMM-ADVI	0.783	0.788	0.784	0.778

Table 4.2: For the AR-HMM-ADVI models, the table shows the average test AUROC of the top 3 runs of each model (as determined by validation AUROC) for that number of states.

if varying the number of states would affect performance. Table 4.1 summarizes the results from the various models. The baseline AR-HMM-ADVI had AUROCs ranging from 0.50 to 0.65. Both models that incorporated knowledge distillation performed much better that the baseline, regardless of the number of states, and achieved average AUROCs between 0.78 and 0.79. However, as can be seen in the DISC-AR-HMM-ADVI results, the ADVI algorithm exhibited stability issues when it came to the discriminator model - although the DISC-AR-HMM-ADVI model had the highest average test AUROC when using 2 states, the AUROC was much lower for higher numbers of states.

To examine this stability issue, we looked at the evidence lower bound (ELBO) metric, which is a lower bound on the log-likelihood and was used to evaluate the generative ability of the model. The average ELBO from the top 3 seeds of each experiment are shown in table 4.3. This table confirms that the DISC-AR-HMM-ADVI model seems to have unstable metrics, with the ELBOs being unreasonably high compared to the other models. Because of these results, we decided to try a different approach to incorporating similarity loss in our models; namely, using a custom AR-HMM model that uses a modified EM algorithm for fitting.

Table 4.3: AR-HMM-ADVI Test ELBO Results ($\times 10^6$)

Model	2 states	3 states	5 states	10 states
AR-HMM-ADVI (Baseline)	-10.8e7	-4.2e7	-9.0e7	-6.0e7
DISC-AR-HMM-ADVI	9.45e3	1.8e7	4.7e7	16.5e7
KD-AR-HMM-ADVI	-5.70	-9.52	0.0316	-12.76
KD-DISC-AR-HMM-ADVI	-5.92	-9.30	-15.63	-14.52

Table 4.4: This table shows the average test ELBO of the top 3 models (as determined by validation AUROC) for each category.

4.2 AR-HMM-EM

For our AR-HMM-EM experiments, we looked at results for models with 5 states and 10 states. For each number of states, as a baseline, we first ran the EM model for 50 iterations of E- and M- steps. We tried several seeds, and the results for each of these seeds can be seen in Tables B.1 and B.2. After trying several seeds and selecting the baseline model with the highest training and validation log-likelihoods, we continued training the model with an adjusted algorithm - between the E- and M- steps, we trained a recognition network to also incorporate the similarity loss. We used this for up to 20 more iterations of the E- and M- steps. Due to the early stopping mechanism we had that was based on the average validation AUROC of the recognition network, most experiments only ran the model with the recognition network for 11 iterations before stopping. We tried several different seeds and hyperparameter settings for the recognition network, and chose the best models by looking at the final validation AUROC and validation log-likelihood.

For this section, most of the figures and tables provided are for one specific KD-AR-HMM-EM model, which we chose to be the the 10-state KD-AR-HMM-EM model with the highest validation AUROC. Select results for the 10-state KD-AR-HMM-EM model with the highest validation log-likelihood can be found in Appendix B.3.

Table 4.5: Effect of Similarity Coefficient on AUROC and Log-Likelihood (5 states)

	AUC_{val}	LL_{val}	AUC_{test}	LL_{test}	LL_{train}
LSTM (Teacher)	0.780	N/A	0.833	N/A	N/A
AR-HMM-EM Base	0.643	2835237	0.629	2830714	14631833
sim coef = 1e1	0.640	136588	0.581	116903	718746
sim coef = 1e2	0.703	129589	0.748	106958	677687
sim coef = 1e3	0.740	105622	0.803	79266	553452
sim coef = 1e30	0.727	83259	0.768	46291	407369

4.2.1 Mortality Prediction

KD-AR-HMM-EM (5 states)

Much of our hyperparameter tuning was done using 5 states. One of the main hyperparameters we wanted to set was the similarity coefficient. To identify the optimal value of this coefficient, we fixed the seed and other hyperparameters, and trained the model using different values for the similarity coefficient. Table 4.5 shows the results from these experiments. An increase in the similarity coefficient corresponds to an increase in the AUROCs, but only up to a certain point. A coefficient of 1 was not sufficient for the knowledge from the teacher model to make an impact, so the AUROC did not improve compared to the baseline. On the other hand, when the coefficient was too large (1e30), this resulted in a lower log-likelihood and did not further improve the AUROC when compared to using a coefficient like 1e3.

Furthermore, we found that an increase in the similarity coefficient corresponds to a decrease in the log-likelihood. Since our goal is to improve the AUROC without affecting the log-likelihood too much, we settled on 1e3 as the ideal similarity coefficient for subsequent experiments.

After narrowing down the hyperparameter search space, we trained various baseline models using different random seeds. The best baseline model for 5 states, as determined by looking at the training log-likelihood, was using a seed of 131 and gave a test AUROC of 0.642 and a final test log-likelihood of -3454621. The test log-likelihood of this baseline model was much lower than other models, but this may

Table 4.6: KD-AR-HMM-EM Best Results (5 states)

Model	AUC_{val}	LL_{val}	AUC_{test}	LL_{test}	LL_{train}
LSTM (Teacher)	0.780	N/A	0.833	N/A	N/A
AR-HMM-EM Base	0.647	2868846	0.642	-3454621	14804169
KD-AR-HMM-EM (AUC)	0.762	131176	0.810	108720	682222
KD-AR-HMM-EM (LL)	0.760	170763	0.795	149569	888422

simply be due to randomness, as the training and validation log-likelihoods for this model were higher than other baseline models.

After incorporating knowledge distillation on top of this chosen baseline model, the final model with the highest validation AUROC had a test AUROC of 0.810 and a test log-likelihood of 108720. The model with the highest validation log-likelihood had a test AUROC of 0.795 and a test log-likelihood of 149569. Metrics for the baseline model and the two best models are shown in Table 4.6. KD-AR-HMM-EM (AUROC) is the model with the highest validation AUROC, and KD-AR-HMM-EM (LL) is the model with the highest validation log-likelihood.

KD-AR-HMM-EM (10 states)

The best baseline model for the AR-HMM-EM with 10 states used a seed of 125 and gave a test AUROC of 0.639, and a final test log-likelihood of 3438851. After incorporating the recognition network and trying several seeds, we chose the best models by looking at the validation AUROC and validation log-likelihood. The model with the highest validation AUROC had a test AUROC of 0.811 and a test log-likelihood of 243512. The model with the highest validation log-likelihood had a test AUROC of 0.802 and a test log-likelihood of 317686. Metrics for the baseline model and the best final models are shown in Table 4.7.

The following figures are derived from the 10-state KD-AR-HMM-EM model with the best validation AUROC. Figure 4-1 shows the recognition network losses for the recognition network with the highest validation AUC, on the last iteration of the EM algorithm. The total loss, MSE loss, and similarity loss all go down throughout

Table 4.7: KD-AR-HMM-EM Best Results (10 states)

Model	AUC_{val}	LL_{val}	AUC_{test}	LL_{test}	LL_{train}
LSTM (Teacher)	0.780	N/A	0.833	N/A	N/A
AR-HMM-EM Base	0.675	3441403	0.639	3438851	17805116
KD-AR-HMM-EM (AUC)	0.761	265154	0.811	243512	1384804
KD-AR-HMM-EM (LL)	0.747	337504	0.802	317686	1760088

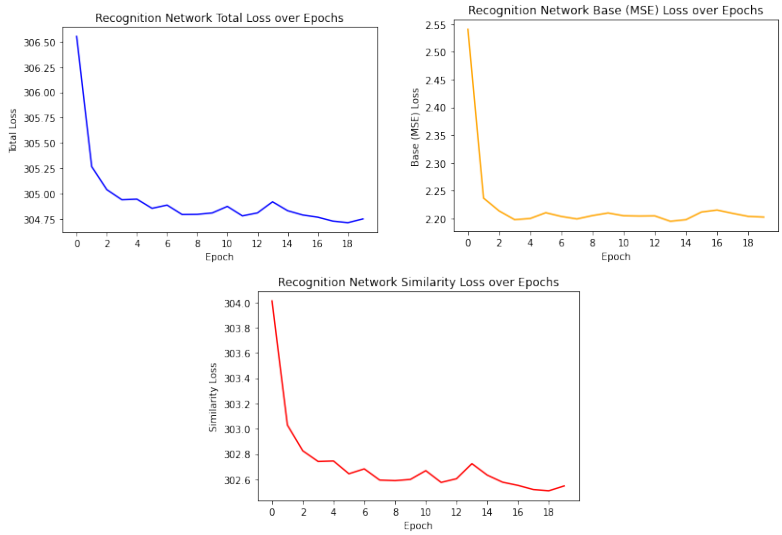
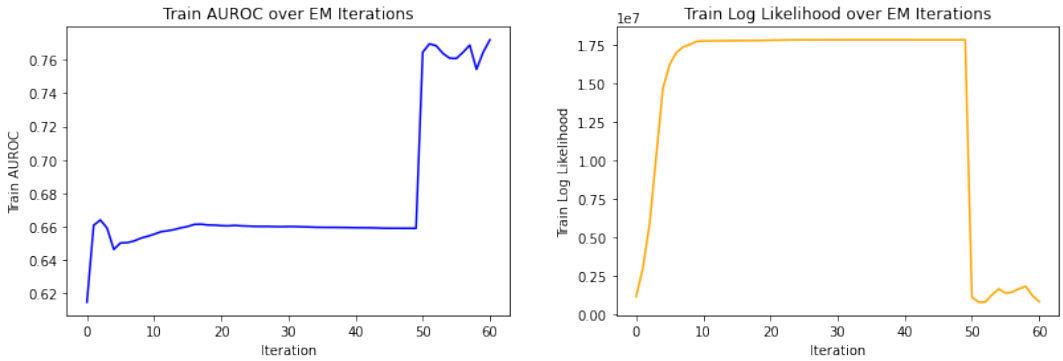


Figure 4-1: KD-AR-HMM-EM (10 states, model with best validation AUC) - Losses over epochs for the best recognition network trained in the last iteration of the EM algorithm

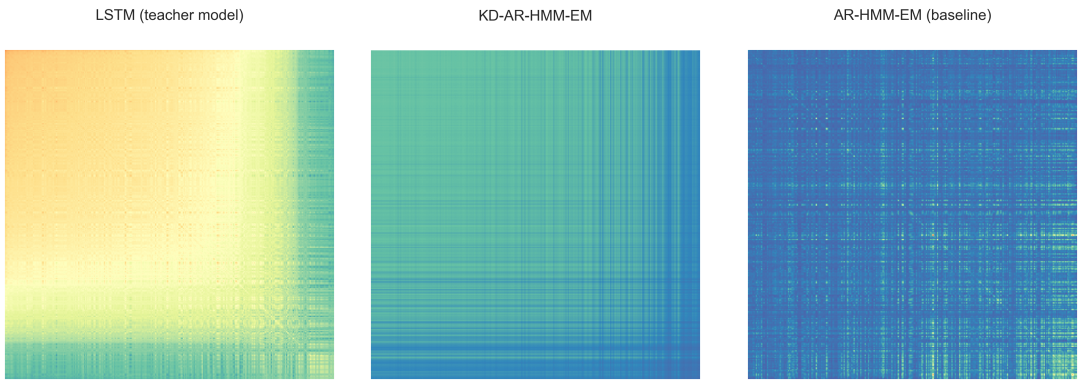
the epochs of training, which suggests that the similarity coefficient of 1e3 was large enough to be considered in the recognition network training and affect the recognition network, but small enough that it did not prevent the base MSE loss from decreasing.

Figure 4-2 shows the train AUROC for mortality prediction and the training log-likelihood over EM iterations. The train AUROC improves slightly in the first few normal EM iterations but then stays roughly the same until iteration 50 (when the recognition network is first used). After just one iteration of using the recognition network that incorporates knowledge distillation, the train AUROC is greatly improved. On the other hand, the train log-likelihood increases significantly for the first few normal EM iterations and then continues slowly increasing, but drops sharply once the recognition network is used.



(a) Train AUROC (b) Train LL

Figure 4-2: KD-AR-HMM-EM (10 states, model with best validation AUC) - Train AUROC and LL over Iterations



(a) (b) (c)

Figure 4-3: KD-AR-HMM-EM (10 states, model with best validation AUC) - Pairwise similarity matrices for the test dataset. Each row and column corresponds to a patient in the test dataset. Brighter colors indicate higher similarity between patients. The model with the knowledge distillation constraint (b) is more similar to the teacher model (a) than the baseline before adding the knowledge distillation constraint is (c).

Figure 4-3 shows the pairwise similarity matrices for the test set for the teacher LSTM model, the KD-AR-HMM-EM that is constrained to be similar to the teacher model, and the baseline AR-HMM-EM. The constrained model is more similar to the teacher model than the baseline is, indicating success with the knowledge distillation technique.

Table 4.8: KD-AR-HMM-EM (10 states, model with best validation AUC) Results for Other Outcome Predictions. # pos /# total shows the number of patients with that final outcome, and the total number of patients remaining in the dataset for that experiment, respectively. The other numbers in the cell are the test AUROCs for the trained models.

Model	Mortality	Edema	Dialysis	Mech. Vent.	Diuretic
# pos/# total	981/7296	911/5034	163/7462	221/4166	970/6413
AR-HMM-EM	0.634	0.643	0.663	0.529	0.671
KD-AR-HMM-EM	0.811	0.619	0.830	0.636	0.587

4.2.2 Other Outcome Predictions

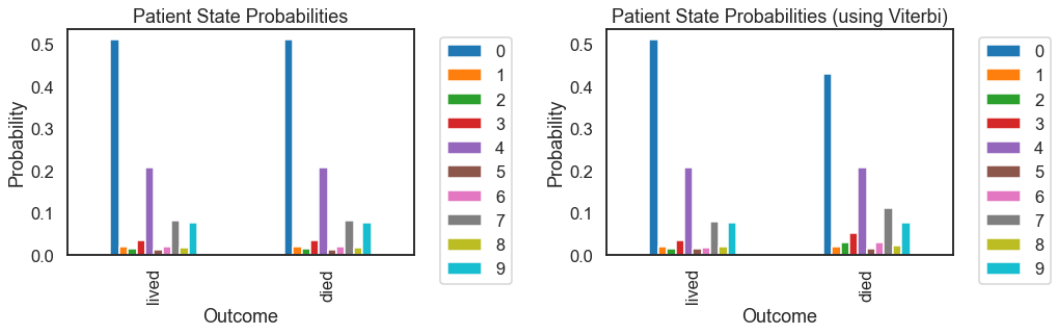
Although the bulk of our work focused on mortality prediction, we were also interested in our model's ability to accurately predict other outcomes. We used the same set of features (state marginals) to train logistic regression models for a variety of outcomes. For each outcome, we took the existing dataset and removed any patients who had that particular outcome within the first 24 hours. The final datasets had the following rates of positive outcomes (i.e. patients who ultimately experienced that outcome): 13.4% for mortality, 18.1% for edema, 2.2% for dialysis, 5.3% for mechanical ventilation, and 15.1% for diuretics. Specific counts for each dataset as well as results from the individual regressions trained to predict each outcome are summarized in Table 4.8. We were unable to predict pulmonary edema, mechanical ventilation, or diuretics very accurately, but we achieved a test AUROC of 0.830 when predicting dialysis, which was much better than the baseline.

4.3 Exploratory Analyses

4.3.1 Population Statistics

State Probabilities

To interpret the states, we divided the test set into patients who lived and patients who died, and looked at the population-level state probabilities for each cohort. Two



(a) AR-HMM-EM (10 states) Baseline State Probabilities, calculated using expected states

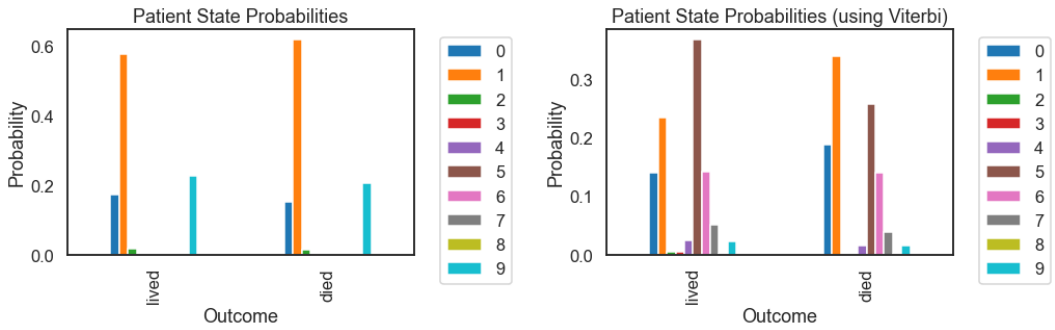
(b) AR-HMM-EM (10 states) Baseline State Probabilities, calculated using Viterbi sequence

Figure 4-4: AR-HMM-EM (10 states) Baseline State Probabilities

methods were used for calculating these state probabilities - one method used the state marginals outputted by the KD-AR-HMM-EM, while the second method used the Viterbi algorithm to decide what state a patient would be in at each timestep. The Viterbi algorithm is a method to get the most likely sequence of hidden states that would result in the sequence of observations,.

Figure 4-4 shows the state probabilities as outputted by the baseline AR-HMM-EM after the 50 normal EM iterations, but before incorporating the similarity loss. Regardless of which method is used to calculate the state probabilities, there does not appear to be a clear difference between the distributions for the two groups, as the many of the patients in both groups fall under one state (state 0).

Figure 4-5 shows the state probabilities as outputted by the KD-AR-HMM-EM after incorporating knowledge distillation. Since it is difficult to see the exact probabilities through the graph, a table with the same probabilities is provided in Table 4.9, along with exact counts for each state predicted by the Viterbi algorithm. When using the state marginals to calculate probabilities, there does not appear to be a clear difference between the distributions for the two groups. However, using the Viterbi algorithm shows some differences between the cohorts - for example, the most prevalent state for patients who lived was state 5, but the most prevalent state for patients who died was state 1. This is an improvement from the baseline because we can identify some differences in state probabilities between the cohorts.



(a) KD-AR-HMM-EM (10 states) State Probabilities, calculated using expected states

(b) KD-AR-HMM-EM (10 states) State Probabilities, calculated using Viterbi sequence

Figure 4-5: KD-AR-HMM-EM (10 states, model with best validation AUC) State Probabilities

Odds Ratios and p-values

To determine whether or not a state is significantly associated with mortality, we looked at the odds ratios and p-values for each state. We fitted a logistic regression on the test set, where the features were the features generated by the KD-AR-HMM-EM and the labels were the mortality outcomes.

We used univariate logistic regressions to try and isolate the effect of each individual state on the outcome. We used two methods of creating features for input to the regression - one method used the state marginals, while the other used the Viterbi algorithm to calculate the patient's probability of being in a state. The state marginal features were multiplied by 100000, and the Viterbi features were multiplied 100. We did not fit a logistic regression for state 2 as doing so resulted in an ill-fitted model. We adjusted the p-values using an FDR (False Discovery Rate) adjustment to account for the fact that we fitted several distinct univariate logistic regressions.

Using the state marginal features, this univariate analysis suggested that all states were significant, with p-values all < 0.001. Since this seems unreasonable, we do not include a table of these results here. The univariate analysis using Viterbi features showed that states 0 and 1 were significantly positively associated with mortality, while states 3, 5, 7, and 9 were significantly negatively associated with mortality, and states 4, 6, and 8 had no significant association in either direction. A summary of

Table 4.9: State Probabilities for KD-AR-HMM-EM (10 states, model with best validation AUC). These are the same values used to make the graph in Figure 4-5. For the Viterbi states, the actual count of how many times those states were predicted across the test set patients is shown in parentheses.

	Probability using State Marginals		Probability using Viterbi States	
State	Lived cohort	Died cohort	Lived cohort	Died cohort
0	0.1733	0.1548	0.1400 (2944)	0.1886 (670)
1	0.5798	0.6204	0.2355 (4951)	0.3387 (1203)
2	0.0185	0.0154	0.0065 (136)	0.0003 (1)
3	0.0001	0.0001	0.0060 (126)	0.0023 (8)
4	0.0003	0.0003	0.0245 (515)	0.0163 (58)
5	0.0001	0.0001	0.3676 (7729)	0.2579 (916)
6	0.0001	0.0001	0.1421 (2987)	0.1399 (497)
7	0.0002	0.0001	0.0521 (1095)	0.0386 (137)
8	0.0006	0.0005	0.0021 (45)	0.0017 (6)
9	0.2270	0.2083	0.0236 (496)	0.0158 (56)

the results can be found in Table 4.10.

AR Coefficients and Covariances

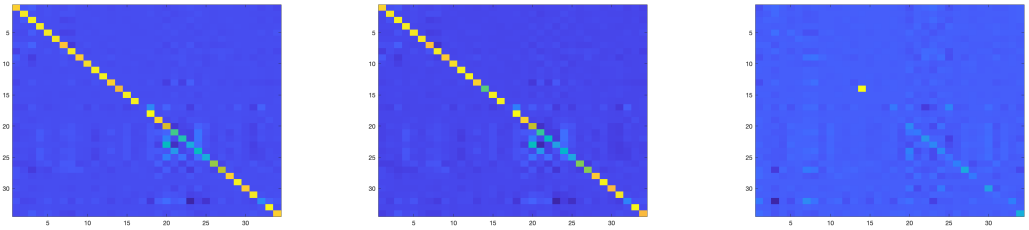
Since states 1 and 5 seemed significant in opposite directions, and since we saw differences in the distributions between cohorts, we compared these two states by looking at heatmaps for AR coefficients and covariances, as shown in Figures 4-6 and 4-7. Although most parts of the heatmaps look similar, there are a few cells in the coefficients heatmap that are brighter or differently colored than others.

Simulation of States

We are interested in examining the dynamics between covariates, and how those dynamics differ among states. We can use our trained KD-AR-HMM-EM model to simulate the covariates of a patient in a particular state over time. For our simulations, we simulated the covariates for 34 timesteps. We did not plot the first 10 timesteps, because the covariates were likely shifting around before reaching dynamics that were more representative of that state.

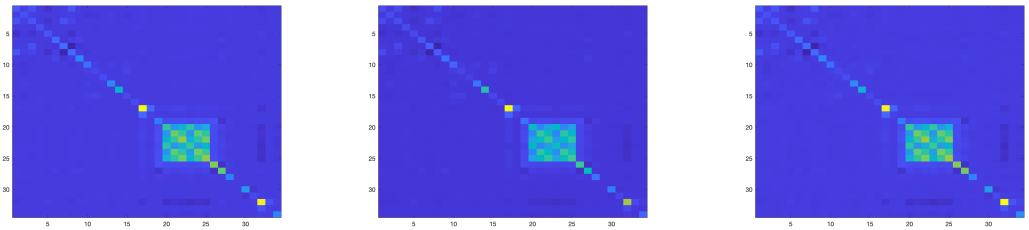
Table 4.10: KD-AR-HMM-EM (10 states, model with best validation AUC), Odds Ratios and P-values for Univariate Logistic Regressions fitted using Viterbi features. Significant p-values are bolded.

State	Odds Ratio (95% CI)	FDR-adj p-value
0	(1.042, 1.082)	$<$ **0.001**
1	(1.028, 1.051)	$<$ **0.001**
3	(0.673, 0.945)	**0.016**
4	(0.913, 1.001)	0.068
5	(0.967, 0.984)	$<$ **0.001**
6	(0.984, 1.013)	0.842
7	(0.910, 0.982)	**0.009**
8	(0.773, 1.158)	0.665
9	(0.881, 0.988)	**0.027**



(a) State 1 AR Coefficients (b) State 5 AR Coefficients (c) Difference between State 1 AR Coefficients and State 5 AR Coefficients

Figure 4-6: KD-AR-HMM-EM (10 states, model with best validation AUC), AR Coefficients for State 1 and State 5



(a) State 1 AR Covariance (b) State 5 AR Covariance (c) Difference between State 1 AR Covariance and State 5 AR Covariance

Figure 4-7: KD-AR-HMM-EM (10 states, model with best validation AUC), AR Covariance for State 1 and State 5

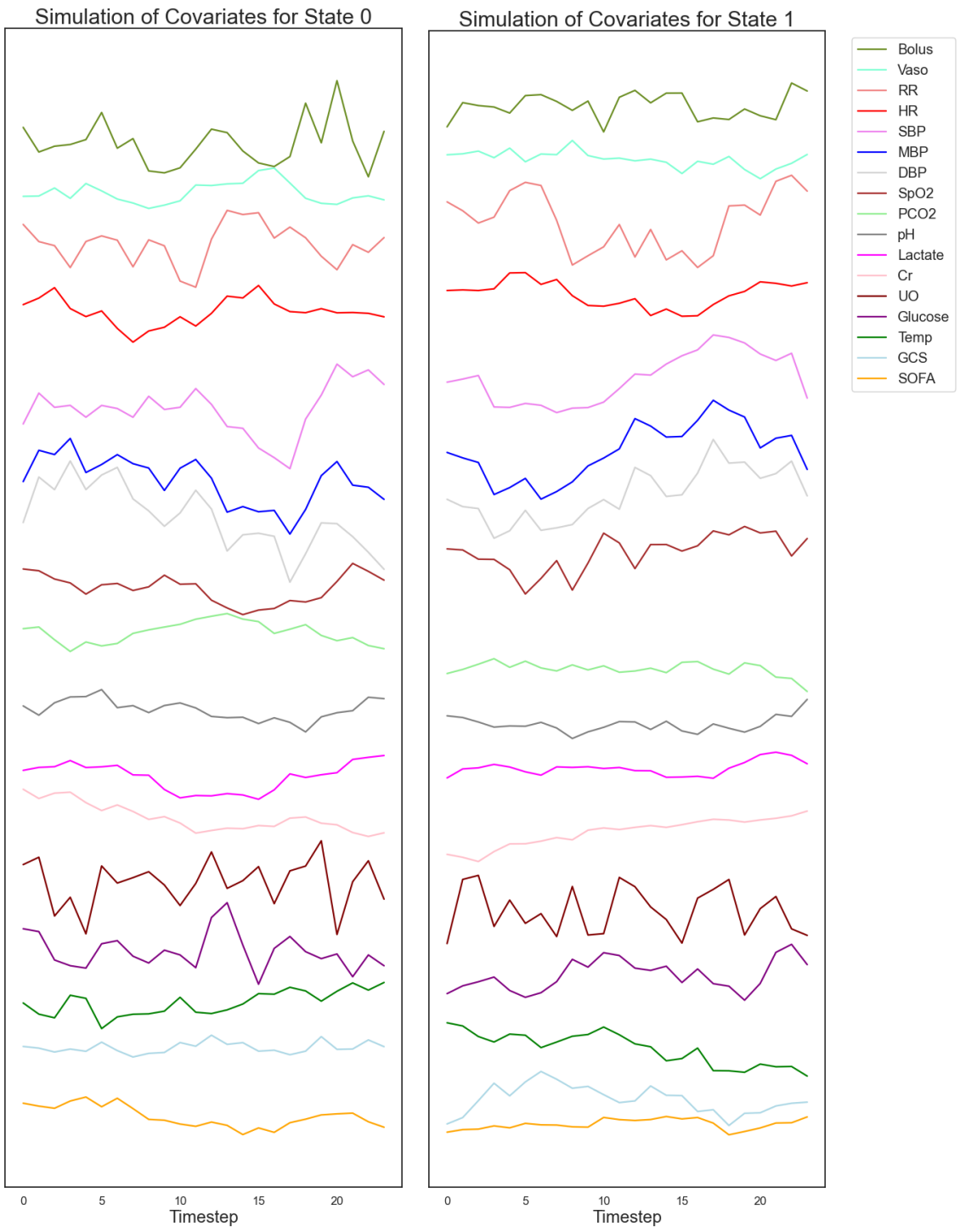
Figures 4-8 and 4-9 show the simulations of certain covariates, including vitals and treatments, for 4 selected states (2 high-risk, 1 low-risk, and 1 that was not significant in either direction). We notice that in general, the high-risk states seem to have much higher variance in these covariates than the low-risk and neutral states. This makes sense because we might expect a low-risk or neutral-risk patient to be more stable, so their vitals and other measurements may not fluctuate as much. States 0 and 1, which are both high-risk states, seem to exhibit the dynamic where if the blood pressure drops, the respiratory rate increases, and vice versa, whereas the other states do not exhibit this trend. An interesting dynamic in State 0 is that when the blood pressure drops, the bolus fluids increase shortly afterwards, and then the blood pressure increases again. This likely indicates that doctors observed the drop in blood pressure and chose to administer more bolus fluids as treatment, and this was successful, at least initially. We notice that all states exhibit the same inverse relationship between pH and PCO2. The high-risk states seem to have more variance in the glucose and creatinine levels.

We also evaluated the significance of the learned states in predicting pulmonary edema using univariate logistic regressions. As previously seen in Table 4.8, we were unable to predict pulmonary edema very successfully. Table 4.11 supports these findings, as only state 0 appeared significant in our regressions. State 0 was positively associated with the development of pulmonary edema, which matches the observation that state 0 was also positively associated with mortality.

A similar simulation for the baseline 10-state AR-HMM-EM can be found in Appendix B as a comparison for the interpretability of the baseline AR-HMM-EM and constrained KD-AR-HMM-EM models.

4.3.2 Individual Patient Analysis

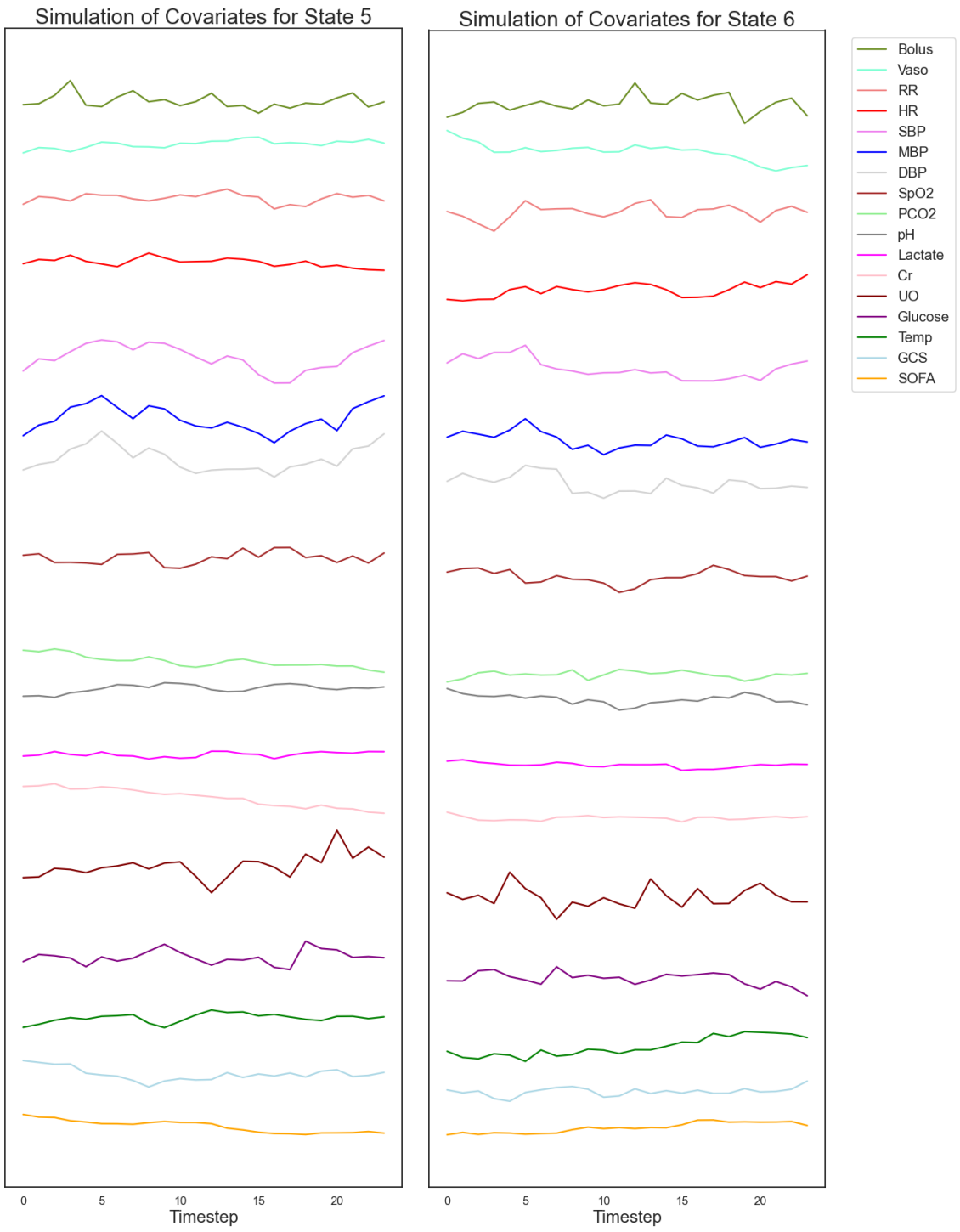
We compared the trajectories and state assignment of individual patients who lived vs. those who died in the hospital. One example of such a comparison is shown in Figure 4-10. This figure shades different states in different colors. The states with higher risk (i.e. states positively associated with mortality, which are states 0 and



(a) State 0 (high-risk)

(b) State 1 (high-risk)

Figure 4-8: KD-AR-HMM-EM (10 states, model with best validation AUC), simulation of select covariates for States 0 and 1 (high-risk). RR represents respiratory rate, Cr represents creatinine, and UO represents urine output.



(a) State 5 (low-risk)

(b) State 6 (neutral)

Figure 4-9: KD-AR-HMM-EM (10 states, model with best validation AUC), simulation of select covariates for States 5 (low-risk) and 6 (neutral). RR represents respiratory rate, Cr represents creatinine, and UO represents urine output.

Table 4.11: KD-AR-HMM-EM (10 states, model with best validation AUC), Pulmonary Edema State Probabilities, Odds Ratios, and P-values for Univariate Logistic Regressions, using Viterbi features. The values under the Pos. Cohort and Neg. Cohort columns indicate what percentage of the states predicted by the Viterbi algorithm corresponded to that state, for the group with pulmonary edema (positive) and without pulmonary edema (negative). The significant p-value is bolded.

State	Pos. Cohort	Neg. Cohort	Odds Ratio (95% CI)	FDR-adj p-value
0	0.1347	0.1674	(1.271, 2.013)	**0.001**
1	0.2380	0.2496	(0.925, 1.194)	0.640
2	0.0072	0.0042	(0.306, 1.521)	0.583
3	0.0062	0.0035	(0.053, 1.247)	0.305
4	0.0271	0.0221	(0.525, 1.240)	0.583
5	0.3683	0.3589	(0.895, 1.074)	0.838
6	0.1410	0.1254	(0.754, 1.063)	0.513
7	0.0511	0.0497	(0.653, 1.388)	0.867
8	0.0023	0.0025	(0.172, 8.060)	0.867
9	0.0243	0.0168	(0.274, 0.984)	0.223

1), are colored in shades of red and orange, while the states with lower risk (states 3, 5, 7, and 9) are colored in shades of green. We also plot a few covariates along with these states. In this example, the patient who lived initially spent some time in the high-risk states, but later spent most of their time in the low risk (green) states throughout the 24 hours, while the patient who did not survive the hospital stay initially spent time in low-risk states, but later on spent most of their time in the high risk (red and orange) states. This matches the odds ratio analysis, which also suggested that states 0 and 1 were more common among patients who died, while states 3, 5, 7 and 9 were more common among patients who lived. Neither patient was administered fluid boluses during these 24 hours, so we did not plot that value. The patient who lived saw a decrease in the amount of vasopressors given to them later on in their stay, whereas the patient who died saw an increase in the amount of vasopressors administered as well as in their SOFA score. Additionally, the patient who lived had a relatively stable heart rate and PCO2 value, both of which seemd to decrease just before the patient entered low-risk states, while the patient who died saw a big spike in PCO2 and heart rate later on in their stay while they were in

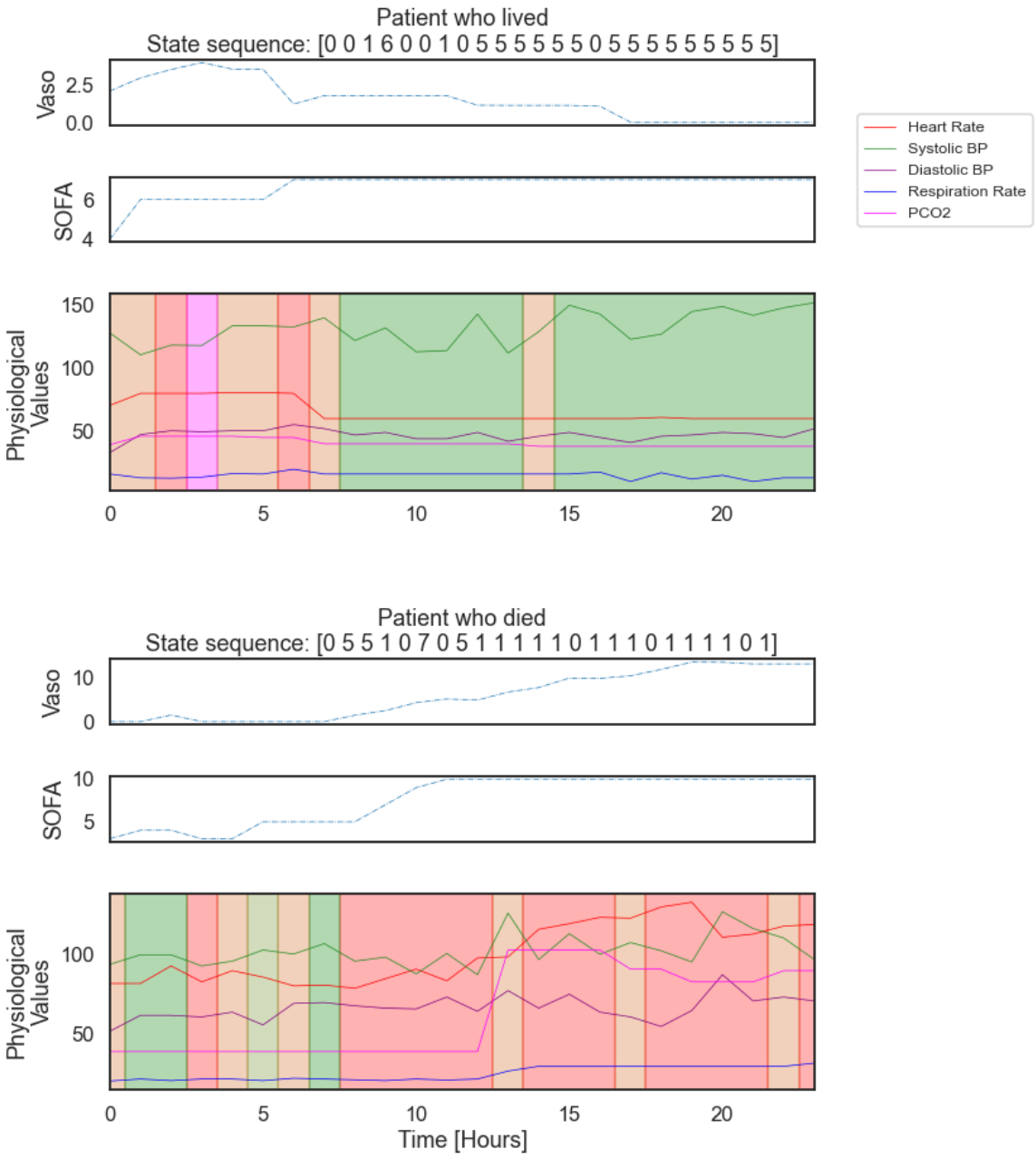


Figure 4-10: Patient Comparison. States are predicted using the Viterbi algorithm on the KD-AR-HMM-EM (10 states) model. "Vaso" indicates the amount of vasopressors administered to the patient.

high-risk states.

Chapter 5

Conclusion

Predicting outcomes for sepsis patients is a challenging problem faced by physicians. This is due to the dynamic and time-varying nature of the condition, as well as the many differences between individuals that may cause one patient to have a better outcome than another. There are a wide variety of factors to consider when trying to predict a patient's outcome, such as the recent history of physiological traits of the patient, how they have responded to any previous treatments, and how similar patients have fared. Due to the large amount of information to consider in the prediction, machine learning presents itself as a promising tool for this setting. However, given the critical nature of a healthcare setting, it is important that any model or algorithm used is interpretable.

In this thesis, we developed a method to learn an interpretable model that can still predict downstream outcomes with high accuracy. We first evaluated an existing approach presented in Saeedi et al. on a new dataset, which used ADVI to learn an AR-HMM, along with a recognition network to incorporate similarity-based constraints from a powerful teacher model LSTM [25]. We then developed an alternative approach - similar to the ADVI model, our model used an LSTM as the teacher model to provide high predictive power, and used an AR-HMM to learn more interpretable hidden states. However, in our approach, we used the EM algorithm instead of ADVI to learn the latent states. We also used a recognition network to incorporate a similarity-based constraint. After developing this model, we evaluated its predic-

tive and generative performance, as well as its interpretability. By incorporating the similarity constraint, we were able to achieve higher predictive power compared to the baseline. Furthermore, differences in the state distributions between patients who died and patients who lived suggest that the learned hidden states are interpretable and can potentially provide meaningful information about patients.

5.1 Discussion & Future Work

The results we provided in this thesis suggest many opportunities for further exploration, some of which are outlined below.

5.1.1 Limitations of Current Analysis

One limitation of our interpretability analysis was our use of univariate analyses to examine significance of states with respect to the mortality outcome. The odds ratios we presented are only indicative of correlation, and there could be confounding variables involved that cause a state to appear significant when it actually is not. In future work, we should adjust for possible confounders in order to better understand how states are associated with outcomes. Another limitation was that we used p-values to determine whether a state was significantly associated with the outcome, but there are known limitations in how valid p-values are at assessing statistical significance [7]. Finally, for our individual patient analysis, we only examined two patients in different categories, but the trends we observed in that analysis may not hold for all patients.

5.1.2 Further Interpretability Analysis

Although we found some differences in state distributions when using the Viterbi algorithm to calculate the most likely states of a patient, there was still a lot of overlap of states between patients who lived and patients who died. While the high predictive power of the model shows that the model was able to learn these differences, more

analysis needs to be done to see if we can identify specific covariates that distinguish between states. Such information would be valuable for physicians, who might learn new trends between certain covariates and sepsis outcomes that could help them predict the trajectory of a patient under the current treatment strategy.

5.1.3 Predicting Other Outcomes

We are interested in seeing if the approach we used in this thesis to predict patient mortality is also successful in predicting other outcomes. In this thesis, we trained the teacher model to predict one outcome (mortality), and used this teacher model to help learn the student model (KD-AR-HMM-EM). Thus, the states learned by the KD-AR-HMM-EM were optimized for mortality prediction. We used these same states to predict other outcomes, but most of these predictions were not as accurate as the predictions for mortality. In the future, we would like to train separate teacher models to predict each individual outcome of interest, and train separate student models on each corresponding teacher model, then evaluate how the student model that is trained specifically on that outcome performs when used to predict that outcome.

5.1.4 Predicting Treatment Response

Ultimately, we would like to be able help physicians decide what treatment to give a patient. Our model is able to accurately predict final outcomes given the patient data and their treatments, but we would like to extend this model to be able to predict how a patient might respond to various treatment options, and choose the option with the best outcomes. This would require causal inference to predict outcomes for patients under counterfactual strategies.

Thus, one avenue to explore in future work is to combine our work with G-Net, a model introduced by Li et al. which uses an RNN for a causal inference method known as g-computation in order to predict responses to both time-varying and dynamic treatments [18] [9]. We could first use our KD-AR-HMM-EM to get interpretable states for patients, and then provide those states as input to G-Net to make

predictions on treatment effects.

In this thesis, we were able to learn states that can be used to have both high predictive power and greater interpretability. We hope that the work presented in this thesis will be able to improve treatment outcomes for sepsis patients.

Appendix A

Data Tables

Table A.1 shows the variables that were used as inputs for both the teacher LSTM and student AR-HMM models. The vasopressor amount variable measures the total amount of vasopressors used during that time period. Vasopressors are standardized by comparing their relative strength to norepinephrine, also known as noradrenaline or norad. The vasopressors included in this standardization are norepinephrine (or levophed), epinephrine, vasopressin, phenylephrine, and dopamine. The measurement unit used is mcg/kg/minute, except for vasopressin, which is expressed as units/minute. The standardization process involves adjusting the dosage rate of each vasopressor by multiplying it with a scaling constant based on the typical dosing of each drug. Norepinephrine is typically administered at a dosage range of 0-1 mcg/kg/minute. If multiple vasopressors were used during the same time period, the combined total dose for each hour is reported.

Table A.1: MIMIC time-varying variables that were used as inputs to both the teacher LSTM and student AR-HMM models.

Variable Name	**Variable Type**	**Units**
Heart Rate	Continuous	beats/min
Diastolic Blood Pressure	Continuous	mmHg
Systolic Blood Pressure	Continuous	mmHg
Mean Blood Pressure	Continuous	mmHg
Minimum Diastolic Blood Pressure	Continuous	mmHg
Minimum Systolic Blood Pressure	Continuous	mmHg
Minimum Mean Blood Pressure	Continuous	mmHg
Temperature	Continuous	°C
SOFA Score	Treated as Continuous	N/A
Glasgow Coma Score	Treated as Continuous	N/A
Platelet	Continuous	counts/10^9L
Hemoglobin	Continuous	g/dL
Calcium	Continuous	mg/dL
BUN	Continuous	mmol/L
Creatinine	Continuous	mg/dL
Bicarbonate	Continuous	mmol/L
Lactate	Continuous	mmol/L
Potassium	Continuous	mmol/L
Bilirubin	Continuous	mg/dL
Glucose	Continuous	mg/dL
pO2	Continuous	mmHg
SO2	Continuous	%
SpO2	Continuous	%
pCO2	Continuous	mmHg
Total CO2	Continuous	mEq/L
pH	Continuous	Numerical[1,14]
Base excess	Continuous	mmol/L
Weight	Continuous	kgs
Respiratory Rate	Continuous	breaths/min
Total Fluids	Continuous	mL
Urine Output	Continuous	mL
Total Urine Output	Continuous	mL
Fluid Bolus	Continuous	mL
Vasopressor Amount	Continuous	mcg/kg/min

Table A.2: MIMIC time-varying variables that were only used for the teacher LSTM model, and not for the student AR-HMM models.

Variable Name	**Type**	**Units**
Minimum Mean Blood Pressure from Baseline	Continuous	mmHg
Glasgow Coma Score - Motor	Ordinal	N/A
Glasgow Coma Score - Verbal	Ordinal	N/A
Glasgow Coma Score - Eye	Ordinal	N/A
O2 requirement level	Ordinal [0,6]	N/A
Pulmonary Edema Indicator	Binary	N/A
Cumulative Edema	Binary	N/A
Diuretics Indicator	Binary	N/A
Diuretics Amount	Continuous	mg
Dialysis Indicator	Binary	N/A
Mechanical Ventilation Indicator	Binary	N/A
Bolus Indicator	Binary	N/A
Vasopressor Indicator	Binary	N/A

Appendix B

Additional Results

B.1 Baseline AR-HMM-EM Results for Different Seeds

This section provides results for the baseline experiments that we ran with different seeds. Figure B.1 shows the different seeds we tried for the 5-state baseline AR-HMM-EM model and their metrics, and Figure B.2 shows the different seeds we tried for the 10-state baseline AR-HMM-EM model and their metrics.

B.2 Baseline 10-state AR-HMM-EM Exploratory Analyses

This section analyzes the interpretability for the baseline 10-state AR-HMM-EM model, which came from running the normal EM steps for 50 iterations. We conducted an odds ratio analysis using univariate logistic regressions with the Viterbi features. We multiplied the features by 100, and the results are shown in Table B.3. The results suggest that state 0 was negatively associated with mortality, and states 2, 3, 6, and 7 were positively associated with mortality.

We simulated covariates for select states using the baseline AR-HMM-EM, the same way we simulated states for the KD-AR-HMM-EM in Section 4.3.1. Figure B-1 shows the simulation for state 3, which was significantly associated with mortality,

Table B.1: AR-HMM-EM (5 states) Baseline Results. This table shows results we got for the baseline AR-HMM-EM model with 5 states, using 10 different random seeds. We selected the best model based on the model with the highest train and validation log-likelihoods, which are in bold.

Seed	AUC_{val}	LL_{val}	AUC_{test}	LL_{test}	LL_{train}
123	0.651	2808492	0.634	1768968	14499823
124	0.646	2808952	0.636	-64675	14497745
125	0.651	2811893	0.632	2805773	14509213
126	0.640	2785814	0.628	2784795	14392350
127	0.648	2791970	0.646	1794463	14415593
128	0.663	2801962	0.643	1978155	14462727
129	0.664	2792485	0.650	-531065	14415111
130	0.652	2810267	0.648	2806297	14495521
131	0.647	**2868846**	0.642	-3454621	**14804169**
132	0.649	2742988	0.626	1608935	14175382

Table B.2: AR-HMM-EM (10 states) Baseline Results. This table shows results we got for the baseline AR-HMM-EM model with 10 states, using 10 different random seeds. We selected the best model based on the model with the highest train and validation log-likelihoods, which are in bold.

Seed	AUC_{val}	LL_{val}	AUC_{test}	LL_{test}	LL_{train}
123	0.675	3212834	0.633	255984	16612384
124	0.671	3357462	0.639	3139094	17395658
125	0.675	**3441403**	0.639	3438851	**17805116**
126	0.668	3193904	0.617	1097147	16510295
127	0.673	3187565	0.632	4132	16483110
128	0.673	3290026	0.616	1946073	17027327
129	0.675	3210484	0.635	3206945	16600615
130	0.673	3215810	0.625	-807511	16636914
131	0.694	3286547	0.644	3283355	16995983
132	0.671	3188060	0.631	3182254	16476596

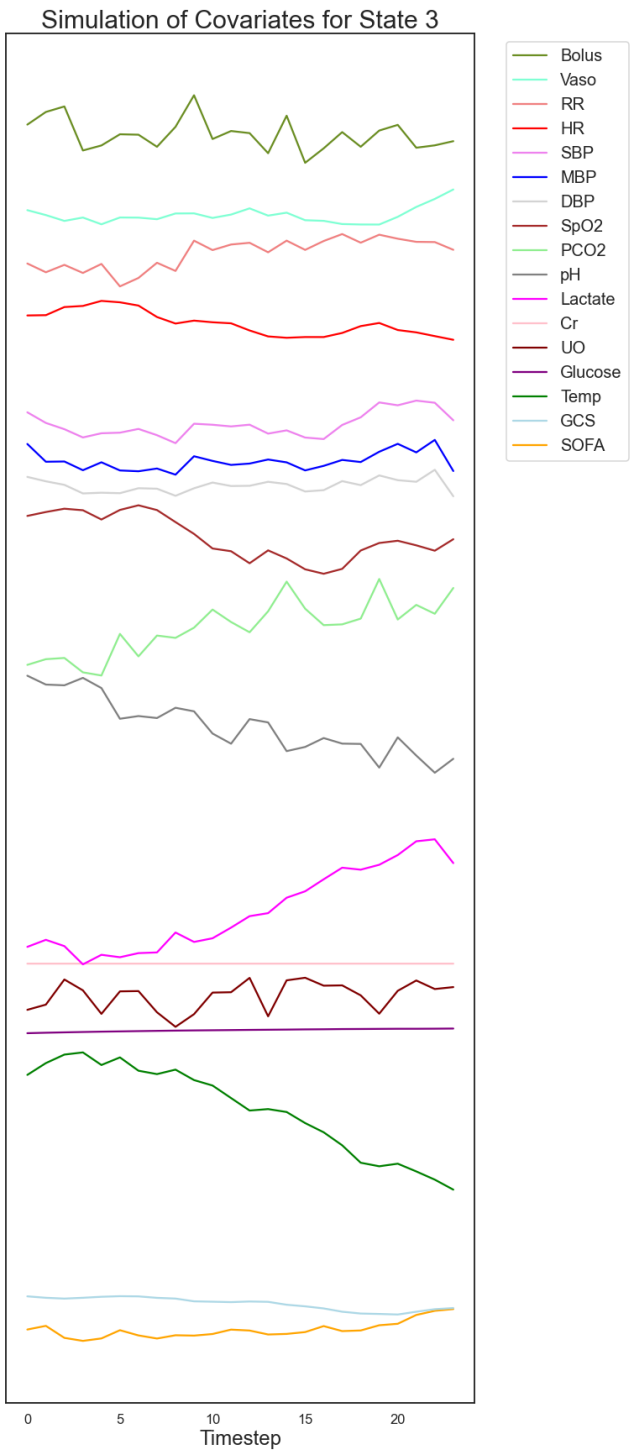
Table B.3: Baseline AR-HMM-EM (10 states), Odds Ratios and P-values for Univariate Logistic Regressions fitted using Viterbi features. Significant p-values are bolded.

State	Odds Ratio (95% CI)	FDR-adj p-value
0	(0.970, 0.988)	$<$ **0.001**
1	(0.967, 1.045)	0.928
2	(1.090, 1.215)	$<$ **0.001**
3	(1.026, 1.085)	**0.001**
4	(0.988, 1.014)	0.928
5	(0.897, 1.067)	0.896
6	(1.023, 1.105)	**0.003**
7	(1.025, 1.065)	$<$ **0.001**
8	(0.974, 1.108)	0.415
9	(0.968, 1.036)	0.928

and Figure B-2 shows the simulations for state 0, which was negatively associated with mortality, and state 4, which had no significant association in either direction. We can see that for the low-risk and neutral states, several of the covariates such as PCO2, pH, lactate, creatinine, and glucose did not seem to vary at all, whereas these values seemed to fluctuate in the high-risk state. The flat lines for many of these covariates suggests that the baseline model did not learn enough to be able to simulate these covariates well. Although the baseline model had higher log-likelihood than the KD-AR-HMM-EM, the baseline simulations do not seem as reliable, so the interpretability of the KD-AR-HMM-EM seems better.

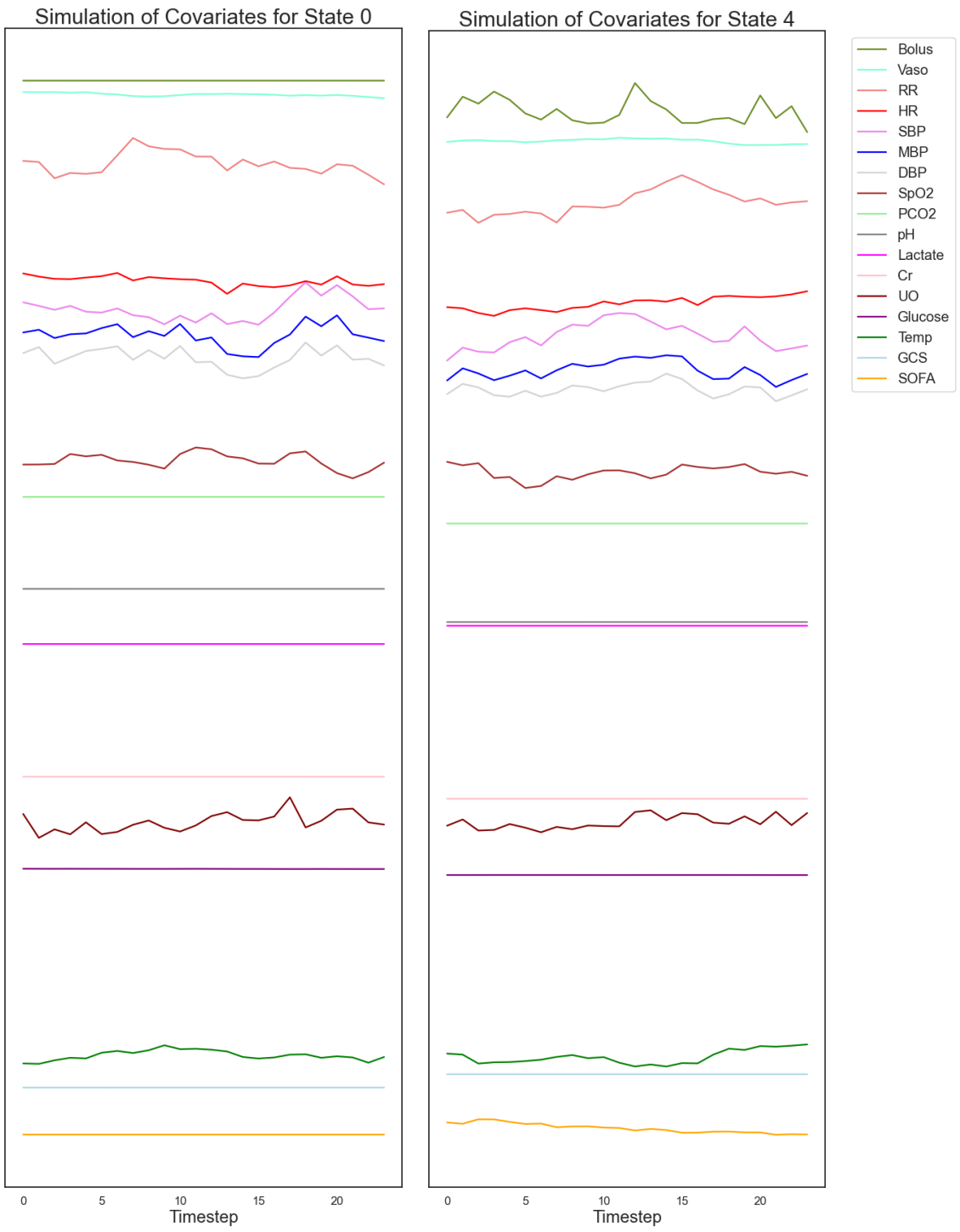
B.3 KD-AR-HMM-EM, 10 states, Model with best validation log-likelihood

This section provides more detailed results and figures for the 10-state KD-AR-HMM-EM that had the highest validation log-likelihood. This model had a test AUROC of 0.802 and a test log likelihood of 317686. The recognition network for this model was trained with a seed of 130, a learning rate of 0.1, a batch size of 64, a hidden



(a) State 3 (high-risk)

Figure B-1: Baseline AR-HMM-EM (10 states), simulation of select covariates for State 3 (high-risk). RR represents respiratory rate, Cr represents creatinine, and UO represents urine output.



(a) State 0 (low-risk)

(b) State 4 (neutral)

Figure B-2: Baseline AR-HMM-EM (10 states), simulation of select covariates for States 0 (low-risk) and 4 (neutral). RR represents respiratory rate, Cr represents creatinine, and UO represents urine output.

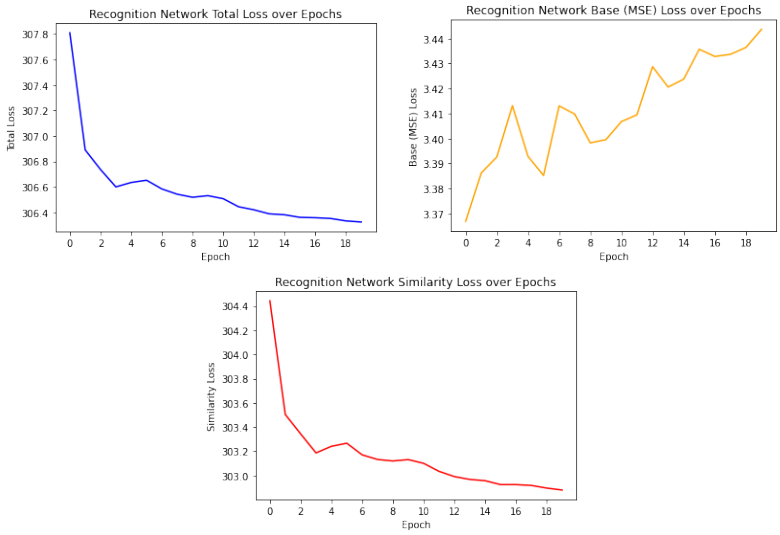


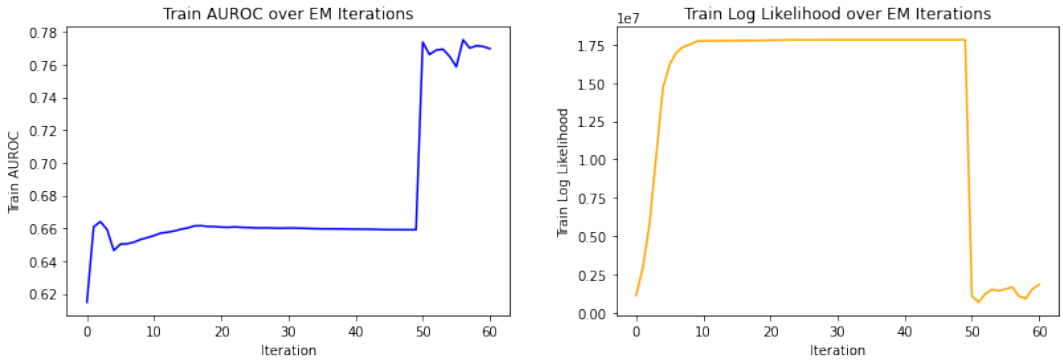
Figure B-3: KD-AR-HMM-EM (10 states, model with highest validation LL) - Losses over epochs for the best recognition network trained in the last iteration of the EM algorithm

dimension of 8, and 4 hidden layers.

Figure B-3 shows the recognition network losses for the recognition network with the highest validation AUC on the last iteration of the EM algorithm. Although the total loss and similarity loss go down throughout the epochs of training, the MSE loss between the true expected states and the predicted expected states increases. This differs from the graphs we saw with the 10-state KD-AR-HMM-EM model in Figure 4-1. In this case, it appears that the similarity coefficient of 1e3 caused the recognition network to focus too much on lowering the similarity loss, so it ignored the MSE loss.

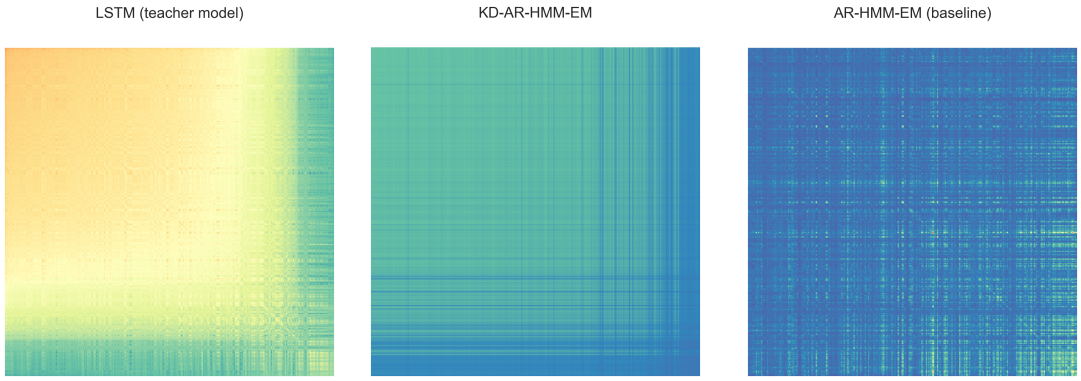
Figure B-4 shows the train AUROC for mortality prediction and the training log-likelihood over EM iterations. The graphs are similar to the results for the KD-AR-HMM-EM with the highest validation AUROC in 4-2, where the train AUROC increases significantly after incorporating knowledge distillation, but the train log-likelihood drops.

Figure B-5 shows the pairwise similarity matrices for the test set for the teacher LSTM model, the 10-state KD-AR-HMM-EM with the highest validation log-likelihood that is constrained to be similar to the teacher model, and the baseline AR-HMM-



(a) Train AUROC (b) Train LL

Figure B-4: KD-AR-HMM-EM (10 states, model with highest validation LL) - Train AUROC and LL over Iterations



(a) (b) (c)

Figure B-5: KD-AR-HMM-EM (10 states, model with highest validation LL) - Pairwise similarity matrices for the test dataset. Each row and column corresponds to a patient in the test dataset. Brighter colors indicate higher similarity between patients. The model with the knowledge distillation constraint (b) is more similar to the teacher model (a) than the baseline before adding the knowledge distillation constraint (c).

EM. Similar to the KD-AR-HMM-EM model with the best validation AUROC, we see that the constrained model is more similar to the teacher model than the baseline, indicating success with the knowledge distillation technique.

B.3.1 Other Outcome Predictions

Table B.4 shows how this model performed at predicting other outcomes. Similar to the model with the best validation AUROC, we were able to predict dialysis with a

Table B.4: KD-AR-HMM-EM (10 states, model with highest validation LL) - Results for Other Outcome Predictions. # pos /# total shows the number of patients with that final outcome, and the total number of patients remaining in the dataset for that experiment, respectively. The other numbers in the cell are the test AUROCs for the trained models.

Model	Mortality	Edema	Dialysis	Mech. Vent.	Diuretic
# pos/# total	981/7296	911/5034	163/7462	221/4166	970/6413
AR-HMM-EM	0.634	0.643	0.663	0.529	0.671
KD-AR-HMM-EM	0.811	0.583	0.805	0.607	0.580

high AUROC of 0.805, but predictions for other outcomes were less successful.

Bibliography

[1] I. Bica, A. M. Alaa, J. Jordon, and M. van der Schaar. Estimating counterfactual treatment outcomes over time through adversarially balanced representations. 2020.

[2] C. Bishop. *Pattern Recognition and Machine Learning*, chapter Mixture Models and EM. Springer, 2006.

[3] C. Cheng, C. Kung, F. Chen, I. Chiu, C. Lin, C. Chu, C. Kung, and C. Su. Machine learning models for predicting in-hospital mortality in patient with sepsis: Analysis of vital sign dynamics. *Front Med (Lausanne)*, 9(964667), 2022.

[4] L. Evans, A. Rhodes, W. Alhazzani, M. Antonelli, C. M. Coopersmith, C. French, F. R. Machado, L. Mcintyre, M. Ostermann, H. C. Prescott, C. Schorr, S. Simpson, W. Wiersinga, A. Joost, A. Fayez, C. Derek, Y. Arabi, L. Azevedo, R. Beale, ..., and M. Levy. Surviving sepsis campaign: International guidelines for management of sepsis and septic shock 2021. *Critical Care Medicine*, 49(11):e1063–e1143, November 2021.

[5] E. Fox, M. Hughes, E. Sudderth, and M. Jordan. Joint modeling of multiple time series via the beta process with application to motion capture segmentation. *The Annals of Applied Statistics*, 8(3), 2014.

[6] E. Fox, E. Sudderth, M. Jordan, and A. Willsky. Bayesian nonparametric methods for learning markov switching processes. *IEEE Signal Processing Magazine Special Issue*, 2010.

[7] A. Gelman. The problems with p-values are not just with p-values., 2016.

[8] K. Henry, D. Hager, P. Pronovost, and S. Saria. A targeted real-time early warning score (trewscore) for septic shock. *Science Translational Medicine*, 7(299):299ra122–299ra122, 2015.

[9] S. Hu. A recurrent network approach to g-computation for sepsis outcome prediction under dynamic treatment regimes. Master's thesis, Massachusetts Institute of Technology.

[10] M. Hughes, G. Hope, L. Weiner, T. McCoy Jr, R. Perlis, E. Sudderth, and F. Doshi-Velez. Semi-supervised prediction-constrained topic models. In *AISTATS*, pages 1067–1076, 2018.

[11] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya, J. Seekins, D. Mong, S. Halabi, J. Sandberg, R. Jones, D. Larson, C. Langlotz, B. Patel, M. Lungren, and A. Ng. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison, 2019.

[12] AEW Johnson, L Bulgarelli, L Shen, A Gayles, A Shammout, S Horng, TJ Pollard, B Moody, B Gow, LH Lehman, LA Celi, and RG Mark. MIMIC-IV, a freely accessible electronic health record dataset. *Nature Scientific Data*, 2023.

[13] D. Kingma and M. Welling. Auto-encoding variational bayes, 2022.

[14] L. Lehman, R. Adams, L. Mayaud, G. Moody, A. Malhotra, R. Mark, and S. Nemati. A physiological time series dynamics-based approach to patient monitoring and outcome predictio. *IEEE Journal of Biomedical and Health Informatics*, 19(3), 2015.

[15] L. Lehman, M. Johnson, S. Nemati, R. Adams, and R. Mark. *Advanced State Space Methods for Neural and Clinical Data*, chapter Bayesian nonparametric learning of switching dynamics in cohort physiological time series: application in critical care patient monitoring. Cambridge University Press, 2015.

[16] L. Lehman, R. Mark, and S. Nemati. A model-based machine learning approach to probing autonomic regulation from nonstationary vital-sign time series. *IEEE Journal of Biomedical and Health Informatics*, 22(1), 2018.

[17] D. Li, J. Fu, J. Zhao, J. Qin, and L. Zhang. A deep learning system for heart failure mortality prediction. *PLOS ONE*, 18(2), 2023.

[18] R. Li, S. Hu, M. Lu, Y. Utsumi, P. Chakraborty, D. M. Sow, P. Madan, J. Li, M. Ghalwash, Z. Shahn, and L. wei Lehman. G-net: a recurrent network approach to g-computation for counterfactual prediction under a dynamic treatment regime. In *Proceedings of Machine Learning for Health*, volume 158 of *Proceedings of Machine Learning Research*, pages 282–299. PMLR, 04 Dec 2021.

[19] X. Li, J. Ouyang, and X. Zhou. Supervised topic models for multi-label classification. *Neurocomputing*, 149:811–819, 2015.

[20] S. Linderman. Ssm: Bayesian learning and inference for state space models, 2022.

[21] S. Mallya, M. Overhage, N. Srivastava, T. Arai, and C. Erdman. Effectiveness of lstms in predicting congestive heart failure onset.

[22] J. Park, T. Hsu, J. Hu, C. Chen, W. Hsu, M. Lee, J. Ho, and C. Lee. Predicting sepsis mortality in a population-based national database: Machine learning approach. *Journal of Medical Internet Research*, 24(2), 2022.

[23] L. Rasmy, M. Nigo, B. Kannadath, Z. Xie, B. Mao, K. Patel, Y. Zhou, W. Zhang, A. Ross, H. Xu, and D. Zhi. Recurrent neural network models (covrnn) for predicting outcomes of patients with covid-19 on admission to hospital: model development and validation using electronic health record data. *The Lancet*, 4(6), 2022.

[24] P. Resnik, A. Garron, and R. Resnik. Using topic modeling to improve prediction of neuroticism and depression in college students. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1348–1353, 2013.

[25] A. Saeedi, Y. Utsumi, L. Sun, K. Batmanghelich, and L. wei Lehman. Knowledge distillation via constrained variational inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8132–8140, 2022.

[26] Sepsis. What is sepsis?, 2022.