INTERPRETABLE DEEP LEARNING ARCHITECTURES FOR MORTALITY PREDICTION INSIDE THE INTENSIVE CARE UNIT

A THESIS SUBMITTED TO AUCKLAND UNIVERSITY OF TECHNOLOGY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Supervisors

Prof. Jairo Gutierrez

Prof. Dave Parry

October 2020

By

William Caicedo-Torres

School of Engineering, Computer and Mathematical Sciences

Abstract

To improve the performance of Intensive Care Units (ICUs), the field of bio-statistics has developed scores which try to predict the likelihood of negative outcomes. These help evaluate the effectiveness of treatments and clinical practice, and also help to identify patients with unexpected outcomes. However, they have been shown by several studies to offer sub-optimal performance. Alternatively, Deep Learning offers state of the art capabilities in certain prediction tasks and research suggests deep neural networks are able to outperform traditional techniques. Nevertheless, a main impediment for the adoption of Deep Learning in healthcare is its reduced interpretability, for in this field it is crucial to gain insight on the why of predictions, to assure that models are actually learning relevant features instead of spurious correlations. To address this, we propose two deep convolutional architectures trained for the prediction of mortality using physiological and free-text data from the Medical Information Mart for Intensive Care III (MIMIC-III), and the use of concepts from coalitional game theory to construct visual explanations aimed to show how important these inputs are deemed by the networks. Our results show our models attain state of the art performance while remaining interpretable. Supporting code can be found at `https://github.com/williamcaicedo/ISeeU` and `https://github.com/williamcaicedo/ISeeU2`.

Contents

List of Tables

List of Figures

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.

Signature of candidate

Publications

Caicedo-Torres, W. & Gutierrez, J. (2020, May). ISeeU2: Visually Interpretable ICU mortality prediction using deep learning and free-text medical notes. ArXiV pre-print. https://arxiv.org/abs/2005.09284

Caicedo-Torres, W. & Gutierrez, J. (2019, October) ISeeU: Visually interpretable deep learning for mortality prediction inside the ICU. Journal of Biomedical Informatics, 98, 103269. https://doi.org/10.1016/J.JBI.2019.103269

Caicedo-Torres, W. (2019, September). Interpretable deep learning for mortality prediction inside the ICU. Poster session presented at the Medical Devices & Technologies Satellite Meeting Poster Session, of the Queenstown Research Week, Queenstown - New Zealand. Awarded second place in the Student Poster Presentation competition.

Caicedo-Torres, W. (2019, August). ISeeU: Visually Interpretable deep learning for mortality prediction inside the ICU. Paper presented at the Auckland University of Technology 3 Minute Thesis Finals, Auckland, New Zealand.

Caicedo-Torres, W. (2018, August). Interpretable deep learning architectures for mortality prediction inside the Intensive Care Unit. Paper presented at the Auckland University of Technology Postgraduate Research Symposium, Auckland, New Zealand.

Acknowledgements

I'd like to thank all the people who believed in me, especially my supervisor Prof. Jairo Gutierrez, who bet on me from day one. You are the essential reason for which this even happened on the first place. Also, I'd like to thank Prof. Dave Parry for his spot-on advice since day one, Dr. Janet Liang, for her invaluable help and expertise, and Dr. Chris Sames for his kindness and willingness to help and point me in the right direction.

A big thank you to my mums Doris and Cecilia. You raised me to the best of your abilities and went above and beyond to give me a shot at life.

Thanks to my Kiwi family, Linda and Rogan. Your love kept me alive during my darkest hours.

Thanks to all my friends and to those who have loved me, those from which I've learned so much. You all know who you are.

And last but not least, thank you my kids, Gabriel and Juan: I've been borrowing from your time, at a great cost to you. Your sacrifice and graciousness will never be forgotten.

Dedication

I call it magic...

Chapter 1

Introduction

1.1 Introduction

Intensive Care Units (ICUs) have helped to make improvements in mortality, length of stay and complication rates among patients (A. E. W. Johnson, Ghassemi et al., 2016), but they are costly to operate and sometimes skilled personnel to staff them seems to be in short supply (A. E. W. Johnson, Ghassemi et al., 2016). For this reason, research efforts to better equip ICUs to handle patients in a more cost-effective manner are warranted.

The field of bio-statistics have produced throughout the years a series of predictive scores which try to quantify the likelihood of negative outcomes (i.e. death) in clinical settings. These tools are necessary to evaluate the effectiveness of treatments and clinical practice, and to identify patients with unexpected outcomes (Rapsang & Shyam, 2014). Scores as the Apache (in its several versions), SAPS, MODS and others have had moderate success (Rapsang & Shyam, 2014). Although their performance is not optimal, they have become de facto standards for severity and mortality risk prediction. These scores have been built using statistical techniques such as Logistic Regression, which are limited to the modeling of linear decision boundaries, when it is quite likely

that the actual dynamics of the related biological systems do not respond to such prior. A reason for limiting the modeling to linear/additive techniques such as Logistic Regression is that they tend to be readily interpretable, allowing medical staff to derive rules and gain insight over the reasons why such a score is predicting certain risk or mortality probability. However, said statistical approaches (APACHE, SAPS, MODS, etc.), have been shown by several studies to generalize sub-optimally (Huang et al., 2013; Paul, Bailey & Pilcher, 2013). (Paul et al., 2013) show that over time, fixed scores performance tends to deteriorate (i.e. APACHE III-j over-predicts mortality in Australasia), and cite as possible reasons changes in medical practice and better care. It's no wonder then, that ICU mortality prediction appears to have reached a plateau (A. E. W. Johnson, Ghassemi et al., 2016).

1.2 Background

Severity scores represented a step forward for the prediction of ICU patient mortality, as they allowed practitioners to have an independent estimate of the seriousness of the patient condition, with less subjectivity than former practices (Rapsang & Shyam, 2014). The idea behind many of these scoring systems is to use certain information from the first day of stay to derive a number (the risk score) and the probability of mortality. The latter is important because dealing in probabilities allows for the comparison of several predictive scores. As it was mentioned before, the score itself is obtained by a linear (affine) combination of predictors, and after a logistic transform, a probability is calculated. Traditional statistical techniques are often used in order to lower the chance of model misspecification and to obtain predictor coefficients that are warranted from the data used to create the score. More over care is taken to arrive to a probabilistic model that is well calibrated (i.e. the predicted mortality probability distribution over all patients in the sample used to develop the model matches the actual empirical

distribution), and that has good predictive performance (i.e. its predictions correspond to the actual outcomes). For the sake of completeness we will discuss relevant details of some the most widely-known scoring systems.

1.2.1 The APACHE family

APACHE stands for Acute Physiology and Chronic Health Evaluation, and was devised as a measure of illness for patients in critical care (Knaus, Zimmerman, Wagner, Draper & Lawrence, 1981). Developed in 1981, 34 predictors from ICU admissions were used to create the model, with interesting results. In this case, predictors and their coefficients were chosen by consensus between a group of experienced ICU clinicians, rather than as the outcome of a data driven or statistical effort. APACHE II (Knaus, Draper, Wagner & Zimmerman, 1985) is a simplification of the original APACHE score that only includes 12 physiological variables, measured during the first 24 hours after admission (the worst measured values for each variable are used to calculate the Acute Physiology Score - APS), plus age and the existence of certain chronic conditions. This simplified version of the original APACHE score was validated using a large, multi-center sample of patients, and results shown that the score's performance was not adversely affected. APACHE III (Knaus et al., 1991) is another refinement that used largely the same predictors from APACHE II but it doesn't use the Glasgow Comma Scale to estimate neurological state. APACHE IV (Zimmerman, Kramer, McNair & Malila, 2006) is a subsequent iteration including new predictors as PO_2/FiO_2 ratio, the presence of mechanical ventilation, alongside a few others.

1.2.2 SAPS-II

SAPS-II or Simplified Acute Physiology Score II, is another severity scoring system that uses measurements from the first 24 hours of ICU stay (Gall, Lemeshow & Saulnier,

Physiological variable	High abnormal range				Normal range		Low abnormal range		
	4	3	2	1	0	1	2	3	4
Rectal temperature (°)	≥41	39-40.9	-	38.5-38.9	36-38.4	34-35.9	32-33.9	30-31.9	≤29.0
Mean arterial pressure (mm Hg)	≥160	130-159	110-129		70-109		50-69		≤49
Heart rate ventricular response	≥180	140-179	110-139		70-109		55-69	40-54	≤39
Respiratory rate per minute-non-ventilated or ventilated	≥50	35-490		25-34	12-24	10-11	6-9		≤5
Oxygen: A-a DO_2 or PO_2 (Torr)									
FiO_2 ≥0.5 record A-a DO_2	≥500	350-499	200-349		≤200	PO_2 61-70		PO_2 55-60	PO_2<55
FiO_2 ≥0.5 record only PO_2					PO_2>70				
Arterial pH	≥7.7	7.6-7.69		7.5-7.59	7.33-7.49		7.25-7.32	7.15-7.24	<7.15
Serum HCO_3 (mmol/L)-only if no ABGs	≥52	41-51.9		32-40.9	23-31.9		18-21.9	15-17.9	<15
Serum sodium (mmol/L)	≥180	160-179	155-159	150-154	130-149		120-129	111-119	≤110
Serum potassium (mmol/L)	≥7	6-6.9		5.5-5.9	3.5-5.4	3-3.4	2.5-2.9		≤2.5
Serum creatinine (μmol/L)	≥350	200-340	150-190		60-140		<60		
Hematocrit(%)	≥60		50-59.9	46-49.9	30-45.9		20-29.9		≤20
White blood cell count (x1,000/mm^3)	≥40		20-39.9	15-19.9	3-14.9		1-2.9		<1
Glasgow comma score = 15 minus actual GCS									

Table 1.1: Apache II APS components and weights. A-a DO_2: alveolar-arterial tension difference; PO_2: arterial oxygen tension; FiO_2: fractional concentration of inspired oxygen; HCO_3: Bicarbonate; ABG: Arterial Blood Gas. Additional weighting is given to the patient age and to the presence of chronic health conditions. Adapted from (Rapsang & Shyam, 2014)

.

1993). It constitutes an improvement over the original SAPS score developed in 1984 (Andreasen, 1984). The idea behind the development of the original SAPS was to provide a mortality score that relied on a fewer number of readily available predictors than competing alternatives (i.e. APACHE), in order to avoid the introduction of bias when dealing with missing observations. SAPS-II use 17 predictors (12 physiological plus age, type of admission and presence of three specific conditions) to output a score between 0 and 163 points, and a subsequent probability of mortality. First described in 1993, validation results showed superior performance when compared to the old SAPS. SAPS-II predictions were found to be consistent with the observed mortality in the validation dataset. Finally, a successor to SAPS-II called SAPS-III (Moreno et al., 2005) was developed in 2005, being comprised of three components: chronic information, acute information (including the presence of sepsis) and a physiological component. Observations in SAPS-III were to be made in the first hour of stay.

.

Physiological variable	Score																			
	26	13	12	11	9	7	6	5	4	3	2	0	1	2	3	4	6	7	9	10
HR (beats/min)				<40							40-69	70-119				120-159		≥160		
SBP (mm Hg)		<70						70-99				100-199		≥200						
Temperature (°C)												<39			≥39					
PO_2/FiO_2 only if VENT or CPAP				<100	100-199	≥200														
Urine output (L/day)				<0.5					0.5-0.999			≥1								
Urea (g/L)												<0.6					0.6-1.7			>1.8
TLC			<1									1-19.9			≥20					
Potassium											<3		3-4.9			≥5				
Sodium								<125				125-144	≥145							
Bicarbonate							<15			15-19		≥20								
Bilirrubin (mg/dL)												< 40				4-5.9			≥ 6.0	
Glasgow Coma Score	< 6	6-8				9-10		11-13				14-15								

Age	Score	Chronic Disease	Score	Type of Admission	Score
< 40	0	Metastatic cancer	9	Scheduled surgical	0
40-59	7	Hematological malignancy	10	Medical	6
60-69	12	AIDS	17	Emergency surgical	8
70-74	15				
75-79	16				
≥ 80	18				

Table 1.2: SAPS-II components and weights. HR: heart rate; SBP: Systolic blood pressure; PO_2: arterial oxygen tension; FiO_2: fractional concentration of inspired oxygen; VENT: ventilation; CPAP: Continuous Positive Airway Pressure; TLC: Total Leukocyte Count. AIDS: Acquired Immuno-Deficiency Syndrome. Adapted from (Rapsang & Shyam, 2014)

.

1.2.3 SOFA

The Sequential Organ Failure Assessment Score (SOFA) is a severity score created in 1994 (and later revised in 1996) by the European Society of Intensive Care Medicine (J. L. Vincent et al., 1996). SOFA tries to quantify severity based on the degree of organ failure experienced by patients. Something that makes SOFA different to other scoring systems is that there is no direct nor widely adopted formula for the calculation of mortality probability from the SOFA score (Rapsang & Shyam, 2014). Also SOFA can be calculated daily, with evidence suggesting that cumulative SOFA scores are more helpful to predict mortality (Rapsang & Shyam, 2014). SOFA considers possible failure in six organs with failure graded from 0 to 4.

Organ System	Score					
	Variable	0	1	2	3	4
Pulmonary	Lowest PO_2 (Torr) / FiO_2 (%)	> 400	≤ 400	≤ 300	≤ 200 + respiratory support	≤ + respiratory support
Coagulation	Lowest platelet (10^3 mm^3)	≥ 150	100-150	50-99	20-49	< 20
Hepatic	Highest billirubin (μmol/L)	< 20	20-32	33-101	102-24	> 204
Circulatory	Blood pressure/vasoactives administered	Mean arterial pressure < 70mmHg	Dopamine dose ≤ or dobutamine any dose	Dopamine dose >5 or epinephrine ≤ 0.1 or norepinephrine ≤ 0.1	Dopamine dose > 15 or epinephrine > 0.1 or norepinephrine > 0.1	
Neurologic	GCS	15	13-14	10-12	6-9	< 6
Renal	Highest creatinine level (μmol/L)	< 110	110-170	171-299	300-440	> 440
	Total urine output (mL/24h)				< 500	< 200

Table 1.3: SOFA components and weights. PO_2: arterial oxygen tension; FiO_2: fractional concentration of inspired oxygen; GCS: Glasgow Coma Scale. Adapted from (Rapsang & Shyam, 2014)

.

1.2.4 Deep Learning for ICU mortality prediction

On the other hand, Deep Learning offers state of the art capabilities in object recognition and several related areas, and those capabilities can be used to learn to detect patterns in patient data and predict the likelihood of negative outcomes. A reliable survival prediction system using Machine Learning concepts such as supervised fine-tuning (with pre-training that uses data from a related domain) and online learning (keep learning after deployment) could overcome the degradation problems exhibited by fixed scores, by being able to learn from the environments where they are being deployed. This would benefit ICUs everywhere, allowing staff to benchmark ICU performance and improve treatment protocols and practice (Rapsang & Shyam, 2014).

Machine Learning models depend on data for training, and in the case of Deep Learning, the amount of data needed to reach adequate performance can be larger than what traditional Machine Learning models require. However, today there is a deluge of data coming from various disparate sources, and said data sometimes sit in databases without much use. In the case of Electronic Medical Records, detailed information about patients such as visit records and socio-demographic data is stored indefinitely which could be leveraged to train predictive models that enable precision healthcare.

However, one of the main impediments for widespread adoption of advanced Machine Learning and Deep Learning in healthcare is lack of interpretability (Che, Purushotham, Khemani & Liu, 2016; Lipton, 2016). There seems to be a trade-off between predictive accuracy and interpretability in the landscape of learning algorithms, and in the case of Deep Neural Networks, models of greater depth consistently outperform shallower ones in some tasks (He, Zhang, Ren & Sun, 2016; Szegedy et al., 2015; Urban et al., 2017), at the expense of simpler representations. Crucially, high capacity Machine Learning models can easily latch onto epistemically flawed correlations and statistical flukes as long as they help minimize the loss in the training set, because the

minimization of the associated loss function does not care for causality but merely for correlation (Lipton, 2016). For instance, in one well-known case a neural network (Cooper et al., 1997) was trained to predict the risk of death in patients with pneumonia, and it was found that the model consistently predicted lower probability of death for patients who also had asthma. There was a counter-intuitive correlation in the training data that did not reflect any causality whatsoever, just the fact that asthma patients were treated more aggressively and thus fared better in average. The model in question performed better than the rest of models considered but it was ultimately discarded in favor of less performant, but interpretable ones. It is crucial then to offer mechanisms to gain insight on the why of predictions, i.e. the features our models attend to when generating an output, to make sure that models are actually learning sensible features instead of spurious and misleading correlations.

1.3 Research Questions

The limitations of traditional models from bio-statistics (e.g. performance degradation, inconsistent results) have motivated the use of Deep Learning architectures trained on clinical data to tackle the problem of mortality prediction inside the ICU. There is some literature showing the use of Deep Learning for ICU mortality prediction with various degrees of success. However the issue of interpretability for Deep Learning models persist, as previous efforts recorded by the literature have focused on either prediction accuracy, or on offering interpretability at its expense by using surrogate models. Given the current situation, we have formulated some research questions that we seek to answer in this work:

- How can we develop a Deep Learning model able to predict, with state-of-the-art performance, survival-related outcomes in a critical care setting?

- How can we provide useful explanations for the decisions of the model using feasible time and computing resources, without resorting to auxiliary classification models?

1.4 Research Methodology

The study methodology has been designed according to standard practice in Machine Learning (Shen, Wu & Suk, 2017; Suresh et al., 2017). This entails the following (1.1):

- Construction of a dataset to train, validate, and test the predictive models. Our datasets are created using the MIMIC-III (A. E. W. Johnson, Pollard et al., 2016) database. MIMIC-III contains data from ICU stays, including vital signs, lab measurements, free-text medical notes, and outcomes. We extract data related to vitals and lab measurements to construct physiological multi-variate time series, stacking the longitudinal physiological signals, in a way similar to (Suresh et al., 2017) and (Che, Purushotham, Cho et al., 2016); together with free-text medical notes written by ICU healthcare professionals involved in the care of patients. To access this database, a special protocol must be followed, including taking a short online course on patient privacy and adequate clinical data handling. Initially a set of entry criteria is defined in order to select a suitable subset of patients for the task at hand, following guidelines found in related works (Che, Purushotham, Khemani & Liu, 2016; Lipton, Kale, Elkan & Wetzell, 2015; Suresh et al., 2017).

- Deep Learning library selection. There are a number of popular libraries for Deep Learning (USENIX Association. et al., 2015; Chollet, 2015; Theano Development Team, 2016). At this stage, we review the most appropriate options in terms of expressiveness, ease of use, and performance.

- Execution environment selection. Given the computational-intensive nature of

Deep Learning, the training and validation process likely needs to be conducted in GPU-enabled computing instances. We examine the available choices and decide on the most appropriate to run our experiments.

- Model construction and model selection. Models for interpretable mortality prediction inside the ICU are constructed using the selected Deep Learning library/libraries. The datasets are partitioned into training and validation sets. Models are learned on the training set to predict patient outcome using electronic medical records data. Model selection, including hyper-parameter selection and architectural details, is carried out using K Fold cross-validation as guide. Given the size of MIMIC-III and the fact that Deep Learning models typically need a great volume of examples to generalize correctly (Goodfellow, Bengio & Courville, 2016) we anticipated the need for regularization and related techniques, like Weight Decay (Goodfellow et al., 2016), Early Stopping (Goodfellow et al., 2016), Dropout (Srivastava, Hinton, Krizhevsky, Sutskever & Salakhutdinov, 2014) and Batch Normalization (Ioffe & Szegedy, 2015). For the evaluation of model performance, the standard metrics of the field are used (Che, Purushotham, Khemani & Liu, 2016; Kale et al., 2015; Suresh et al., 2017), including Sensitivity (Recall), Specificity, and the Area Under the Receiving Operating Characteristic Curve (AUROC); and our results are compared to the most relevant works using the same dataset for mortality prediction, in terms of the aforementioned metrics and other relevant criteria.

- Model interpretability. After successful training of models, extraction of insight about the models' predictions is attempted. We construct suitable visualizations that convey how important are the input features to the models, at prediction time. Interpretability analysis is performed at the patient level and also at the dataset level when possible, to further characterize the impact of the input features on

the predictions of our models, so we can generate effective post-hoc explanations that help us clarify the predictive behavior of our models.

- Dissemination. Procedures and experimental results are recorded periodically throughout the course of our research for their dissemination in suitable venues, including conferences and journals. We also plan to open source our code, uploading it to a freely available, online source control repository.

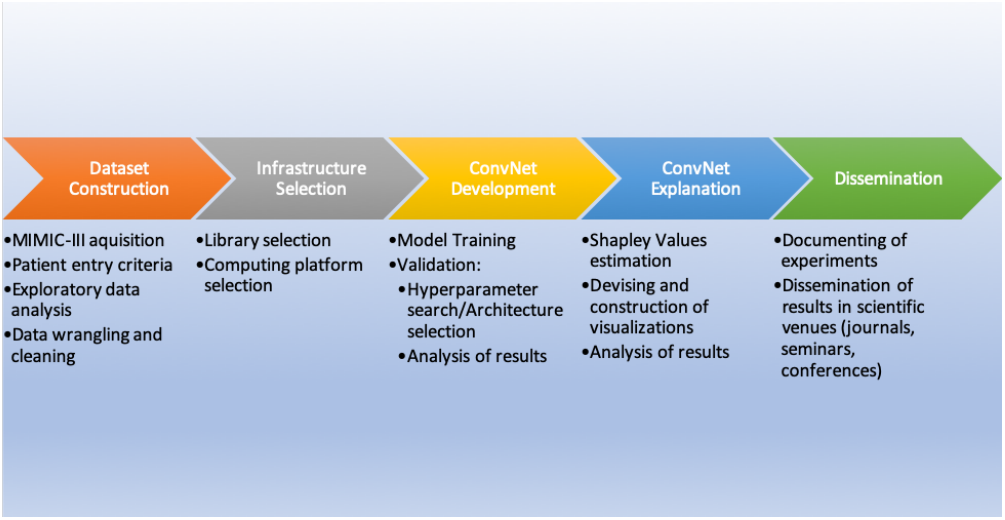


Figure 1.1: Research methodology.

1.5 Research Contributions

Mortality prediction inside ICU has plateaued (A. E. W. Johnson, Ghassemi et al., 2016) but we have more and more data, and the limited feature space commonly associated with the usual bio-statistical techniques is a possible reason for this paradox. There is growing evidence that supports the claim that Machine Learning methods over-perform traditional scores from bio-statistics (Silva, Moody, Scott, Celi & Mark, 2012). This is further evidenced in our literature review, where most papers present comparisons and benchmarks that involve traditional risk scores or Logistic Regression stand-ins,

and results show Machine Learning models finishing on top by a considerable margin, and Deep Learning models outperforming classical Machine Learning alternatives (Lipton et al., 2015; Che, Purushotham, Khemani & Liu, 2016; Che, Purushotham, Cho et al., 2016; Purushotham, Meng, Che & Liu, 2018). Here in our work we present SAPS-II and Logistic Regression (as a representative for traditional bio-statistics and Machine Learning) benchmarks in which experimental performance results show our Convolutional Neural models beating the baseline and alternative models. As motivating reasons for our choice of Deep Learning architecture we can mention that due to their architectural nature, Convolutional Neural Networks are easier to parallelize, sometimes offering superior performance than Recurrent Neural Networks (e.g. our ISeeU2 benchmark results). Also, temporal Convolutions (i.e. 1D convolutions - the kind we use in our proposed models) have been applied to sequence problems with success (Liang et al., 2014; Razavian & Sontag, 2015; Razavian et al., 2016; Nguyen et al., 2017). One of the big realizations of late in the Natural Language Processing and Speech Recognition fields is that Recurrent Neural Networks are not that necessary to solve Machine Learning problems with sequential inputs (Zhang, Zou & Gan, 2018; Van Den Oord et al., 2018; Vaswani et al., 2017).

The relationship of this work with the existing literature and its main contributions are summarized next. Our work relates to the existing literature in a number of ways. We use Deep Learning for mortality prediction inside the ICU as it also has been used by Che et al (Che, Purushotham, Cho et al., 2016; Che, Purushotham, Khemani & Liu, 2016), Grnarova et al (Grnarova et al., 2016) and Purushotham et al (Purushotham et al., 2018), but our work has key differences compared to the state of the art. We are presenting two ConvNet architectures for the prediction of ICU mortality, the first of which uses physiological time series data as input, while the second one uses free-text medical notes.

Our research contributions are,

- We are able to show that Deep Learning models called Convolutional Neural Networks (ConvNets) offer predictive performance comparable to the reported performance of Recurrent Neural Networks (RNNs) when dealing with physiological time-series data from MIMIC-III.

- We show evidence that a deep convolutional architecture can handle both static and dynamic data from MIMIC-III, making hybrid architectures (feed-forward plus recurrent) unnecessary for this particular task and performance level.

- We show that a convolutional architecture trained on MIMIC-III nursing free-text notes from the first 48 hours of patient stay using a standard log loss, can predict mortality with performance comparable to more involved approaches that use custom losses or different types of medical notes.

Regarding the problem of interpretability, the works most related to ours are Che et al (Che, Purushotham, Khemani & Liu, 2016) (physiological time-series inputs) and (Grnarova et al., 2016) (free-text medical notes inputs). However, in both cases there are also some important differences, which we highlight next:

- Che et al sidestep the problem of interpreting a deep model directly by using Mimic Learning with an interpretable student model (Gradient Boosted Trees) (Ba & Caruana, 2014), while our work focus instead on interpreting directly a deep model trained to predict ICU mortality, without using any surrogate model.

- Grnarova et al use an auxiliary prediction head in their model to obtain predictions of mortality using individual sentences. This is introduced as a regularization mechanism but doubles as a heuristic to provide interpretability by using the predicted probabilities as a proxy for sentence importance (Grnarova et al., 2016). Instead, we rely on a robust concept from coalitional game theory called the

Shapley Value (Shapley, 1953) to offer a more principled approach to feature importance.

- In the case of our physiological time-series data model, we are able to provide not only dataset-level interpretability but also patient-level interpretability. For our free-text model, we provide a way to annotate nursing notes with information about the words the model interprets as evidence for survival or death, in order to make notes more useful.

On the other hand, our physiological time series architecture uses multi-scale convolutional layers and a "channel" input representation, similar to (Suresh et al., 2017), but for a different task (mortality prediction instead of clinical intervention prediction). We also note that the use of Shapley Values (Shapley, 1953) or their approximations for providing interpretability in the ICU setting has not, to the best our knowledge, been reported by the relevant literature.

1.6 Organisation of Thesis

The thesis is organized as follows: An in-depth revision of the state of the art covering Deep Learning approaches to diagnostic problems using electronic medical records as learning substrate can be found in chapter two.

Chapter three introduces MIMIC-III and its structure, together with some exploratory data analysis showing some statistical descriptors and general trends. After this an introduction to Convolutional Neural Networks is presented. Lastly, we include some theoretical details from our game theory approach to provide interpretability.

Chapter four contains our experimental results. Details about the training and validation process are shown, and we present some neural architectures of interest with their respective performance metrics. Finally we show the application of the Shapley

Value to the interpretation of our best models, and the visualizations constructed for explanation purposes.

Chapter five deals with a discussion of the results obtained, and an analysis of our best model in terms of feature importance via our Shapley Value visualizations.

Finally we present our conclusions and suggested directions for further research.

Chapter 2

Literature Review

2.1 Introduction

According to Mitchell "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E" (Mitchell, 1997). Machine Learning is part of the broader area of Artificial Intelligence, which in turn is the study of intelligent agents and actors (devices, software) that plan a course of action to maximize their probability of achieving a specific goal (Russell, Norvig & Davis, 2016). It is generally believed that Artificial Intelligence (AI) as a study field in itself appeared with John McCarthy and the Dartmouth conference in 1956 (Russell et al., 2016). The field of AI was dominated on its first decades by the symbolic approach, which sought to replicate the process of human decision making by relying on the use of propositional and first order logic. This approach called for the specification of human knowledge in the form of rules and predicates and the use of powerful inference engines. Systems such as CYC (Elkan & Greiner, 1993), MYCIN (Shortliffe, 1977), DENDRAL (Lindsay, Buchanan, Feigenbaum & Lederberg, 1993), leveraged this approach and were introduced as "expert systems" in different domains, able to draw inferences from

facts and answer difficult questions based on their extensive knowledge bases. However, it was soon noted that these systems were too brittle and did not scale properly to real-world usage as the effort required to express human (often intuitive) knowledge was too high (Goodfellow et al., 2016).

This "sub-symbolic" approach of Machine Learning (ML), is different from the classical rule-based expert systems in that the focus of ML is to learn how to perform a task without being explicitly programmed to do so. To this end, ML draws heavily from statistical and probability theory, mathematical optimization, calculus, linear algebra and computer science (Goodfellow et al., 2016). ML tasks can be loosely categorized in three main groups: supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, the learning model is given samples of correct behaviour, and with enough samples it learns to approximate the mapping between inputs and outputs. In unsupervised learning, the model tries to figure out on its own a representation that describes adequately the statistical regularities of a dataset. In the case of reinforcement learning, the agent is given a reward that is a function of the quality of the plan it has devised to solve a multi-step task.

One of the many ML algorithms/models is the Neural Network (NN). NNs are actually a ML family of models inspired (sometimes loosely) in the brain (Goodfellow et al., 2016). A NN is an arrangement of processing units called neurons, each of which receives a set of inputs and produces an output according to an internal activation function (McCulloch & Pitts, 1943). The Multilayer Perceptron (Rumelhart, Hinton & Williams, 1986), a network composed of multiple layers of neurons connected in a feed-forward manner, is the quintessential example of such arrangement.

The underpinnings of NNs date at least back to the work of McCulloch and Pitts (McCulloch & Pitts, 1943) and Rosenblatt's Perceptron (Rosenblatt, 1957). Interestingly, NN development has enjoyed periods of over-inflated expectations followed by

disillusion (Goodfellow et al., 2016). These disillusion periods have usually corresponded to what is popularly known as "AI winters" (Crevier, 1993). After the initial hype consequence of Rosenblatt's early work, the interest in NNs took it first dip when Minsky and Papert showed some key shortcomings of the Perceptron (Minsky & Papert, 1972). Later, the second wave of interest in NNs was spearheaded by the efforts of the connectionist approach to brain understanding and the development of the Backpropagation algorithm (Goodfellow et al., 2016). After this second coming, the appearance of the Support Vector Machine and Kernel Methods among other factors, led to the view that NNs were outdated (Goodfellow et al., 2016). This attitude against NNs in the ML scene started to change circa 2008, with the advent of larger datasets and algorithms capable of training deep (in the sense of having many layers) networks. The third wave of NN research and development has been called Deep Learning.

Deep Learning is a Machine Learning subfield specialized in creating learning models that combine data in a hierarchical fashion in order to construct concepts with ever-increasing complexity. This hierarchical nature has allowed Deep Learning models to become the state of the art in tasks such as object recognition, facial recognition, speech recognition, Natural Language Processing, machine translation, among others (Goodfellow et al., 2016). Different models as the Convolutional Network (LeCun, Bottou, Bengio & Haffner, 1998), the Long Short Term Memory (Hochreiter & Schmidhuber, 1997b), Deep Belief Networks (Hinton, Osindero & Teh, 2006) and Deep Autoencoders (Bengio, Lamblin, Popovici & Larochelle, 2006) are being used by major firms in various real applications, ranging from voice-enabled search, to pedestrian detection in images (Goodfellow et al., 2016).

2.2 Deep Learning for clinical event prediction

Although the most natural application of Deep Learning algorithms to medical diagnosis is automated medical image diagnosis (Shen et al., 2017), we will concentrate on the usage of Physiological Time Series (PTS) and Electronic Medical Record (EMR) data, as a more general source of data on which machine learning models can be trained. EMRs are very attractive as a potential data source since their use is widespread, which makes them abundant and accessible electronically. However, there are certain challenges associated with their "secondary use" in Machine Learning (A. E. W. Johnson, Pollard et al., 2016). Despite this, several works have reported the successful use of EMRs and PTS to train Machine Learning/Deep Learning based models for diagnosis. Now follows a review of the most relevant literature about Deep Learning applications on clinical prediction based on EMR and PTS data, and then a review of literature related to the ICU setting, including mortality prediction. Table 2.1 shows a summary of the usage of Deep Learning models in the cited literature.

2.2.1 Medical diagnosis and event prediction

Lasko et al (Lasko et al., 2013) used a deep Autoencoder for unsupervised clinical phenotype discovery from serum uric acid measurements. In machine learning parlance, a phenotype corresponds to feature detectors (archetypes) and learned transformations to a feature space. Using these features, pre-trained logistic regression classifiers were constructed to distinguish between gout and leukemia, showing performance that rivalled that of a classifier that used features hand-crafted by a domain expert. In order to account for missing data (uric acid levels were sampled at irregular times, giving rise to a sparse design matrix), imputation based on Gaussian Process Regression (Rasmussen, 2006) was used.

An Autoencoder (Goodfellow et al., 2016) is a specialized neural architecture

Model	Works
Deep Autoencoders/Deep Belief Networks	(Lasko, Denny & Levy, 2013) (Liang et al., 2014) (Kale et al., 2015) (Miotto et al., 2016)
Convolutional Neural Networks	(Razavian & Sontag, 2015) (Razavian et al., 2016) (Grnarova et al., 2016) (Nguyen et al., 2017) (Suresh et al., 2017)
Recurrent Neural Networks	(Lipton et al., 2015) (Che, Purushotham, Cho et al., 2016) (Choi, Bahadori, Schuetz, Stewart & Sun, 2016) (Choi, Bahadori, Sun et al., 2016) (Lipton, Kale & Wetzel, 2016) (Pham, Tran, Phung & Venkatesh, 2016) (Razavian et al., 2016) (Suresh et al., 2017)
Hybrid Models	(Esteban et al., 2016) (Che, Purushotham, Khemani & Liu, 2016)

Table 2.1: Usage of different Deep Learning models on the literature.

which learns an efficient representation of the data in an unsupervised fashion. In an Autoencoder, the task consists in trying to recover the input from a constrained representation of it (i.e. minimize some function that captures the discrepancy between inputs and outputs).

The simplest Autoencoder is composed by three layers. Layers one and two represent the encoder, and the activations of the middle layer correspond to the latent representation discovered by the Autoencoder, given by $h = g(Wx + b)$, where g corresponds to a non-linearity (usually sigmoidal). On the other hand, the decoder outputs the reconstructed version of the input, $\hat{x} = g'(W'h + b')$ (g' corresponds to a non-linearity as well), with $W' = W^T$ usually. The loss is often expressed as a cross-entropy between the distributions of the inputs and outputs. In the case of deep Autoencoders (also known as stacked Autoencoders), a greedy layer-wise approach (Bengio et al., 2006) is used: at every step, a new feature layer is learned by minimizing the reconstruction error of the input. Once training has ended, the latent representation found, h_i, is used

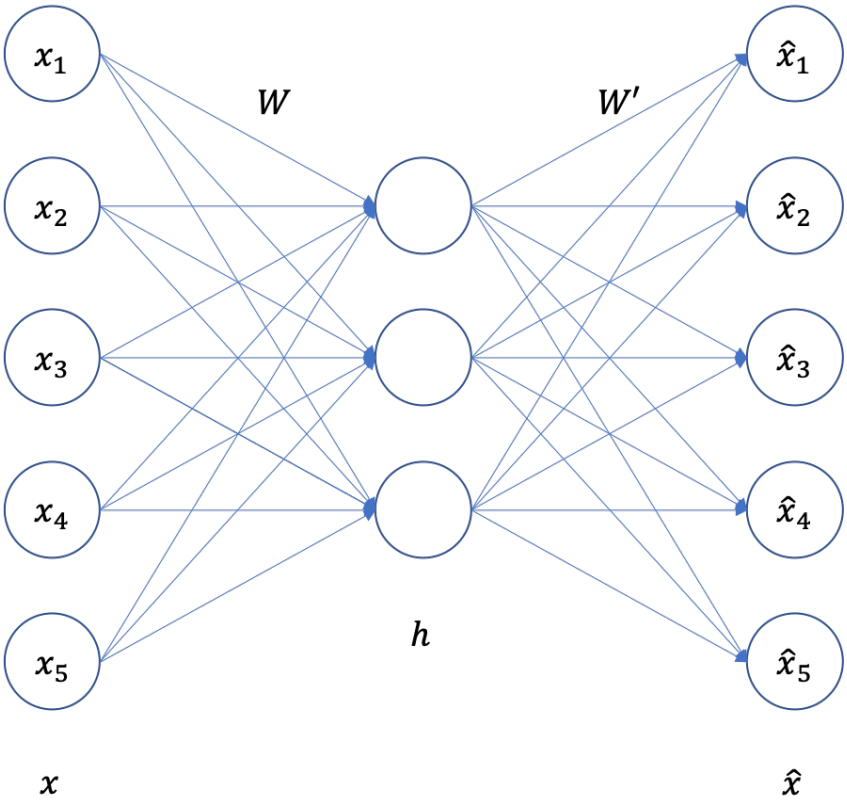


Figure 2.1: A simple autoencoder.

as the new input to a new Autoencoder, and a new representation h_{i+1} is discovered by repeating the training process. After a series of training stages, a set of feature layers, each with an increased abstraction level, are produced. If such layers are used to build a supervised classifier, the procedure is called unsupervised pre-training (Bengio et al., 2006).

Liang et al (Liang et al., 2014) developed a Deep Belief Network (DBN) (Bengio, 2009) to learn to diagnose different diseases, using two clinical EMR datasets. The first data set contained information about the diagnosis of 21 different diseases, and the second one had information about hypertension patients and a number of clinical and para-clinical features. In this study, the DBN acted as a feature extractor, similar

as how an Autoencoder is used to perform unsupervised pre-training. The DBN was composed of stacked Restricted Boltzmann Machines, a two-layer undirected graphical model which uses the Contrastive Divergence algorithm for training (Bengio, 2009) (Hinton et al., 2006). After this, supervised fine tuning was carried out, using an SVM as the top classifier. Backpropagation was used to train the DBN and SVM weights.

Diagnosis	DBN+SVM	SVN	decision tree
Arthralgia Syndrome	84.17	74.98	63.04
Acne	86.02	76.68	69.80
Epilepsy	69.67	71.57	68.91
Tinnitus & deafness	86.18	73.15	72.93
Abdominal pain	80.28	72.86	74.19
Allergic rhinitis	69.05	65.84	75.09
Neck & shoulder pain	72.85	71.11	65.52
Cervical spondylosis	78.48	61.42	73.59
Cough	87.11	72.12	73.10
Facial paralysis	87.26	58.64	63.25
Traumatic brain injury	70.31	63.54	62.38
Migraine	87.38	58.92	69.97
Ankylosing Spondylitis	87.10	59.94	79.19
Insomnia	77.19	74.47	66.81
Headache	83.81	71.90	71.71
Flaccidity Syndrome	69.98	64.34	64.48
Stomachache	75.86	77.00	75.03
Asthma	86.23	58.69	65.10
Palpitation	83.64	66.77	70.12
Lumbocrural pain	87.15	65.63	73.98
Urticaria & Rubella	80.77	73.31	77.82

Table 2.2: Results (accuracy) on the first dataset. Best results are in bold (Liang et al., 2014).

As part of the experiments in (Liang et al., 2014) the DBN + SVM model was compared to a standard SVM and a decision tree, all trained using the same data. In both classification tasks, the DBN+SVM model showed superior performance for almost all classes (table 2.2, table 2.3). According to the authors, for the cases where the shallow models showed better performance, substantial professional expertise to handcraft features was required. This suggests that the power of learned hierarchic

representations is indeed superior to that of shallow models.

	DBN+SVM	SVN	decision tree
Inspection	84.32	81.11	79.40
Tongue	83.29	78.94	78.55
Inquiry	79.57	80.90	78.48
Palpation	79.94	76.34	75.67
Others	80.45	79.02	78.93
Fusional	85.23	82.31	81.08

Table 2.3: Results (precision) on the second data set. Best results are in bold (Liang et al., 2014).

Miotto et al show the use of Stacked Denoising Autoencoders to build an unsupervised representation called "Deep Patient" using EMR data (Miotto et al., 2016). Deep Patient used three representation layers learned out from approximately 700.000 patient records from Mount Sinai Hospital. Miotto et al mention that the use of EMRs as learning substrate is ridden with difficulties, for instance, inconsistencies in expression of patient phenotypes (Miotto et al., 2016), their high-dimensionality, the free-text nature of clinical notes, among others.

The general approach of Miotto et al was unsupervised feature discovery and representation using EMR as training data, followed by supervised fine-tuning for prediction of disease. SDAs are trained to minimize the reconstruction error and the top classifier is trained to minimize the negative log-likelihood (cross-entropy error), similar as in (Kale et al., 2015). In the experiments reported, the results of this approach overperform other common representations as Principal Component Analysis, Gaussian Mixture Models, K-Means and Independent Component Analysis (Miotto et al., 2016). Results can be seen in Table 2.4 and Table 2.5. Deep Patient showcases the power of Deep Learning to deliver great results without resorting to expensive manual feature design. One interesting characteristic of the approach of Deep Patient is that it is able to learn compact, general purpose representations, suitable to other medical prediction tasks, using EMR data as learning substrate; as shown in (Miotto et al., 2016; Kale et

al., 2015; Liang et al., 2014; Lasko et al., 2013).

Patient representation	AUROC	Accuracy	F-Score
RawFeat	0.659	0.805	0.084
PCA	0.696	0.879	0.104
GMM	0.632	0.891	0.072
K-Means	0.672	0.887	0.093
ICA	0.695	0.882	0.101
DeepPatient	0.773	0.929	0.181

Table 2.4: Comparison of the performance of different representations. Deep Patient comes on top, with a statistically significant (95% confidence) lead over the second best performer (Miotto et al., 2016).

Disease	RawFeat	PCA	DeepPatient
Diabetes mellitus with complications	0.794	0.861	0.907
Cancer of rectum and anus	0.863	0.821	0.887
Cancer of liver and intrahepatic bile duct	0.830	0.867	0.886
Regional enteritis and ulcerative colitis	0.814	0.843	0.870
Congestive heart failure (non- hypertensive)	0.808	0.808	0.865
Attention-deficit and disruptive behavior disorders	0.730	0.797	0.863
Cancer of prostate	0.692	0.820	0.859
Schizophrenia	0.791	0.788	0.853
Multiple myeloma	0.783	0.739	0.849
Acute myocardial infarction	0.771	0.775	0.847

Table 2.5: Deep Patient performance (AUROC) per disease. Best results are in bold. Source (Miotto et al., 2016).

Razavian et al (Razavian & Sontag, 2015) use a Convolutional Neural Network (ConvNet) trained on lab tests to predict disease onset. ConvNets are Multi-Layer Neural Networks that use a particular architecture with sparse connections and parameter sharing (LeCun et al., 1998). They can be thought of performing a discrete convolution operation between the input (often a two-dimensional image) and a set of trainable kernels at each layer. The discrete convolution operation, in the context of Deep Learning and computer vision is defined as

$$s(i,j) = (x * w)(i,j) = \sum_{m,n} I(m,n)K(i-m, j-n) \tag{2.1}$$

where I is a two-dimensional image and K is a two-dimensional kernel. The kernel is a local feature detector that is displaced all over the image. Each convolution between the input and a kernel produces a spatial receptive field, also called a feature map. After the convolution operation, the output of the receptive field is run through a non-linear activation function which allows the network to work with transforms of the input space and construct non-linear features. The feature map can be thought as a 2-D tensor (matrix) of neurons, where the weights of each neuron are the same but shifted spatially (hence the parameter sharing), and which are not connected to every single pixel of the input (which also can be seen as having the corresponding weight set to zero). ConvNets were one the first models to use Gradient Descent with Backpropagation (Rumelhart et al., 1986) with success (Goodfellow et al., 2016). Convolution based filters are extensively used to detect features as shapes and edges in computer vision (Shapiro, 2001). However, fixed kernels are used to detect specific features, in contraposition to ConvNets where kernels are learned from the data.

(Razavian & Sontag, 2015) present a multi-resolution convolutional architecture which resembles the organizing principles and philosophy behind GoogLeNet and the Inception module (Szegedy et al., 2015). However, instead of using different convolution sizes in parallel as part of the same layer/module, Razavian et al use max pooling with different neighbourhood sizes, which they call, resolution levels (Razavian & Sontag, 2015), hence a multi-resolution ConvNet. In this fashion, their architecture is able to deal with both short-term and long-term features, taking into account the fact that the temporal dynamics of the physiological variables measured, are different from each other. The dataset used by (Razavian & Sontag, 2015) comes from 298.000 individuals over 8 years, for which 18 lab measurements have been recorded. This data is used to train the ConvNet to perform early detection (at least three months in advance) of 171 diseases and conditions. The input representation is similar to the one used by Lasko et al (Lasko et al., 2013), consisting of stacked physiological time series (lab test

measurements) from a backward window of 36 months, but in this case, they are fed to a ConvNet instead of a Deep Belief Network as in (Lasko et al., 2013). The output labels (i.e. patient diagnostic class) were assigned to patients only if there were two independent diagnosis annotations of a specific disease in a period of 24 + 3 months after the backward observation window.

The authors make the point that the utilization signal, i.e. the spacing between lab measurements is predictive in itself, because it reflects physician intuitions about what could be wrong with the patient, so they include this signal into the input representation and perform experiments to find out whether the inclusion of this signal improves the ConvNet performance.

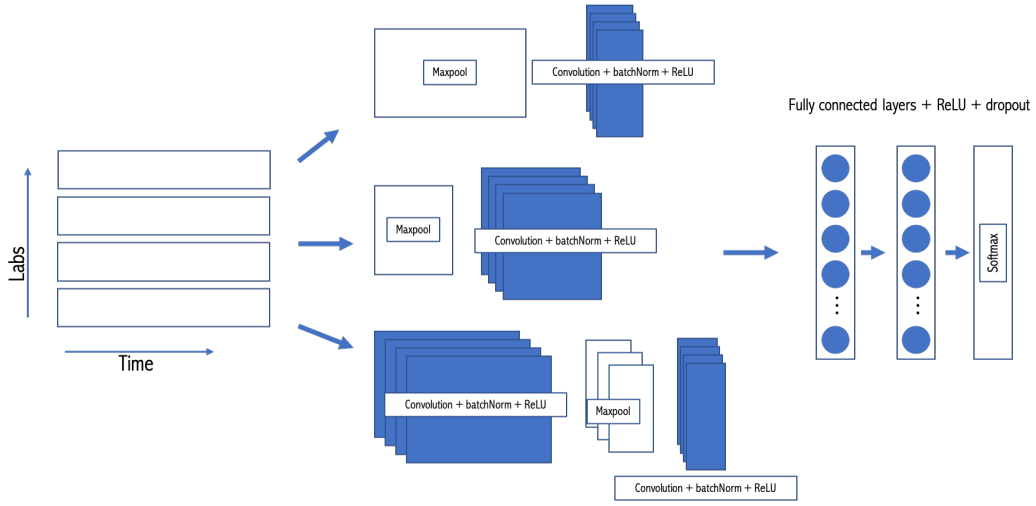


Figure 2.2: Temporal ConvNet architecture.

In another contribution of (Razavian & Sontag, 2015), the authors propose a novel method to perform imputation on the dataset. The proposed architecture has a mathematical formulation roughly equivalent to Multivariate Gaussian Processes. It consists of a differentiable, convolutional, non-parametric kernel regression formulation; where the kernels are learned from the data using back-propagation (Rumelhart et al., 1986). Assuming a model $x = f(t) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$, and a set of observed data points

$x_{t_1}, x_{t_2}, \ldots, x_{t_n}$; we can define the regression estimation as

$$x(t_{new}) = \mathbb{E}_{x \sim P(x|t=t_{new})}[x] \tag{2.2}$$

$$\mathbb{E}_{x \sim P(x|t=t_{new})}[x] = \int_x xP(x|t = t_{new})dx = \int_x x\frac{P(x, t = t_{new})}{P(t_{new})}dx \tag{2.3}$$

Using Kernel Density Estimation to approximate the probability distributions, the Nadaraya-Watson estimator (Nadaraya, 1964; Watson, 1964) allows to reformulate the above as

$$\mathbb{E}_{x \sim P(x|t=t_{new})}[x] = \frac{\sum_{i=1}^{n} x_{t_i} K(t_{new}, t_i)}{\sum_{i=1}^{n} K(t_{new}, t_i)} \tag{2.4}$$

where $K(t_i, t_j)$ is a positive semi-definite kernel function. At this point, using functional formulations for the data points and time that involve the Dirac Delta (δ), the authors express the expected value in a convolutional manner,

$$\mathbb{E}_{x\ simP(x|t=t_{new})}[x] = \frac{(K * \hat{X}_{train})(t_n ew)}{(K * I(\hat{X}_{train:observed})(t_n ew)} \tag{2.5}$$

Here $\hat{X}_{train}(t) = \sum_{i=1}^{n} x_{t_i}\delta(t, t_i)$ and $\delta(t, \tau_0) = 1$ when $t = \tau_0$ and zero for every other value of t. On the other hand, $I(\hat{X}_{train:observed}) = \sum_{i=1}^{n} \delta(t, t_i)$ is a function that outputs a 1 for every observation t_i. As previously said, the Kernel function is learned using Back-propagation. The authors perform imputation separately before classification, even though they mention that the differentiable nature of both parts of the pipeline makes it amenable to end-to end training (Razavian & Sontag, 2015). Razavian et al extend the former convolutional regression to the 2D case, using the full matrix of lab observations to impute the value of the missing data, and compare the performances of both approaches.

The reported results show that the convolutional regression attains lower MSE at data imputation than the non-convolutional formulation and standard Gaussian Process

regression, especially in the 2D (multivariate) case. At classification time, the ConvNets trained on imputed data show a little edge over ConvNets trained on raw (non-imputed data with missing values set to zero). However, the differences in performance are not dramatic and don't seem to warrant the added pipeline complexity. In general, the ConvNets showed better performance than selected baselines (Multi-layer Perceptron and Logistic Regression).

Pham et al (Pham et al., 2016) developed DeepCare, a model based on the Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997b) to predict disease progression, needed interventions (treatment), and future risk (i.e. unplanned admissions to the hospital). The Long Short-Term Memory, is a specialized Recurrent Neural Network (RNN) with neurons explicitly designed to make it easier to learn long-term dependencies (Hochreiter & Schmidhuber, 1997b). The LSTM network and a related model, the Gated Recurrent Unit (GRU) network (Cho, van Merrienboer, Bahdanau & Bengio, 2014; Chung, Gülçehre, Cho & Bengio, 2014), belong to the class of Gated RNNs (Goodfellow et al., 2016). LSTMs capitalize on the idea of creating shorter paths in time for the gradient to flow more easily. This idea had been previously introduced in the form of skip connections (Mozer, 1992), where some recurrent connections bypassed entire time-steps; and Leaky Units (El Hihi & Bengio, 1995), where inside each neuron of the recurrent state, an inner linear self-connection was maintained in order to give the neuron a chance to remember important information.

The dataset used in this work was constructed using EMRs from a large regional hospital in Australia (Pham et al., 2016), including data from a cohort of diabetic patients and a cohort of mental patients. A language is built around ICD codes for diagnoses, Current Procedural Terminology (CPT) or International Classification of Health Interventions (ICHI) codes for interventions, and Anatomical Therapeutic Chemical (ATC) codes for medication. An embedding layer is used to encode the language words into a vector representation that captures the information of patient hospital visits. Each

visit is coded as a collection of diagnoses, a collection of interventions, the type of admission (planned or unplanned), and the time between the last admission and the current one. Before feeding the admission data to the LSTM, diagnosis vectors are pooled, and the intervention vectors are pooled as well.

The authors propose an interesting way of feeding the LSTM at each admission time: only the diagnosis vector is presented as input in a traditional way, whereas the intervention vector is used to separately influence the LSTM output and forget gates, as the authors hypothesize that interventions somewhat "erase" disease from the sequence. Also, the type of admission code (1 for planned, 2 for unplanned) is used separately to modulate the input gate. Lastly, the time between visits is also modeled as a weight on the value of the forget gate itself.

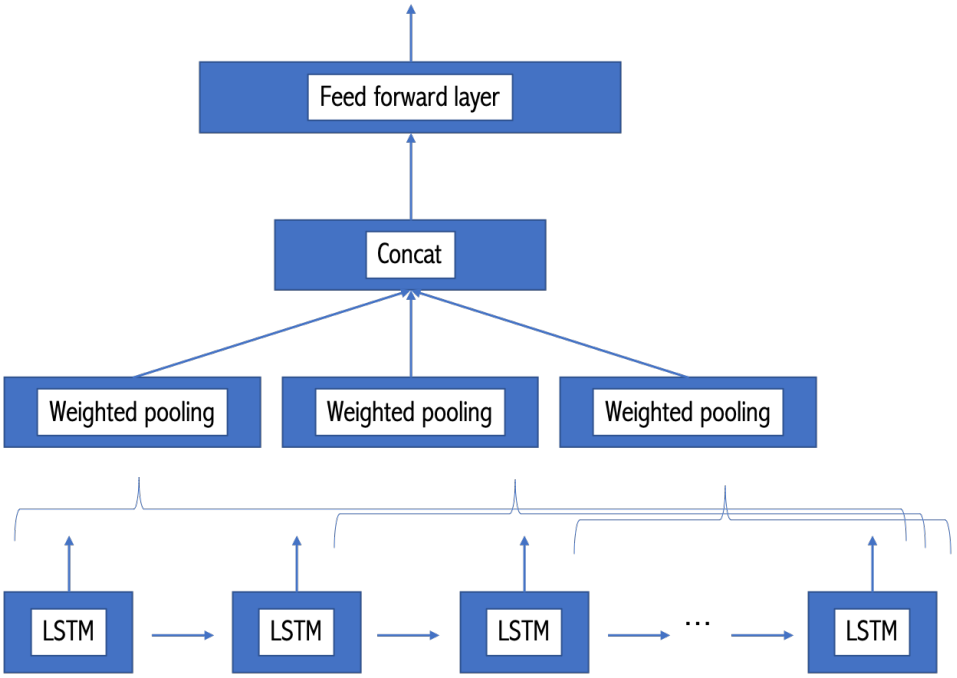


Figure 2.3: DeepCare Architecture. The bottom layer is a LSTM RNN, followed by a multi-scale pooling (attention) layer, and a feed-forward neural network on the top layer..

All hidden states generated by the LSTM are then pooled by a multi-scale pooling

layer that resembles an attention mechanism (but without trainable parameters) to produce a single vector representation that is fed to a feed-forward network to obtain a prediction. The results reported beat a series of baselines, and show promise for the three prediction tasks considered.

In (Razavian et al., 2016) Razavian et al propose a refinement of the Convolutional architecture they had already presented in (Razavian & Sontag, 2015) and a Long Short Term Memory (LSTM) network (Hochreiter & Schmidhuber, 1997b) for multilabel prediction of disease onset. This work is essentially an offshoot of (Razavian & Sontag, 2015), given that they use the same input representation, dataset, and define similar prediction tasks.

The refined ConvNet architecture, which the authors call CNN2, has two layers which convolve vertically over the different labs, followed by Max-Pooling. Then a subsequent convolutional layer that acts over time. The output of the last convolutional layer is max-pooled and fed into the first of two fully-connected layers, followed by a Softmax. In this way, they attempt to capture not only the time dependent features, but also the correlation between labs. The authors report the use of Dropout and Batch Normalization. Test results of this architecture are compared with the original multi-resolution ConvNet (CNN1) of (Razavian & Sontag, 2015), the LSTM model, an ensemble of deep models (Ens), and a shallow baseline consisting on a Logistic Regression (LR) classifier using hand-engineered features, with the CNN2 architecture narrowly surpassing the rest in most tasks (Table 2.6).

Choi et al (Choi, Bahadori, Schuetz et al., 2016) presented Doctor AI, a Deep Learning model to predict clinical events (diagnoses, medication orders, time between visits), trained on time-stamped EMR data from Sutter Health Palo Alto Medical Foundation. This work is similar to (Lipton et al., 2015), which also use RNNs, and to (Pham et al., 2016) in the sense that both use longitudinal EMR data to model the next patient visit events. Each past visit is modeled as a multi-hot vector $x_i \in (0,1)^D$

ICD9 Code and disease description	LR	LSTM	CNN1	CNN2	Ens	Pos
585.6 End stage renal disease	0.886	0.917	0.910	0.916	0.920	837
285.21 Anemia in chr kidney dis	0.849	0.866	0.868	0.880	0.879	1598
585.3 Chr kidney dis stage III	0.846	0.851	0.857	0.858	0.864	2685
584.9 Acute kidney failure NOS	0.805	0.820	0.828	0.831	0.835	3039
250.01 DMI wo cmp nt st uncntrl	0.822	0.813	0.819	0.825	0.829	1522
250.02 DMII wo cmp uncntrld	0.814	0.819	0.814	0.821	0.828	3519
593.9 Renal and ureteral dis NOS	0.757	0.794	0.784	0.792	0.798	2111
428.0 CHF NOS	0.739	0.784	0.786	0.783	0.792	3479
V053 Need prphyl vc vrl hepat	0.731	0.762	0.752	0.780	0.777	862
790.93 Elvtd prstate spcf antgn	0.666	0.758	0.761	0.768	0.772	1477
185 Malign neopl prostate	0.627	0.757	0.751	0.761	0.768	761
274.9 Gout NOS	0.746	0.761	0.764	0.757	0.767	1529
362.52 Exudative macular degen	0.687	0.752	0.750	0.757	0.765	538
607.84 Impotence, organic orign	0.663	0.739	0.736	0.748	0.752	1372
511.9 Pleural effusion NOS	0.708	0.736	0.742	0.746	0.749	2701
616.10 Vaginitis NOS	0.692	0.736	0.736	0.746	0.747	440
600.01 BPH w urinary obs/LUTS	0.648	0.737	0.737	0.738	0.747	1681
285.29 Anemia-other chronic dis	0.672	0.713	0.725	0.746	0.739	1075
346.90 Migrne unsp wo ntrc mgrn	0.633	0.736	0.710	0.724	0.732	471
427.31 Atrial fibrillation	0.687	0.725	0.728	0.733	0.736	3766
250.00 DMII wo cmp nt st uncntr	0.708	0.718	0.708	0.719	0.728	3125
425.4 Prim cardiomyopathy NEC	0.683	0.718	0.719	0.722	0.726	1414
728.87 Muscle weakness-general	0.683	0.704	0.718	0.722	0.723	4706
620.2 Ovarian cyst NEC/NOS	0.660	0.720	0.700	0.711	0.719	498
286.9 Coagulat defect NEC/NOS	0.690	0.694	0.709	0.715	0.718	958

Table 2.6: Test results of the different classifiers for the top-25 diseases in terms of ROC-AUC. In bold, diseases for which the proposed models improve the AUC by at least 0.05 compared to Logistic Regression. Pos refers to the number of positive examples in the test set for each disease. Source (Razavian et al., 2016).

where D is the total number of possible diagnosis, medication and procedures codes that can appear on a particular medical consultation. Given that the dimensionality of such vectors is high, an embedding layer is used for dimensionality reduction. The embedding layer was initialized by either 1. using orthonormal matrices, or 2. using the skip-gram embedding (Mikolov, Sutskever et al., 2013). A sequence of vectors $[h_i^{(1)}, d_i]$ (where $h_i^{(1)}$ is the compressed representation of the input generated by the embedding layer and d_i is the time elapsed between the last and current visit) represents the EMR history of a patient. A two-layer deep Gated Recurrent Unit (GRU) based RNN receives the vector sequence as input, and a Softmax layer uses the deep GRU layer output to predict the diagnosis, medication and procedure codes of the next visit, as well as the time until said next visit. Results show Doctor AI outperforming several baselines such as last visit codes, most frequent codes for a patient, Logistic regression, and a Multilayer Perceptron, in terms of Recall@k (number of true positives in the top k predictions divided by the number of true positives). The Recall@10 results are interesting and promising, but still fall short of a truly usable decision support system.

The authors of (Choi, Bahadori, Schuetz et al., 2016) also show that the features learnt by Doctor AI can be transferred to different prediction settings (different datasets, different hospitals). They trained Doctor AI on the MIMIC-II dataset (Saeed et al., 2011) and reported a sizable performance boost when using the original parameters of the model obtained on the Sutter Health data set as initial values. This suggests that the model generalized well and could be deployed across different hospitals with supervised fine-tuning.

Esteban et al (Esteban et al., 2016) developed a hybrid recurrent + feed-forward architecture to combine static and dynamic information to predict renal transplant outcomes (rejection, loss, death) in the next 6 and 12 months. Their network used a RNN (LSTM (Hochreiter & Schmidhuber, 1997b), GRU (Chung et al., 2014)) to compute a temporal representation out of the lab measurements taken for each patient

in their visits to the clinic, and a feed-forward network to compute a representation for the static data of the patient. Both representations are concatenated into a single vector which is fed to a Softmax output layer.

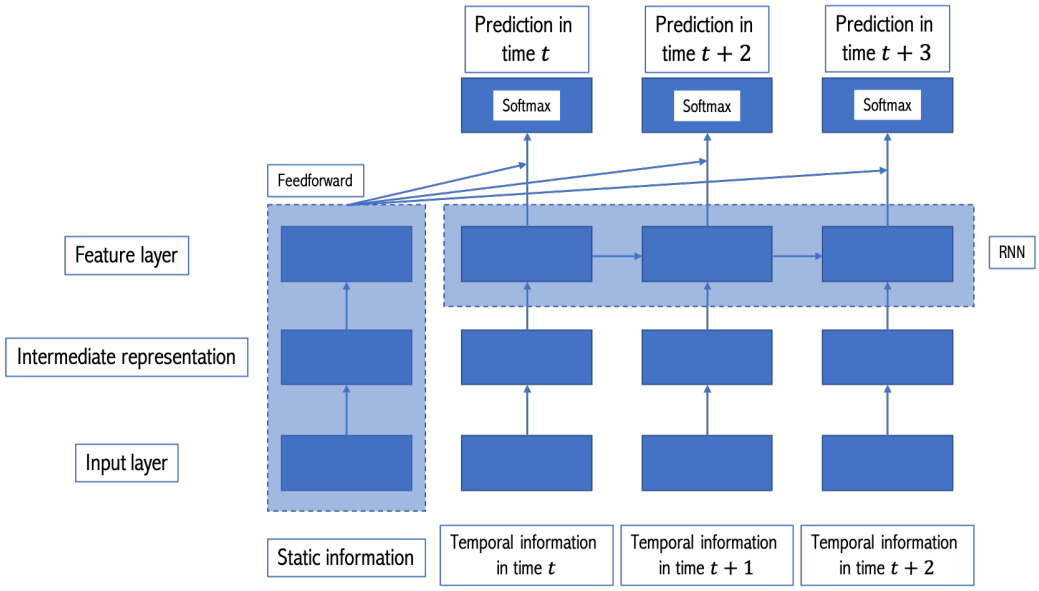


Figure 2.4: Hybrid RNN + feed-forward architecture from (Esteban et al., 2016).

Results (Area Under the Precision-Recall curve, Area Under the Receiving Operator Characteristic curve) show that the hybrid networks outperform several baselines (Table 2.7).

Models	AUPRC	AUROC
GRU + static info	0.345 ± 0.013	0.833 ± 0.006
LSTM + static info	0.330 ± 0.014	0.826 ± 0.006
RNN + static info	0.319 ± 0.012	0.822 ± 0.006
TLE	0.313 ± 0.010	0.821 ± 0.005
Logistic Regression	0.299 ± 0.009	0.808 ± 0.005
Random	0.073 ± 0.002	0.5

Table 2.7: Results of several models on renal transplant event prediction. TLE (Temporal Latent Embeddings) corresponds to a Multilayer Perceptron. Source (Esteban et al., 2016).

In (Choi, Bahadori, Sun et al., 2016) Choi et al introduced RETAIN, a new recurrent architecture based on attention (Bahdanau, Cho & Bengio, 2014) (Cho, Courville &

Bengio, 2015) to predict diagnoses based on EMR data from patient visits to the doctor at Sutter Health Palo Alto Medical Foundation. The purpose of the RETAIN network is very similar to other works listed here that use RNNs (Choi, Bahadori, Schuetz et al., 2016) (Pham et al., 2016), but RETAIN is an architecture specifically designed to improve its interpretability. The main difference of the RETAIN architecture with respect to other RNNs with attention is that, the recurrent mechanism does not operate on the hidden layer(s) of the network itself, but on the calculation of the attention coefficients. So, RETAIN can be seen as inverting the traditional RNN plus attention architecture, as traditional RNNs use feed-forward mechanisms to compute the attention coefficients and recurrent loops for the hidden activations in each time-step. Another characteristic of RETAIN's operation is that the attention coefficients are calculated going backwards in time, as seen in Figure 2.5.

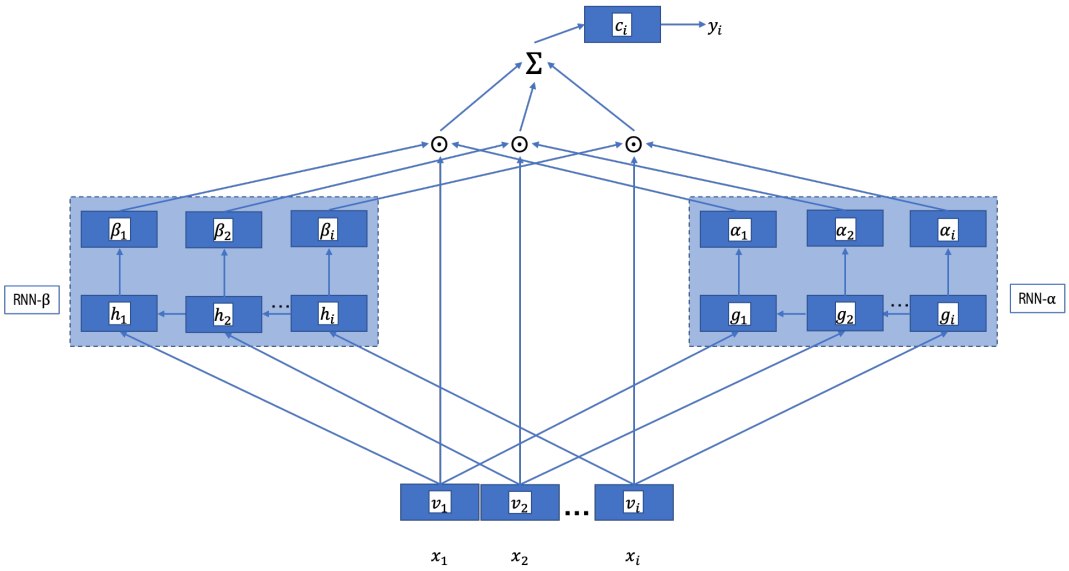


Figure 2.5: RETAIN architecture.

RETAIN's architecture use a patient visit representation similar to (Choi, Bahadori, Schuetz et al., 2016) and a linear embedding to calculate a time-step activation v_i for each patient. Then, two recurrent networks (RNN_α, RNN_β) are used to calculate the

attention coefficients for each v_i. RNN_α calculates a single scalar attention α_i (the visit-level attention), while RNN_β calculates an attention vector β_i (the event-level attention). Given that the activations v_i are computed by a linear embedding W_embx_i, it is straightforward to calculate the contribution not only of each visit, but the contribution of each clinical event $x_{i,k}$. In fact, breaking down the relevant equations, the contribution ω of each clinical event to the predicted outcome y_j is given by

$$\omega(y_j, x_{i,k}) = \alpha_i W(\beta_k \odot W_{emb}[:, k])x_{i,k} \tag{2.6}$$

where W is the Softmax layer weight matrix. The results 2.8 show that RETAIN achieves competitive performance with standard RNNs while offering added interpretability.

Model	Test Neg Log Likelihood	AUC
LR	0.3269 ± 0.0105	0.7900 ± 0.0111
MLP	0.2959 ± 0.0083	0.8256 ± 0.0096
RNN	0.2577 ± 0.0082	0.8706 ± 0.0080
RNN+α_M	0.2691 ± 0.0082	0.8624 ± 0.0079
RNN+α_R	0.2605 ± 0.0088	**0.8717** ± 0.0080
RETAIN	**0.2562** ± 0.0083	0.8705 ± 0.0081

Table 2.8: RETAIN performance (AUC) compared to several baselines. LR is Logistic Regression, MLP is a single hidden layer Multilayer Perceptron, RNN is a deep (two hidden layer) GRU-RNN. RNN+α_M corresponds to a standard RNN with attention. RNN+α_R is similar to RNN+α_M but the attention coefficients are calculated backwards in time. Best results are in bold. Source (Esteban et al., 2016).

Nguyen et al (Nguyen et al., 2017) proposed a novel Deep Learning pipeline for disease risk prediction dubbed Deepr, trained on a EMR dataset extracted from a large hospital chain in Australia. This pipeline is composed as follows: first, a tokenizing stage encodes a patient's EMR using ICD-9 codes for representing diagnoses and procedures pertaining to a particular visit, and special tokens to represent the time between visits. Sequences of tokens form sentences, and sentences end with a time-between-visits token. After this, a second stage creates a matrix EMR representation

by embedding the tokens into a lower dimensional representation (word embeddings) and stacking them together. This embedding is learned from the dataset and the authors compare initializing the embedding matrix randomly, and using Word2Vec (Mikolov, Chen, Corrado & Dean, 2013). The third stage is a convolutional layer, that uses p kernels with window size $2d + 1$. After convolution, the ReLU non-linearity is applied, with subsequent Max-Pooling (in a somewhat unconventional way, Max-Pooling is applied not to the elements inside the same receptive field, but to the elements in the same position through the receptive fields). Then the results of pooling are concatenated into a p-dimensional vector that serves as input to a softmax classifier. The architecture is trainable end to end, using Stochastic Gradient Descent and Backpropagation.

	3 months		6 months	
Method	W/o time	With time	W/o time	With time
BoW + LR	0.786	0.797	0.797	0.811
Deepr (rand init)	0.7910.797	0.806	0.814	
Deepr (word2vec)	0.795	0.800	0.809	0.819

Table 2.9: Deepr performance against Logistic Regression with a Bag of Words Representation (BoW + LR). Source (Nguyen et al., 2017).

Nguyen et al test their architecture against a Logistic Regression baseline which used a Bag of Words (BoW) representation, in a task that consisted in predicting unplanned readmissions inside three and six-month time windows after initial discharge. In all tasks, Deepr outperformed the baseline by a slight margin, measured by ROC-AUC. Results can be seen in Table 15. 2D visualizations constructed using t-SNE (van der Maaten & Hinton, 2008) show that the Deepr representation is able to separate patients according to their risk of unplanned readmission, in a much cleaner way than the BoW representation.

In (Choi, Bahadori, Song, Stewart & Sun, 2016) Choi et al propose the use of medical ontologies (i.e. ICD-9) formulated as Directed Acyclical Graphs (DAGs) to regularize deep learning models via an attention mechanism. In such DAGs, the

leaf nodes correspond to medical conditions that can be observed, and their ancestors correspond to categories that group related conditions together. Here, an RNN is trained using different datasets (MIMIC-III, Sutter Palo Alto Medical Foundation), to learn several tasks (heart failure prediction and prediction of medical conditions to be observed on the next doctor visit (Choi, Bahadori, Sun et al., 2016)). The visits are initially represented by a binary vector that encodes the presence or absence of the different conditions specified by the medical ontology.

In this work attention is used in a way different from the traditional one: instead of defining a dynamic probability distribution over the hidden activations of an encoder (RNN, CNN), attention is used to define a probability distribution over a specific observed medical condition (related to an ICD code) and its ancestors in the ontology, which in turn is used to compute an embedding vector for each possible observation. Since each node of the DAG is assigned an embedding vector e_i, the attention-generated embedding vector can be understood as the expected value of the embedding. This schema works as a regularizer that helps to generalize better when certain conditions are not observed often in the training data, by leveraging the auxiliary information that can be extracted from other, more often observed conditions, that belong to the same ontology subgroup. The set of all embedding vectors form an embedding matrix G which is used to compute a patient visit representation which is fed to a RNN. The results in (Choi, Bahadori, Song et al., 2016) show that in simulated data-scarce scenarios, GRAM regularization indeed helps the network generalize better, compared to a set of strong baselines.

2.2.2 Mortality prediction inside the ICU

Che et al (Che, Kale, Li, Bahadori & Liu, 2015) proposed a feed-forward deep model with sigmoidal activations to predict survival into the ICU, trained using data from

the PhysioNet Challenge 2012 (Silva et al., 2012). This model was also trained for disease diagnosis using data from a private clinical database. Their work included the usage of graph Laplacian priors (Weinberger, Sha, Zhu & Saul, 2007) to regularize the network and improve its performance in the absence of massive datasets, and to make use of structured domain knowledge. Another interesting contribution of (Che et al., 2015) was to develop an incremental training procedure to learn networks of increasing complexity, as a way to tackle the inherent hardness of training deep architectures.

The authors formulated the prediction task as a multilabel classification problem, where in the case of the PhysioNet Challenge dataset, 4 categories were considered: mortality, length of stay (LOS) less than 3 days, whether the patient was recovering from surgery, and whether the patient had a heart-related condition. The PhysioNet Challenge 2012 dataset contains data from 8000 ICU units in the form of multivariate time series. Each data point consists of approximately 48 hours of measurements of over 30 physiological variables. Che et al assume every time series has the same length and proceed to stack them, creating a matrix representation X for each patient. Subsequently the matrix is flattened into a vector x and fed into the network, and its corresponding set of labels is used for supervised training.

The conditional log-likelihood of the output y_i given the input x_i, parameterized by Θ; according to the model proposed by (Che et al., 2015) for the K-class multilabel classification task, can be stated as

$$logP(y_i|x_i, \Theta) = \sum_{k=1}^{K} [y_i k log\sigma(\beta_k^T h_i) + (1 - y_i k) log(1 - \sigma(\beta_k^T h_i))] \tag{2.7}$$

where σ is the logistic sigmoid function.

The use of Laplacian priors is interesting as it allows to encode prior knowledge expressed via graph representations. Given a certain matrix A that captures correlations between data (i.e. similarity, adjacency etc.), the information it encodes can be leveraged

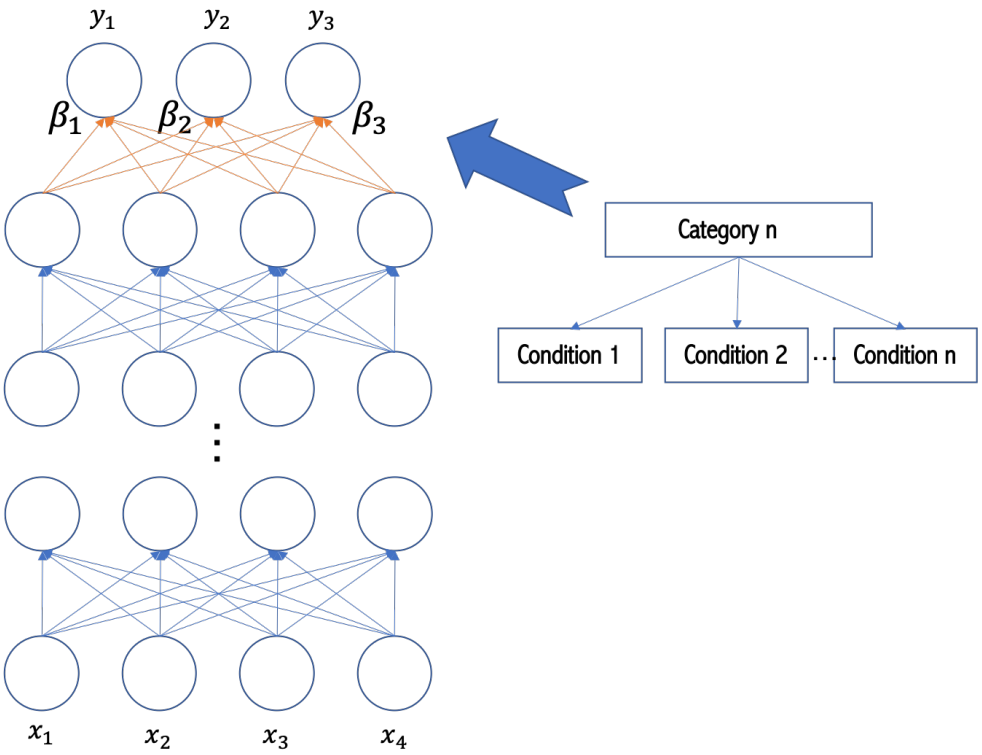


Figure 2.6: Network structure and the role of Laplacian regularization.

through a regularization term of the form $tr(\beta^T \boldsymbol{L}\beta)$, where $\boldsymbol{L}$ is the Laplacian matrix, defined as $\boldsymbol{L} = \boldsymbol{C} - \boldsymbol{A}$ and $\boldsymbol{C}$ is a diagonal matrix with its non-zero elements given by $\boldsymbol{C}(k,k') = \sum_{k'=1}^{K} A_{k,k'}$. β is a matrix with K columns corresponding to parameter vectors β_k whose components weight the un-normalized network predictions for each class. The regularization effect is more apparent considering that

$$tr(\beta^T \boldsymbol{L}\beta) = \frac{1}{2}\sum_{k=1}^{k}\sum_{k'=1}^{k} A_{k,k'}||\beta_k - \beta_{k'}||_2^2 \tag{2.8}$$

The regularization term forces the predictions for the different classes to be similar if the correlation between them is high, according to the prior knowledge. For the disease diagnosis prediction task (Che et al., 2015) use ICD-9 (international Classification of Diseases, version 9) hierarchical diagnostic codes to account for similarity between diseases, by constructing an adjacency matrix representing a graph with edges between

diagnoses corresponding to the same ICD-9 category. For the PhysioNet data, a co-occurrence matrix $A \in R^{K*K}$ was used to model priors after the empirical distribution of the classes. The elements of such matrix can be defined as

$$A_{k,k'} = \frac{1}{N} \sum_{i=1}^{N} 1_{y_{ik} * y_{ik'=1}} \tag{2.9}$$

where $1_{y_{ik} * y_{ik'=1}}$ is the indicator function, which outputs 1 when example i has labels corresponding to classes k and k'. The authors compare the performance of a deep feed-forward network with Laplacian Regularization, without regularization, and an independent baseline classifier. Results show that the deep feed-forward network model performed impressively for the surgery and cardiac labels even without Laplacian regularization, while the Laplacian regularized deep network improves substantially on both the baseline and the un-regularized network result for Mortality and LOS< 3 labels.

Kale et al (Kale et al., 2015) used Stacked Denoising Autoencoders (SDA) trained on the PhysioNet Challenge 2012 dataset (Silva et al., 2012) and on a dataset extracted from the EMR system of the Children's Hospital LA PICU, to predict ICU mortality and diagnose 17 disease groups (according to ICU-9 hierarchical codes), respectively; using an input representation similar to (Che et al., 2015). This is followed by the use of causal inference algorithms to quantify the predictive power of the features discovered by the SDAs. A Denoising Autoencoder (P. Vincent, Larochelle, Bengio & Manzagol, 2008) is a regular Autoencoder that uses multiplicative noise to corrupt the inputs, zeroing-out some of its components randomly as a regularization mechanism. The authors used a greedy layer-wise unsupervised training procedure to learn multiple representation features that were later used as part of a Deep Belief Net, essentially in the same fashion as (Lasko et al., 2013) for the PhysioNet data (but with an SVM on top); and used supervised fine-tuning of the features, minimizing the negative log-likelihood of the

whole (SDA layers plus a Logistic Regression classifier on top instead of an SVM) by gradient descent and using backpropagation, for the PICU data.

	AUROC	AUPRC	Precision@90% Recall
Raw Time Series (R)	0.786848 ± 0.028957	0.407419 ± 0.042878	0.221303 ± 0.017106
Hand-designed Features (H)	0.828652 ± 0.021065	0.467742 ± 0.047852	0.259324 ± 0.049400
NNet(R,3)	0.820760 ± 0.021021	0.444315 ± 0.032367	0.255792 ± 0.030306
H+R	0.822907 ± 0.018251	0.438160 ± 0.035444	0.255608 ± 0.031871
H+NNet(R,3)	0.845015 ± 0.016525	0.486791 ± 0.047373	0.291411 ± 0.033500

Table 2.10: Kale et al Performance on the PhysioNet Challenge 2012 dataset (Kale et al., 2015).

The main contribution of (Kale et al., 2015) was to show that is was possible to discover and interpret the causal direction between the input features and model output, and to quantify the causal power of such features. For the former task, Kale et al use the Pairwise LiNGAM algorithm (Hyvärinen & Smith, 2013) and for the latter they turn to fit an auxiliary Logistic Regression classifier using the features selected as most causally related on the first stage. Informally, they use the L2 norm of the coefficient vector for the features considered, to measure their total causal power (Kale et al., 2015).

10-fold Cross Validation results on the PhysioNet dataset show the SDA features achieving better results (Table 2.10) in terms of the Area Under the Receiver Operating Characteristic curve and Area under the Precision-Recall curve, than those obtained by using the raw data as input to the SVM. However, the features discovered by the SDA don't show any performance advantage over hand designed features that were included as benchmarks into the experiment. Despite this, when the learned and hand designed features are used in tandem, the SVM performance improves noticeably, which suggests that both feature sets are somewhat complementary. Also, the authors report that the causal power of the learned features was superior to the one of the hand designed features, which further suggest their validity and value.

Lipton et al (Lipton et al., 2015) use LSTM RNNs (Hochreiter & Schmidhuber, 1997b) for multilabel classification of diagnoses inside Children's Hospital Los Angeles

PICU, in what according to the authors is the first work that uses LSTM RNNs in a clinical prediction setting. In (Lipton et al., 2015), the inputs correspond to multivariate time-series of physiological data extracted from PICU patients, and the prediction task involves 128 non-mutually exclusive diagnostics. The multivariate time series involved measurements of 13 variables sampled irregularly, which led to missing data when resampling all series to the same (hourly) rate. To fill the gaps, forward and backward filling was performed (imputing the missing observation with the closest measurement in time). If a variable had no samples in a one-hour window, a normal value for it (according to the medical literature) was imputed. When multiple measurements were available in the same window, a mean measurement was taken. The LSTM was trained using a convex combination of the mean negative log-likelihood over all labels at each time-step (the authors call this, sequential target replication) and the mean negative log-likelihood over all labels at the final time-step. A two-hidden layer LSTM trained with L2 regularization and dropout outperformed two strong baselines (Logistic Regression, Multi-layer Perceptron). Although the prediction targets were 128 possible diagnoses, additional diagnoses were included as prediction targets in the training phase to achieve an additional regularization effect.

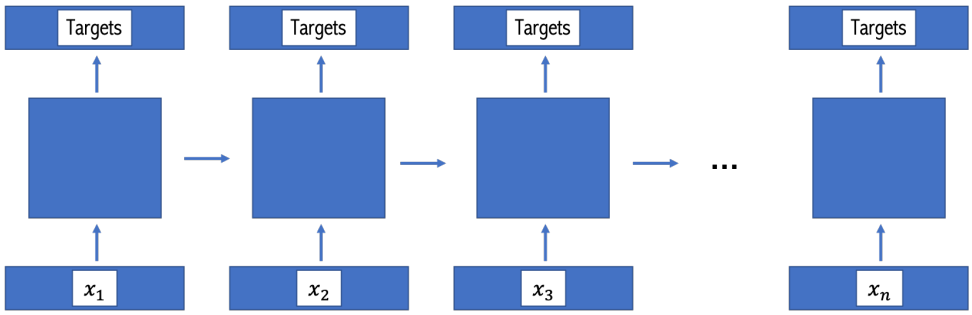


Figure 2.7: Sequential target replication. A loss is computed in each time-step and their average is pooled into a single loss via convex combination with the loss in the final time-step.

In follow-up work (Lipton et al., 2016), the authors treat the missing data problem from an interesting point of view. As similarly shown in (Razavian & Sontag, 2015), the patterns of testing in patients convey information in themselves. The authors explore the benefits of imputation against simple zero-filling, and modelling directly the missingness of data by concatenating a binary mask to the input at each time-step. The results show that the LSTM benefits more from zero-filling and the direct modelling of missingness than from imputed values, which is a surprising finding. Also, the performance of the LSTM RNN was superior to the other baselines considered.

0.4	0.5	0.8	0.2	0.5
0	1	0	0	1

Figure 2.8: Example of the modelling of missing data with imputation and a binary mask.

Che et al (Che, Purushotham, Cho et al., 2016) used a RNN based on a modified version of a Gated Recurrent Unit (GRU) to learn from multivariate time series with missing values (GRU-D). The authors used data from the MIMIC-III database (A. E. W. Johnson, Pollard et al., 2016) and the PhysioNet Challenge 2012 (Silva et al., 2012), as well as synthetic data to show the performance of their model. The authors note that in their experiments, the pattern of missingness is correlated with ICU mortality and hypothesize that it could be beneficial to model the predictions as a function of missing values. The authors also note that, one, the missing values for the variables tend to be close to some default value if the last observed value occurred a while ago, and two, the importance of a variable for the prediction diminishes, if its values have been not recorded for a while. The authors attribute these phenomena to homeostasis and temporal dependencies, however in our view it is possible that the missingness pattern instead reflects the opinion of physicians and their approach to tests and measurements: If the condition of a patient doesn't warrant the measurement of

certain variables, those variables will tend to be absent from record more often. Che et al propose an adaptive (with trainable parameters) imputation schema that model missing data as a function of the mean of the observations for the corresponding variable, with a decay term that depends on the time since the last observed value. The equation for the adaptive imputation schema is

$$x_t^d \leftarrow m_t^d x_t^d + (1 - m_t^d)\gamma_{x_t^d} x_{t'}^d + (1 - m_t^d)(1 - \gamma_{x_t^d})\tilde{x}^d \tag{2.10}$$

where x_t^d is the value of the variable d in time t, $\tilde{x}^d$ is the mean value for d, m_t^d is a binary mask that indicates if the observationx_t^d is missing or not (0 missing, 1 present), and γ is the adaptive decay rate, which in turn is given by

$$\gamma_t = exp(-max(0, W_\gamma \delta_t + b_\gamma)) \tag{2.11}$$

W_γ and b_γ are parameters jointly trained with the rest of the GRU parameters. Moreover, a similar decay rate is introduced to the network recurrence, and the GRU equations are altered to include the binary mask vectors for each time-step t. The network was trained to predict (among other tasks) ICU mortality, together with several baselines including RNNs with alternative imputation and masking schemas. The results for the prediction of mortality on MIMIC-III and PhysioNet datasets are shown in Table 2.11.

Grnarova et al (Grnarova et al., 2016) proposed an interesting application of Natural Language Processing (NLP) to mortality prediction in ICU patients. Their approach consisted in the use of a ConvNet trained on free-text clinical notes extracted from the MIMIC-III database (A. E. W. Johnson, Pollard et al., 2016). The ConvNet architecture has two convolutional layers that operate on the word level, and on the sentence level respectively. The input to the word level layer comes in the form of 50-dimensional Word2Vec embeddings. Word2Vec (Mikolov, Chen et al., 2013) is a framework for

Models	MIMIC-III	PhysioNet
Logistic Regression - forward filling	0.7589 ±0.015	0.7423 ±0.011
SVM - forward filling	0.7908 ±0.006	0.8131 ±0.018
Random Forest - forward filling	0.8293 ±0.004	0.8183 ±0.015
Logistic Regression - binary missingness mask	0.7715 ±0.015	0.7625 ±0.004
SVM - binary missingness mask	0.8146 ±0.008	0.8277 ±0.012
Random Forest - binary missingness	0.8294 ±0.007	0.8157 ±0.013
LSTM - mean imputation	0.8142 ±0.014	0.8025 ±0.013
GRU - mean imputation	0.8192 ±0.013	0.8195 ±0.004
GRU - forward filling	0.8252 ±0.011	0.8162 ±0.014
GRU - binary missingness mask	0.8380 ±0.008	0.8155 ±0.004
GRU-D	**0.8527 ±0.003**	**0.8424 ±0.012**

Table 2.11: Results (AUC score) for ICU mortality prediction. Best results are in bold. Source (Che, Purushotham, Cho et al., 2016).

efficient generation of vector spaces that allows to create vector representations for large text corpora. These embeddings are used as input for neural networks in many NLP applications (Lau & Baldwin, 2016). Typically, in such applications, text sentences are represented by matrices in which each column corresponds to a word vector. In (Grnarova et al., 2016) this representation s_i is used as input to the first layer, which use 50 kernels of dimension 50×5, 50 kernels of dimension 50×4, and 50 kernels of dimension 50×3 (the dimensions of each kernel can be read as $\langle$ embedding dimensions $\rangle \times \langle$ number of words $\rangle$). The convolution operation generates receptive fields size 1×5, 1×4, and 1×3; on which a non-linearity is applied element-wise. Max-Pooling is then performed with neighborhood size equal to the receptive field size. After these operations, a 150-dimensional sentence representation vector x_i is created.

The second layer receives the sentence representations, which concatenated constitute a patient representation. Additionally, each sentence representation is augmented with a 10-dimensional vector z_k that encodes the type of note from which the sentence came. The notes are grouped in categories, e.g. nursing and social work (Grnarova et al., 2016). This is done in order to give additional context to the classifier to judge the relative importance of sentences. The second convolutional layer use 50 kernels

with size 1×3, which are convolved over the concatenated sentence representations to produce 50 feature maps. A non-linearity is applied element-wise on the feature maps, followed by Max-Pooling as in the first layer. As a result, a 50-dimensional patient representation is created. The network architecture is completed by a fully connected layer and a Softmax classifier on top.

Task	LDA	doc2vec	ConvNet
Hospital	0.930	0.930	**0.963**
30-day	0.800	0.831	**0.858**
1-year	0.790	0.824	**0.853**

Table 2.12: ConvNet performance on the MIMIC-III database. Best results are in bold. Source (Grnarova et al., 2016).

One interesting feature of the architecture in (Grnarova et al., 2016), is that each sentence feature is connected directly to a Softmax for mortality prediction, in what the authors call Target Replication. The losses of these N Softmax classifiers are included on the total loss as an additional term. This is found to improve substantially the performance of the whole network (AUC goes from 0.682 to 0.858), as it improves gradient flow and regularizes the model by forcing it to learn sentence features that are predictive in themselves. In addition, the inclusion of individual sentences into the loss directly, allows for quantifying their individual contribution to the prediction, giving some interpretability to the model. The authors report results of several experiments, in which they attempted to predict mortality during the ICU stay, within 30 days after discharge, and within a year after discharge. Performance was compared to several popular baselines, and results showed that the ConvNet surpassed the baseline models in terms of the AUROC (Table 2.12).

In (Suresh et al., 2017), Suresh et al use deep networks that leverage demographic information, physiological time series data and free text clinical notes extracted from MIMIC-III to predict the onset and weaning of medical interventions (invasive ventilation, non-invasive ventilation, vasopressors, colloid boluses, and crystalloid boluses)

inside the ICU. Each patient is represented (similarly to (Esteban et al., 2016)) as a matrix that concatenates the physiological time series data, the static demographic info, and a Latent Dirichlet Allocation (LDA) (Blei, Ng & Jordan, 2003) topic distribution of the clinical notes; which is used as input for a deep learning model. The models use a 6-hour data window to predict the onset or weaning of a medical intervention inside a window of 4 hours, 6-hours into the future.

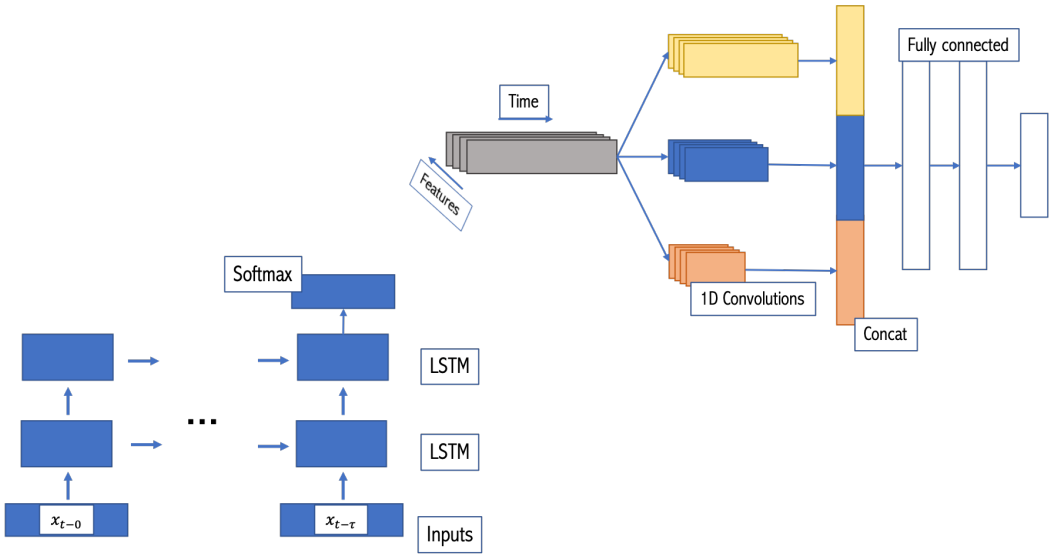


Figure 2.9: Deep models used for medical intervention prediction in (Suresh et al., 2017).

The models considered in this work are a ConvNet similar to the one presented in (Razavian et al., 2016), and a deep Long Short-Term Memory (LSTM) RNN (Hochreiter & Schmidhuber, 1997b). The best performer was a LSTM deep RNN that used physiological words (i.e. the time series were pre-processed and converted to z-scores). Also, the authors propose the use of occlusion-based sensitivity analysis (i.e. setting inputs to zero) (Zeiler & Fergus, 2014) to gain interpretability into the predictions of the LSTM, and perform gradient ascent to find inputs that maximize the probability of the class labels (Simonyan, Vedaldi & Zisserman, 2013) in the case of the ConvNets. In this way, they are able to introspect the models and derive interesting insights.

2.2.3 Interpretable mortality prediction inside the ICU

To the best of our knowledge at the time of this writing, the only work that explicitly tried to tackle the issue of offering interpretable mortality prediction in the ICU setting is (Che, Purushotham, Khemani & Liu, 2016). Che et al (Che, Purushotham, Khemani & Liu, 2016) propose the use of deep models as Deep feed-forward Networks (DNN) (Rumelhart et al., 1986) and the Gated Recurrent Unit (GRU) (Chung et al., 2014) to predict ICU outcomes (two different tasks, prediction of mortality and prediction of ventilator-free days >14). They used data from the Pediatric ICU dataset for acute lung injury (Khemani, Conti, Alonzo, Bart & Newth, 2009) to train and validate the performance of the aforementioned models. Data included static variables as demographic info and diagnosis annotations, and a set of 21 temporal variables, recorded daily. Results showed the deep models over-performing several baseline classifiers as Support Vector Machines (SVM), Logistic Regression (LR), Decision Trees (DT) and Gradient Boosted Trees (GBT) (Table 2.13). The DNN + GRU architecture proposed is similar to the one found in (Esteban et al., 2016).

Model	Mortality		Ventilator Free Days	
	AUROC	AUPRC	AUROC	AUPRC
SVM	0.6437 ± 0.024	0.3408 ± 0.034	0.7251 ± 0.023	0.7901 ± 0.019
LR	0.6915 ± 0.027	0.3736 ± 0.038	0.7592 ± 0.021	0.8142 ± 0.019
DT	0.6024 ± 0.013	0.4369 ± 0.016	0.5794 ± 0.022	0.7570 ± 0.012
GBT	0.7196 ± 0.023	0.4171 ± 0.040	0.7528 ± 0.017	0.8037 ± 0.018
DNN	0.7266 ± 0.089	0.4117 ± 0.122	0.7752 ± 0.054	0.8341 ± 0.042
GRU	0.7666 ± 0.063	0.4587 ± 0.104	0.7723 ± 0.053	0.8131 ± 0.058
DNN + GRU	0.7813 ± 0.028	0.4874 ± 0.051	0.7896 ± 0.019	0.8397 ± 0.018
Best Mimic Model	0.7898 ± 0.030	0.4766 ± 0.050	0.7889 ± 0.018	0.8324 ± 0.016

Table 2.13: 5-Fold Cross Validation results reported in (Che, Purushotham, Khemani & Liu, 2016). Best results for each task are in bold.

Regarding interpretability, the main point of (Che, Purushotham, Khemani & Liu, 2016) is the use of mimic learning to distill the knowledge gained by the deep models into a shallow, explainable model. Mimic learning (Ba & Caruana, 2014) is a learning

technique in which a master model (a deep neural network) is trained until convergence, and then its "knowledge" is transferred to a shallow net (the student model), by training the student model on the same data set as the master, but replacing the ground truth labels with the labels predicted by the master model. In this way, it is possible to achieve performances that would have been out of reach for the student network, if it were to be trained on the original data. Instead of using a shallow multilayer perceptron as the student, Che et al chose to use Gradient Boosted Trees (GBT) (Friedman, 2001) in what they call Interpretable Mimic Learning (Che, Purushotham, Khemani & Liu, 2016). GBT is a model part of the Gradient Boosting Machines framework (Friedman, 2001), in which an ensemble of (weak) learners is built in a greedy fashion, to approximate a complex classification/regression hyper-surface. At each stage a new learner is trained to fit the negative gradient of the loss and then added to the ensemble. It can be shown that this negative gradient corresponds to the residual between the current ensemble prediction and the ground truth. At inference time, the predictions of the members of the ensemble are summed and a final prediction is generated. The model at stage M can be expressed as

$$F_M(X) = \sum_{i=1}^{M} \upsilon \gamma_i h_i(X) \tag{2.12}$$

where υ corresponds to a shrinkage coefficient (regularizer) that is multiplied to the current model in each stage before adding a new ensemble member $h_i(X)$. γ_i is a hyper-parameter found by line search to minimize the ensemble loss. In the case of GBT, the individual members of the ensemble are Decision Trees. Results show that the distilled GBT performs better than a GBT trained on the ground truth directly. This suggests that at least some of the knowledge gained by the original model was indeed transferred to the shallow one.

In order to interpret the decisions of the student model, a number of approaches

are considered, including Feature Influence (Friedman, 2001), and comparing the top decision rules used by the most prominent tree in the ensemble.

2.2.4 Recent relevant developments

The results by Grnarova et al have inspired further research into the use of medical notes for mortality prediction using deep learning. For example, Jo et al (Combining LSTM and Latent Topic Modeling for Mortality PredictionJo, Lee & Palaskar, 2017) used a hybrid Latent Dirichlet Allocation (LDA) + Long Short Term Memory (LSTM) model for ICU mortality prediction trained on medical notes from MIMIC-III, in which the LSTM used the topic LDA features as input. The authors divided patients stays into 12-hour segments and grouped all clinical notes in each of them, to attempt the prediction of mortality at the end of each segment. Multiple recurrent networks were trained and compared to an LDA baseline, obtaining interesting results. Suchil et al (Sushil, Šuster, Luyckx & Daelemans, 2018) used stacked denoising autoencoders to create patient representations out of medical free-text notes, to be used for downstream tasks as mortality prediction, in a way similar to (Miotto et al., 2016). The evaluation framework and reported results are similar to those by Grnarova et al. Si et al (Si & Roberts, 2019) proposed the use of a ConvNet for multitask prediction (mortality, length of stay), using all available patient medical notes up until time of discharge. The architecture presented in this work is a so-called multi-level (word level and sentence level) convolutional architecture, reminiscent of the one used by Grnarova et al for a similar purpose. Finally, the regularizing effects of multitask learning are studied and it was found that training the network to jointly perform several tasks (in-hospital, 30-day, and 1-year mortality prediction) using a shared patient representation provides some performance gains. Jin et al (Jin et al., 2018) proposed a multimodal neural network architecture and a Named Entity Recognition (NER) text pre-processing pipeline to

predict in-hospital ICU mortality using all available types of free-text notes and a set of vital signs and lab results from the first 48 hours of patient stay, extracted from MIMIC-III.

On the deep learning-based mortality prediction using physiological time series front, Ge et al (Ge et al., 2018) propose the use of multiple RNNs (LSTM) to encode physiological time series into vector representations to be fed to a logistic regression classifier, together with some static features, for the prediction of mortality. The authors provide a way to interpret the model by analyzing the weights of the logistic classifier, finding clinically valid correlations with predictors such as Do Not Resuscitate (DNR) status, heart failure, and the use of certain medications, among others.

Finally, Purushotham et al (Purushotham et al., 2018) carried out a comprehensive benchmark of several machine learning and deep learning models trained on MIMIC-III for various tasks, with results showing deep models consistently outperforming the rest. A hybrid model combining a recurrent architecture to handle temporal data, and a feedforward architecture for static data obtained the best result, slightly outperforming the rest (ROC AUC 0.8783 ± 0.0037) at predicting mortality inside the ICU.

2.3 Conclusion

We have reviewed the relevant literature to provide the context in which our work is framed. The use of Deep Learning in medicine for clinical event prediction is a nascent field, which has attracted some interest in the last years. Although there have been interesting and remarkable results, several challenges as providing model interpretability and adequate performance in larger, diverse populations remain. We expect to see continued research in these directions in the short term.

Chapter 3

Interpretable Deep Learning architectures for mortality prediction

3.1 Introduction

In this chapter we will introduce the characteristics of the dataset used, and the inclusion criteria for this study. Next we will review the theoretical basis behind our classification models and describe their architecture. Finally we will board the issue of feature importance attribution from the perspective of game theory and briefly describe the DeepLIFT algorithm.

3.2 Participants

We used the Medical Information Mart for Intensive Care III (MIMIC-III v1.4) to train our deep models. MIMIC III is a database comprised of more than a decade worth of ICU patient records, including vital signs, laboratory reports, radiology reports, and therapeutic data, from patients admitted to the Beth Israel Deaconess Medical Centre in Boston, Massachusetts, freely available for research (A. E. W. Johnson, Pollard et

al., 2016). The median age of adult patients (age > 16 years old) is 65.8 years and the median length of stay (LOS) in the ICU is 2.1 days (Q1-Q3: 1.2-4.6) (A. E. W. Johnson, Pollard et al., 2016).

To establish a cohort and build our dataset, several entry criteria were defined: we only considered stays longer than 48 hours, only patients older than 16 years old at the time of admission were included, and in case of multiple admissions to the ICU, only the first one was considered (this is to preserve the independence assumption of the dataset instances). Application of these entry criteria led to a dataset containing 22.413 distinct patients, which correspond to 58.06 % of the total number of patients in the database. Median Length of Stay (LoS) was 3.9 days (Q1-Q3: 2.7-7.1). Table 3.1 and Figure 3.3 show some dataset statistics. Figure 3.1 shows a plot of our LoS distribution.

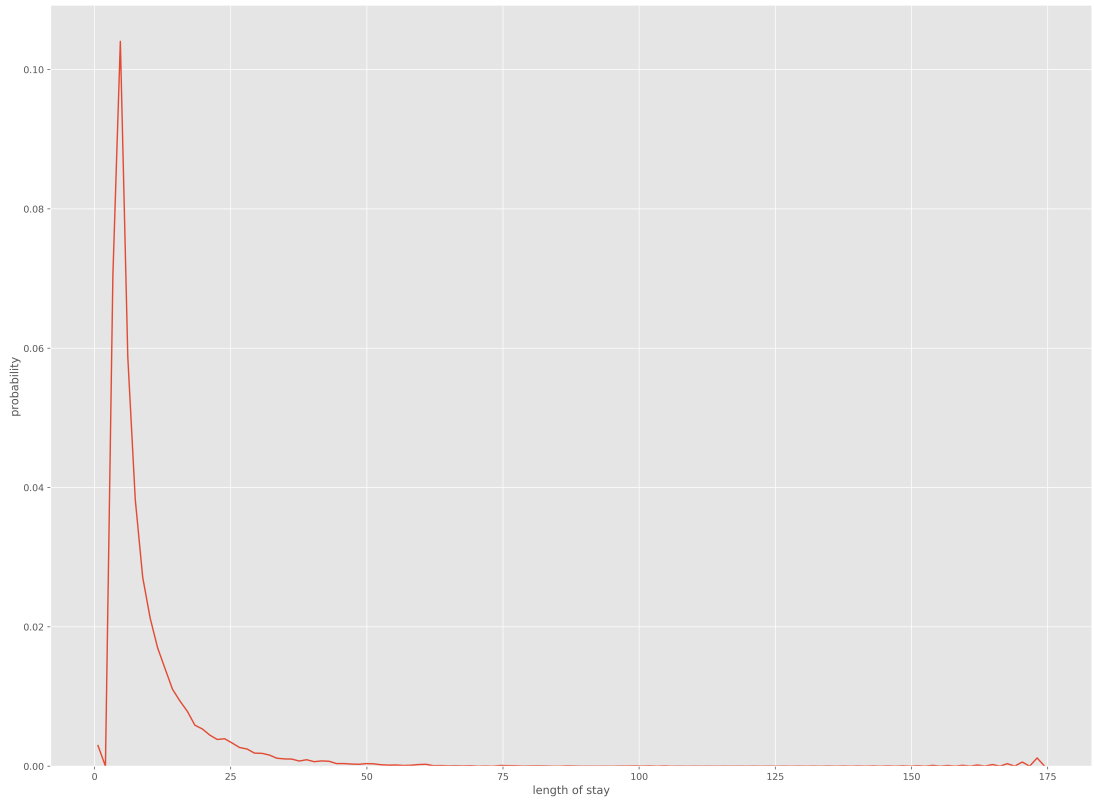


Figure 3.1: Kernel Density Estimation (KDE)-generated LoS distribution.

Feature	n	Mean	Std.	Min	Q1	Q2	Q3	Max	Percent of total pop.
Temporal									
Bicarbonate	99668	23.296	4.733	5.000	20.000	23.000	26.000	52.000	N/A
Bilirrubin	24765	3.098	6.170	0.100	0.500	1.000	2.900	82.000	N/A
BUN	101133	27.497	22.493	1.000	13.000	20.000	34.000	240.000	N/A
Diastolic BP	1752075	59.648	14.090	1.000	50.000	58.000	68.000	298.000	N/A
FiO2	294740	50.100	20.030	0.400	40.000	50.000	50.000	100.000	N/A
GCSEyes	442646	3.140	1.142	1.000	3.000	4.000	4.000	4.000	N/A
GCSMotor	440486	5.256	1.440	1.000	5.000	6.000	6.000	6.000	N/A
GCSVerbal	441269	3.123	1.902	1.000	1.000	4.000	5.000	5.000	N/A
Heart rate	1755492	87.913	18.951	0.350	75.000	86.000	99.000	280.000	N/A
PO2	160600	149.763	95.724	14.000	87.000	119.000	177.000	763.000	N/A
Potassium	176528	4.192	0.700	0.600	3.700	4.100	4.500	26.500	N/A
Sodium	125440	138.294	5.350	1.210	135.000	138.000	141.000	183.000	N/A
Systolic BP	1755083	118.518	22.973	0.150	102.000	116.000	133.000	323.000	N/A
Temperature	563035	37.007	0.8610	15.000	36.444	37.000	37.600	42.222	N/A
Urine output	947826	113.900	162.357	-4000.000	37.000	70.000	140.000	4800.000	N/A
WBC	94209	12.743	11.377	0.100	8.000	11.200	15.200	528.000	N/A
Static									
Age	22413	63.828	15.576	16.016	53.998	67.1016	78.533	80.000	N/A
Elective admission	3618	N/A	N/A	N/A	N/A	N/A	N/A	N/A	14.134%
Surgical admission	8030	N/A	N/A	N/A	N/A	N/A	N/A	N/A	35.827%
AIDS	113	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.504%
Metastatic cancer	688	N/A	N/A	N/A	N/A	N/A	N/A	N/A	3.069%
Lymphoma	317	N/A	N/A	N/A	N/A	N/A	N/A	N/A	1.414%
Mortality	2185	N/A	N/A	N/A	N/A	N/A	N/A	N/A	9.748%

Table 3.1: Some dataset statistics. FiO_2 and PO_2 stand for fraction of inspired oxygen and oxygen pressure in blood, respectively. BUN stands for Blood Urea Level, and WBC stands for White cell Blood Count.

3.3 Physiological and demographic features

For each patient, we extracted measurements of 22 different concepts/variables during the first 48 hours of each patient stay. These concepts roughly match the concepts used by the Simplified Acute Physiology Score II (SAPS-II), used to predict ICU mortality (Gall et al., 1993). The differences between SAPS-II concepts and our input variable set are summarized next:

- *Type of admission.* SAPS-II considers a concept/variable called type of admission, which has three possible values, i.e. “scheduled surgical", “unscheduled surgical", and “medical"; and as such is considered a categorial variable. In order for our models to work correctly we de-composed type of admission into two indicator variables, surgical admission and elective admission, whose combination models the original concept. According to SAPS-II original paper, surgeries are considered scheduled when patients are added to the operating room schedule at

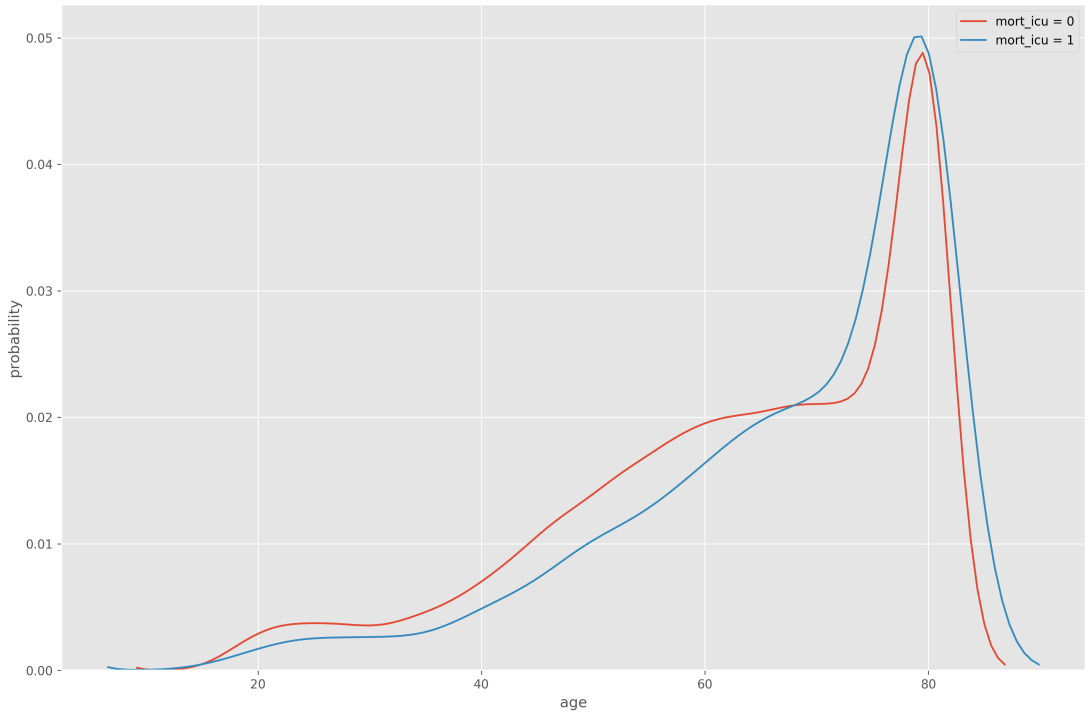


Figure 3.2: KDE-generated age distribution, grouped by mortality. The large bump close to 80 years of age reflects our pre-processing.

least 24 hours in advance, while those surgeries scheduled within 24 hours of the operation are considered unscheduled. Medical admissions refer to patients with no surgery performed on them within 1 week of admission to the ICU. In MIMIC-III for each admission a flag in the ADMISSIONS table (ADMISSION_TYPE) indicates whether hospital admissions were planned in advance or not, and we use this flag in conjunction with the type of service the patient was admitted to (table SERVICES).

- *Chronic disease.* Similarly, SAPS-II includes the concept of chronic disease, which is another categorical variable that takes the values of “AIDS", “metastatic cancer", and “hematological malignancy". Again, we decomposed it into three indicator variables for our models to be able to use this concept correctly. In MIMIC-III the respective ICD-9 codes are available in the DIAGNOSES_ICD

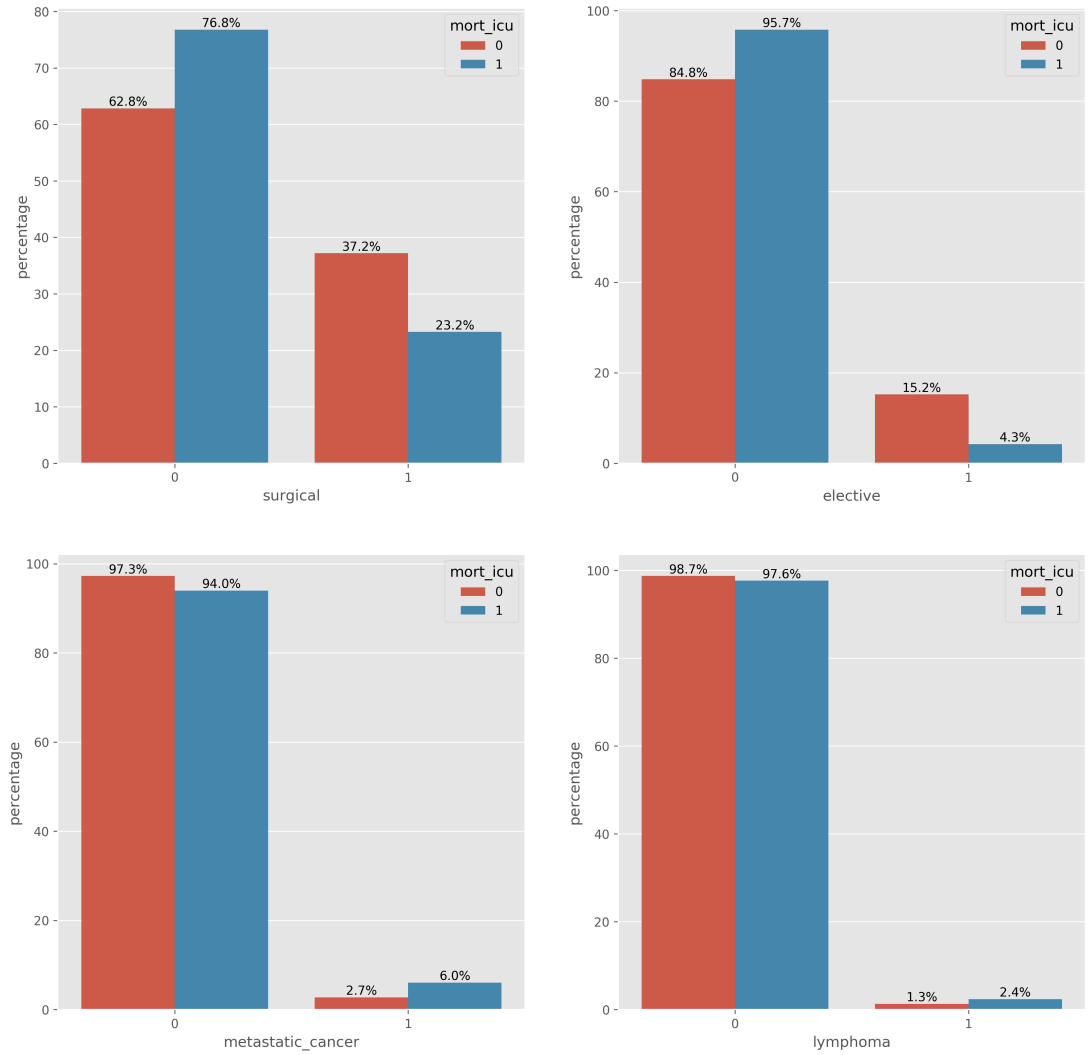


Figure 3.3: Distribution of some static variables, grouped by patient outcome (0 represents survival, 1 represents death).

table.

- *Glasgow Coma Scale.* The Glasgow Coma Scale (GCS) is comprised of three components, i.e. eyes, motor and verbal. In SAPS-II all components are aggregated into a single number. We opted for using the three components of the scale as individual inputs.

- PO_2/FiO_2. SAPS-II uses the PO_2/FiO_2 quotient as input if the patient is either ventilated (VENT) or receiving Continuous Positive Airway Pressure (CPAP).

We decided to include PO_2 and FiO_2 as separate input variables, with no regard to whether the patient is under VENT or CPAP.

- *Diastolic blood pressure* We added the diastolic blood pressure variable to our input variable set in order to have a dataset more comparable to some reference works in the relevant literature (Purushotham et al., 2018).

Table 3.4 shows the correspondence between the SAPS-II input variable set and the variables considered by our model.
In the case of temporal data, all measurements were extracted and in case of multiple measurement in the same hour, values were averaged (except urine output which was summed). Sampling distribution statistics for are shown in Table 3.2 and Figure 3.4.

	measurement interval							
	n	Mean	Std.	Min	Q1	Q2	Q3	Max
Feature								
Bicarbonate	55273	12.551	7.424	0.017	6.450	10.800	18.600	45.500
Bilirubin	8998	14.154	8.842	0.033	6.600	11.600	22.667	47.450
BUN	56317	12.441	7.312	0.017	6.500	10.767	17.633	44.917
Diastolic BP	1254101	0.810	0.579	0.017	0.500	1.000	1.000	41.833
FiO_2	217055	2.203	2.286	0.017	0.567	2.000	3.667	46.750
Glasgow coma scale (eyes)	322848	2.986	1.853	0.017	1.500	3.000	4.000	41.000
Glasgow coma scale (motor)	321212	2.996	1.866	0.017	1.500	3.000	4.000	41.000
Glasgow coma scale (verbal)	321810	2.992	1.872	0.017	1.500	3.000	4.000	41.000
Heart rate	1335396	0.765	0.524	0.017	0.333	1.000	1.000	37.500
PO_2	104726	4.124	4.621	0.017	1.350	2.633	5.133	47.317
Potassium	111342	6.918	6.325	0.017	2.200	5.167	9.517	45.600
Sodium	74106	9.859	7.567	0.017	4.033	7.883	13.567	45.600
Systolic BP	1254613	0.810	0.579	0.017	0.500	1.000	1.000	41.833
Temperature	403546	2.349	1.972	0.017	1.000	2.000	4.000	46.850
Urine output	711945	1.334	1.029	0.017	1.000	1.000	1.000	44.367
White cell blood count	50268	13.239	8.245	0.017	6.100	11.367	22.500	46.317

Table 3.2: Distribution of (inter) sampling times in hours for temporal features. Zero valued intervals corresponding to duplicate measurements have been omitted. FiO_2 and PO_2 stand for fraction of inspired oxygen and oxygen pressure in blood, respectively.

To resolve inconsistencies and harmonize sometimes seemingly disparate concepts (i.e. temperature is reported both in Celsius and Fahrenheit, different codes for the

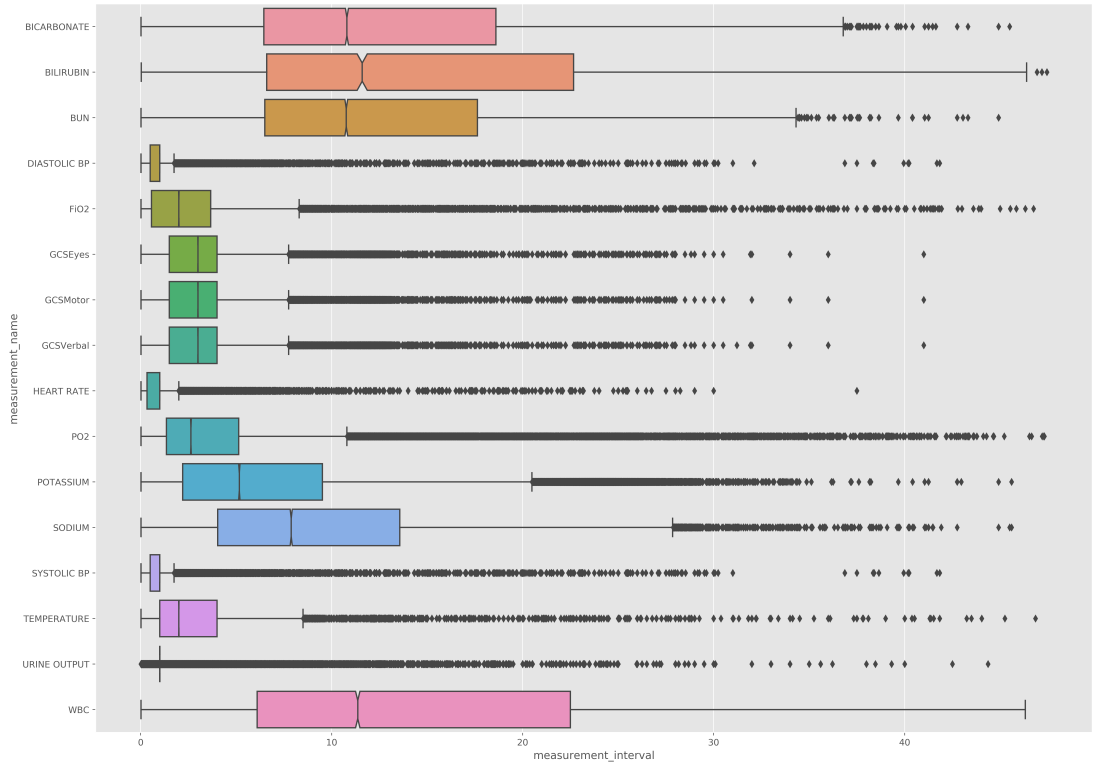


Figure 3.4: Boxplot of (inter) sampling times in hours for temporal features. Zero valued intervals corresponding to duplicate measurements have been omitted. GCSEyes, GCSMotor, and GCSVerbal stand for Glasgow coma scale (eyes), Glasgow coma scale (motor), and Glasgow coma scale (verbal) respectively. WBC stands for white cell blood count. FiO_2 and PO_2 stand for fraction of inspired oxygen and oxygen pressure in blood, respectively.

same measurement are used, same or related concepts are present in different tables, etc), data was pre-processed, measurements merged and renamed, and un-physiological values were discarded (See Table 3.3 for details of discarded values). For privacy reasons, MIMIC-III shifts ages greater than 89 years (i.e. patients appear to be 300 years old). To address this, we decided to use the SAPS-II criteria for patient age which assigns equal weight to all ages over 80 years and clipped all ages greater than 80 years to 80 years (see Figure 3.2). For reproducibility, all of our code is available at `https://github.com/williamcaicedo/ISeeU`.

It is understood that there is a trade-off between having more data (and more data

Feature	Number of discarded values	Exclusion criteria
Blood potassium (mEq/L)	1	value < 0 OR value > 30
Blood sodium (mEq/L)	0	value < 0 OR value > 200
Diastolic blood pressure (mm Hg)	74	value < 0 OR value ≥ 300
FiO_2 (%)	869	value < 21 OR value > 100
Glasgow coma scale (eyes)	77616	value < 1 OR value > 6
Glasgow coma scale (motor)	35380	value < 1 OR value > 6
Glasgow coma scale (verbal)	188591	value < 1 OR value > 5
Heart rate (bpm)	2	value < 0 OR value ≥ 300
PO_2 (mm Hg)	1	value < 0 OR value > 800
Systolic blood pressure (mm Hg)	6	value < 0 OR value ≥ 400
Temperature (C °)	512	value < 10 OR value > 50
Urine output (mL)	31	value > 5000
White cell blood count (k/uL)	0	value < 0 OR value ≥ 1000

Table 3.3: Discarded values during preprocessing. FiO_2 and PO_2 stand for fraction of inspired oxygen and oxygen pressure in blood, respectively.

Feature	SAPS-II mapped variable	MIMIC-III table
Age	Age	ICUSTAYS, PATIENTS
Presence of AIDS	Chronic disease	DIAGNOSES_ICD
Blood bicarbonate	Blood bicarbonate	LABEVENTS
Blood bilirubin	Blood bilirubin	LABEVENTS
Blood Urea Nitrogen (BUN)	BUN	LABEVENTS
Diastolic blood pressure	N/A	CHARTEVENTS
Systolic blood pressure	Systolic blood pressure	CHARTEVENTS
Temperature	Temperature ≥ 39 C°	CHARTEVENTS
Admission to the ICU after surgery	Type of admission	SERVICES
Elective admission to the ICU	Type of admission	ADMISSIONS
FiO_2	PO_2/FiO_2	CHARTEVENTS, LABEVENTS
Glasgow coma scale (eyes)	GCS	CHARTEVENTS
Glasgow coma scale (motor)	GCS	CHARTEVENTS
Glasgow coma scale (verbal)	GCS	CHARTEVENTS
Heart rate	Heart rate	CHARTEVENTS
Presence of lymphoma	Chronic disease	DIAGNOSES_ICD
Presence of metastatic cancer	Chronic disease	DIAGNOSES_ICD
PO_2	PO_2/FiO_2	LABEVENTS
Blood potassium	Blood potassium	LABEVENTS
Blood sodium	Blood sodium	LABEVENTS
Urine output	Urine output	OUTPUTEVENTS
White cell blood count	White cell count	LABEVENTS

Table 3.4: Features extracted from MIMIC-III for each patient, and their mappings to SAPS-II. FiO_2 and PO_2 stand for fraction of inspired oxygen and oxygen pressure in blood, respectively.

could lead to more accurate predictions) and making earlier and possibly more useful predictions. However, since the advent of the PhysioNet Challenge (Silva et al., 2012)

in which data from MIMIC-II was used to predict mortality, the use of a 48 hour patient representation has been a popular choice and different works in the relevant literature have shown its adequacy (Calvert et al., 2016; Purushotham et al., 2018; Che, Purushotham, Cho et al., 2016).

Missing data Due to the nature of patient monitoring, different physiological variables and features are sampled at different rates. This lead to large number of missing observations, as not all measurements were available in an hourly fashion. Given this situation, simple data imputation techniques were applied to obtain a $22x48$ dense observation matrix for each patient (static features like age and admission where replicated). Concretely, except for Glasgow Comma Scale (GCS) and urine observations, forward/backward filling imputation was attempted. Missing GCS and FiO2 values were imputed to their normal values. On the other hand, when multiple observations were present in the same hour, values were averaged. In cases where a patient did not have a single observation recorded, we imputed the whole physiological time series using mean values. Our data imputation procedure is summarized in table 3.5.

Feature	% of missing values	Imputation procedure
Bicarbonate	92.79%	Forward/Backward filling, mean value imputation
Bilirubin	98.23%	Forward/Backward filling, mean value imputation
BUN	92.69%	Forward/Backward filling, mean value imputation
Diastolic BP	10.07%	Forward/Backward filling, mean value imputation
FiO2	83.04%	Forward and backward filling imputation, normal value (0.2) imputation
GCSEyes	68.30%	Normal value (4) was imputed
GCSMotor	68.45%	Normal value (6) was imputed
GCSVerbal	68.39%	Normal value (5) was imputed
Heart rate	7.53%	Forward/Backward filling, mean value imputation
PO2	89.35%	Forward filling, mean value imputation
Potassium	88.21%	Forward/Backward filling, mean value imputation
Sodium	91.27%	Forward/Backward filling, mean value imputation
Systolic BP	10.05%	Forward/Backward filling, mean value imputation
Temperature	66.15%	Forward/Backward filling, mean value imputation
Urine output	33.15%	Zero value imputation
WBC	93.27%	Forward/Backward filling, mean value imputation

Table 3.5: Imputation procedure summary.

3.4 Free-text features

MIMIC-III contains free-text medical notes as part of patient data, comprising nursing reports, radiology reports, medical evolution reports and physician observations, among others; which contain a large amount of data that could be used for predictive purposes. We identified all patients in our cohort with at least one medical note associated to their stay leading to a sub-sample of n = 21415 patients. Table 3.6 shows the different types of medical notes included in our dataset together with their respective counts.

Note Type	**Count**	Percentage
Nursing/other	83147	36.78%
Radiology	61096	27.02%
Nursing	43790	19.37%
Physician	27789	12.30%
Respiratory	5728	2.53%
General	1775	0.78%
Nutrition	1549	0.68%
Rehab Services	521	0.23%
Social Work	501	0.22%
Case Management	134	0.060%
Consult	40	0.018%
Pharmacy	12	0.0053%
Overall	226082	100%

Table 3.6: Distribution of free-text medical notes in our dataset.

The most prevalent note types are nursing/other, radiology, and nursing. Combined, all nursing-related note types represent 56.14% of the total (126937 notes). Given their prevalence and our assumption that nursing notes are more readily available than other note types, for the remainder of this study we will restrict ourselves to nursing-related notes, reducing our patient sub-sample to n = 16970, with 1659 recorded deaths (9.78%) and 15311 patients who survived (90.22%). The mean note length is 1252.59 words, with a standard deviation of 1087.48. Table 3.7 and figure 3.5 show details about the distribution of notes' lengths. The median age of patients with at least one nursing-related

note is 67.2 years, and the median length of stay is 3.96 days (Q1-Q3:2.8-7.16). Figures 3.6 and 3.7 show the distribution of age and length of stay in our note sub-sample.

Statistic	Positive class (survival)	Negative class (death)	Overall
Count	15311	1659	16970
Mean	1233.3	1430.6	1252.6
Std	1083	1112.4	1087.5
Min	34	144	34
Q1	711.0	890	724
Q2	934	1135	952
Q3	1286	1492	1310
Max	33771	9756	33771

Table 3.7: Length distribution of nursing notes.

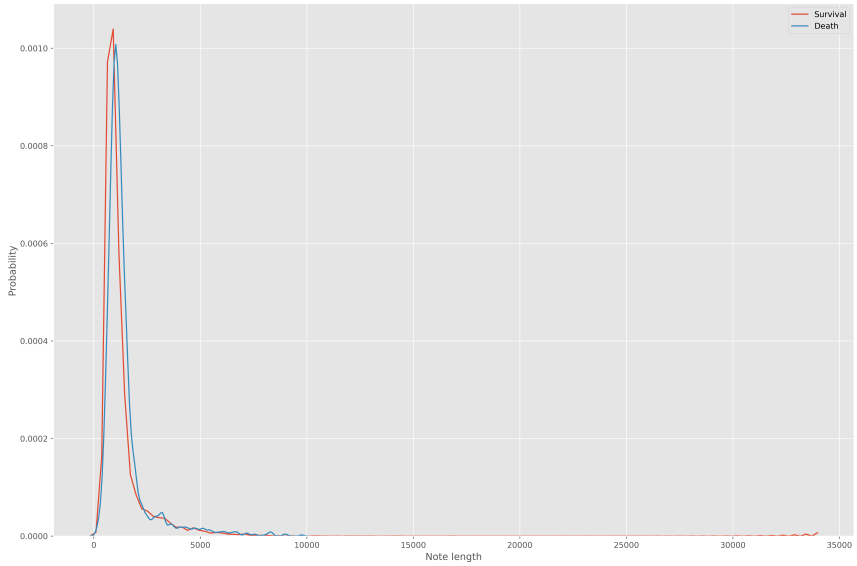


Figure 3.5: Estimated nursing notes length distribution.

Figure 3.6: Histogram of age distribution by outcome. As a result of privacy preserving measures, MIMIC-III shifts ages greater than 89 years (i.e. patients appear to be 300 years old).

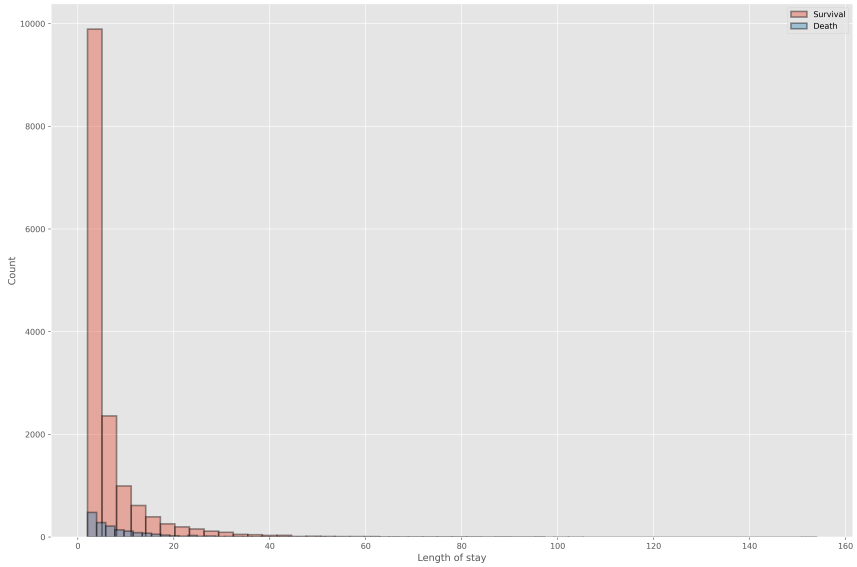


Figure 3.7: Histogram of length of stay distribution by outcome.

3.5 Deep Learning models

Our prediction models are Deep Convolutional Neural Networks (ConvNets). ConvNets are Multi-Layer Neural Networks that use a particular architecture with sparse connections and parameter sharing (LeCun et al., 1998). They are specialized in handling grid-like data as temporal series (1-D grids), images (2-D grids), video (3-D), among others. Their name references the convolution, a linear operator ConvNets use in-lieu of traditional matrix multiplication in at least one layer.

The convolution takes two functions as arguments, performs the product of the two functions as one "slides" over the other and then integrate over the resulting products. As a motivating example, let's consider the problem of smoothing out a noisy signal x coming from some sort of sensor. To do so, it would be convenient to average the signal intensity over contiguous time steps and try to recover the original readings, by multiplying the original signal at each time step by a Gaussian function w whose area integrates to one and its output is zero for negative arguments (i.e. a valid probability function) and then averaging the result. This can be expressed as

$$s(t) = \int_{t_0}^{t_s} x(a)w(t-a)da \tag{3.1}$$

In the above, the Gaussian function is first reversed $w(-a)$ and then shifted by an amount t. The integration over products returns a locally smoothed version s of the original x, as the Gaussian transforms the values from the original function into linear combinations with terms weighted according to its spread. This operation we just performed corresponds to the convolution $s(t) = (x * w)(t)$ (given that we make $t_0 = -\inf$ and $t_s = \inf$).

If we restrict ourselves to functions only defined for $a \in \mathbb{Z}$, the discrete version of the convolution would arise. In this setting, ConvNets can be thought of performing a discrete convolution operation between the input (often a two-dimensional image) and

a set of trainable "kernels" (sliding functions) at each layer. The discrete convolution [1] operation, in the context of Deep Learning and computer vision is defined as

$$s(i,j) = (x * w)(i,j) = \sum_{m,n} I(m,n)K(i-m, j-n) \tag{3.2}$$

where I is a two-dimensional image and K is a two-dimensional kernel. The kernel acts as a local feature detector that is displaced all over the image and its value is only non-zero in a small region (sparsity). Each convolution between the input and a kernel produces a spatial receptive field, also called a feature map, in which each kernel-image multiplication can be thought of as pattern matching, producing an output that is a function of the similarity between certain image region and the kernel itself. After the convolution operation, the output of the receptive field is ran through a non-linear activation function which allows the network to work with transforms of the input space and construct non-linear features. The feature map can be thought as a 2-D tensor (matrix) of neuron outputs, where the weights of each neuron are the same but have been shifted spatially (hence the parameter sharing), and that are not connected to every single pixel of the input (which also can be seen as having the corresponding weight set to zero). ConvNets were one the first models to use Gradient Descent with Backpropagation (Rumelhart et al., 1986) with success (Goodfellow et al., 2016). Convolution based filters are extensively used to detect features as shapes and edges in computer vision (Shapiro, 2001). However, in traditional computer vision fixed kernels are used to detect specific features, in contrast to ConvNets where kernels are learned directly from the data.

Convolutional layers offer advantages over feed-forward, fully connected layers used in multilayer perceptrons. One of these advantages is translational equivariance,

[1] In software implementations it is the discrete cross-correlation operation the one that is implemented, since the sliding function is not reversed. In practical terms reversing it does not offer any noticeable performance advantage.

which refers to the ability of the convolutional layer to shift its response in a way that is proportional to the shifts in its input. For instance, if an image contains a face and the position of the face changes, the output of a convolutional layer will change spatially in the same way.

In ConvNets, the convolution plus non-linear activation operation is often followed by another transformation called pooling. Pooling is a subsampling operation that replaces the output of a convolutional layer for some spatial statistic like the maximum (max-pooling) or average (average-pooling). The introduction of pooling helps to keep the total number of network parameters in check by reducing the dimensionality of the layer output, and further helps to attain invariance with respect to some transformations of the input (Goodfellow et al., 2016). The architecture of ConvNets can be seen as able to encode an infinitely strong prior that biases the model for local information (use of convolution with sparse kernels) and to be invariant with respect to small translations (pooling). Such priors are not easily derived by standard penalized maximum likelihood or Bayesian inference (Goodfellow et al., 2016).

3.6 Shapley Values and input relevance attribution

The Shapley Value (Shapley, 1953) is a concept from game theory that formalizes the individual contribution of a player part of a coalition to the attainment of a reward in a game (Strumbelj, Kononenko & Wrobel, 2010). Shapley Values are the expectation of such contribution over the set of all possible permutations and values of the player coalition, taking into consideration all possible interactions between players. Formally, for a coalitional form game $\langle N, v \rangle$, where N is a finite set of players and $v : 2^N \to \mathbb{R}$ describes the worth of a player coalition, we have that

$$Sh_i(v) = \sum_{S \subseteq N \setminus \{i\}, s=|S|} \frac{(n-s-1)!s!}{n!}(v(S \cup \{i\}) - v(S)) \tag{3.3}$$

where Sh_i is the individual contribution of player i to the total worth $v(N)$, i.e. its Shapley value (Shapley, 1953). The summation runs over all possible subsets of players $S \subseteq N$ that don't include player i, and each term involves the difference between the reward when player i is present and absent, $v(S \cup \{i\}) - v(S)$.

This particular definition of player contributions satisfies certain assumptions of fairness in the distribution of the game reward among its members, which are represented by the following set of axioms (Shapley, 1953):

Axiom 1 The sum of individual player contributions (represented hereafter by ϕ_i) must be equal to the total worth of the coalition, i.e. $\sum_{i \in N} \phi_i(v) = v(N)$. This is the *efficiency axiom*.

Axiom 2 Given two distinct players i and j, if $v(S \cup \{i\}) = v(S \cup \{j\})$ for every subset of players S so that $S \subseteq N$, then $\phi_i(v) = \phi_j(v)$. This is the *symmetry axiom*.

Axiom 3 If $v(S \cup \{i\}) = v(S)$ for every subset of players S so that $S \subseteq N$ and $i \notin S$, then $\phi_i(v) = 0$. This is the *dummy axiom*.

Axiom 4 For any pair of games v, w it must hold that $\phi_i(v+w) = \phi_i(v) + \phi_i(w)$, given that $(v+w)(S) = v(S) + w(S)$ for every coalition S. This is the *additivity axiom*.

Equation 3.3 not only considers the presence of a particular player, but also the position it occupies in the coalition. This is extremely well-suited to the context of our study, in which input values are time/order sensitive.

As an illustrative example, consider the case of a coalition comprised of at most, by two players, p_1 and p_2. If only p_1 participates then the worth of the coalition is

$v(\{p_1\}) = 7$, whereas if only p_2 participates, the worth of the coalition would be $v(\{p_2\}) = 10$; and finally if both players are present $v(\{p_1, p_2\}) = 21$ (notice the non-linear effects on the coalition worth mediated by the interaction of the players). Also consider that it is possible for the players to arrive to the coalition in different order, which affects its relative contribution to the total worth of the coalition: if p_1 comes first and then p_2, the contribution of p_1 would be the aforementioned 7, and the contribution of p_2 would be $v(\{p_1, p_2\}) - v(\{p_1\}) = 21 - 7 = 14$; while on the other hand, if p_2 arrives first its contribution would be 10 while the contribution of p_1 would be $v(\{p_1, p_2\}) - v(\{p_1\}) = 21 - 10 = 11$. Furthermore, assume that both orderings have the same probability of occurrence.

Given this configuration, we have that the expected contribution of p_1 to the coalition is $\frac{1}{2}7 + \frac{1}{2}11 = 9$, and the expected contribution of p_2 is $\frac{1}{2}14 + \frac{1}{2}10 = 12$. Table 3.8 summarizes the results of the calculations.

Probability	Ordering	Player 1 contribution	Player 2 contribution
0.5	Player 1, Player 2	7	14
0.5	Player 2, Player 1	11	10
Expected contribution (Shapley Value)		**9**	**12**

Table 3.8: Shapley values for example coalition players.

Strumbelj et al (Strumbelj et al., 2010) showed that such values can be used to represent the relevance of each input to a machine learning classifier in order to gain insight on the patterns it considers important to predict a particular class, and proposed a feature importance attribution method equivalent to calculating the Shapley Values for a coalition of players (inputs) that work together to attain a reward (prediction). The use of Shapley values for importance attribution is particularly advantageous since they are able to take into account the possible interactions between input features in a way occlusion-based methods (Zeiler & Fergus, 2014) cannot.

To this end, they propose the concept of prediction difference $\Delta(S)$ (Strumbelj et

al., 2010) to quantity the influence a subset of features S has on the output of a machine learning classifier. Formally, if only features in S are known, the prediction difference associated to S is defined as

$$\Delta(S) = \frac{1}{|\mathcal{A}_{N \setminus S}|} \sum_{y \in \mathcal{A}_{N \setminus S}} f_c(\tau(x, y, S)) - \frac{1}{|\mathcal{A}|} \sum_{y \in \mathcal{A}_N} f_c(y) \tag{3.4}$$

where $\mathcal{A}_N$ represents a feature space of dimension N, f_c is the component of the output of a machine learning classifier that corresponds to predicted probability of class c, and $\tau(x, y, S)$ is a masking function defined as

$$\tau(x, y, S) = (z_1, z_2, \ldots, z_n), \qquad z_i = \begin{cases} x_i & i \in S \\ y_i & i \notin S \end{cases} \tag{3.5}$$

Examination of equation 3.5 shows that on the first term the features in S are held constant while the rest take each one of their possible values. This corresponds to the prediction the classifier makes, marginalized over the all the features not in S. On the other hand, the baseline term $\frac{1}{|\mathcal{A}|} \sum_{y \in \mathcal{A}_N} f_c(y)$ corresponds to the expected prediction of the classifier, when no features are known. The marginalization over a subset of features is a classifier-agnostic way to simulate their absence from the feature space, allowing to compute the contribution of subset S to the prediction.

Given the possible nonlinear nature of f_c, methods that rely on feature independence assumptions to attribute importance can yield erroneous results. Consider for example, a binary OR function. If examined separately, each input carries zero importance as the OR value does not change if one input changes, revealing the need to account for feature interactions. To address this problem and manage interactions properly, Strumbelj et al explicitly define $\Delta(S)$ as a sum of feature interactions

$$\Delta(S) = \sum_{W \subseteq S} I(W) \tag{3.6}$$

Notice that all the 2^N possible subsets of S are considered. In the above, $I(W)$ is a function that quantifies how much each interaction between subsets of S contributes to prediction difference $\Delta(S)$. If we assume that $I(\varnothing) = 0$, we can manipulate equation 3.6 to obtain a recursive definition of I for any feature subset

$$I(S) = \Delta(S) - \sum_{W \subset S} I(W) \tag{3.7}$$

Now that the contributions of interactions are formalized, we can define how much every feature i contributes to prediction difference Δ as a function of the interaction it is part of

$$\varphi_i(\Delta) = \sum_{W \subseteq N \setminus \{i\}} \frac{I(W \cup \{i\})}{|W \cup \{i\}|}, \quad i = 1, 2, \ldots, n \tag{3.8}$$

So far we have followed (Strumbelj et al., 2010) going from the formalization of the concept of prediction difference of an input feature set/subset $\Delta(S)$, to its alternative formulation as a sum of feature interactions $I(W)$, and finally to an expression that quantifies the contribution of each individual feature to $\Delta(S)$, $\varphi_i(\Delta)$. At this point (Strumbelj et al., 2010) introduce the crux of their argument: they show that given the characteristics of Δ, the problem of importance attribution can be understood as a coalitional game $\langle N, \Delta \rangle$ where the players N are the input features and Δ maps a set of input features to an importance value. In this setting $\varphi(\Delta)$ corresponds to the Shapley Values of the game (equation 3.3). A formal proof can be found in (Strumbelj et al., 2010).

3.6.1 DeepLIFT

The computations involved in equations 3.3 and 3.8 have combinatorial complexity, making them unfeasible for several practical applications, reason why we must resort to

approximations. In this context, we will discuss a new importance attribution method, called DeepLIFT. DeepLIFT (Shrikumar, Greenside & Kundaje, 2017) is an importance attribution method for feed forward neural networks, that is akin to the Layer-wise Relevance Propagation method (LRP) proposed by (Bach et al., 2015), in the sense that both use a backpropagation-like approach to the calculation and attribution of relevance/importance scores to the input features. DeepLIFT overcomes problems associated with gradient-based attribution methods (Simonyan et al., 2013; Springenberg, Dosovitskiy, Brox & Riedmiller, 2014) as saturation, overlooking negative contributions and contributions when the associated gradient is zero, and discontinuities in the gradients (Shrikumar et al., 2017). Since the attribution output of LRP was later shown to be roughly equivalent to a factor of a gradient method's output (Kindermans, Schütt, Müller & Dähne, 2016), it follows that LRP suffers from similar problems to those outlined before.

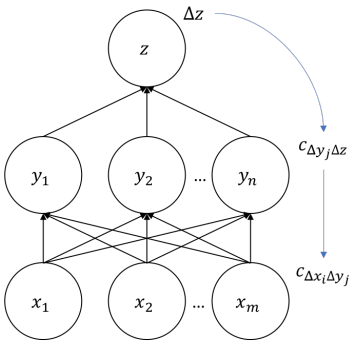


Figure 3.8: DeepLIFT's multipliers and chain rule allows the propagation of feature importances.

To compute feature importance the following procedure is carried out: first a reference input value must be provided. This reference value can be informed by domain knowledge or simply be the empirical mean of the input features, and once the references have been defined the corresponding network output is computed for both the original input and the reference input. Then the outputs' difference is backpropagated

through the network layers using rules provided by DeepLIFT.

More formally, for a target neuron t and a collection of neurons $x_1, x_2, ..., x_n$ whose outputs are needed to compute the output of t, the method assigns importance attributions $C_{\Delta x_i \Delta t}$ subject to the fact that such attributions are additive and must satisfy

$$\sum_{i=1}^{n} C_{\Delta x_i \Delta t} = \Delta t \tag{3.9}$$

where $\Delta t = t_o - t_r$ is the difference between the original and reference outputs of t. DeepLIFT introduces multipliers $m_{\Delta x_i \Delta t} = \frac{c_{\Delta x_i \Delta t}}{\Delta x}$ that allow to use a chain-rule to backpropagate the neuron attributions through a hidden layer. The rule takes the form

$$m_{\Delta x_i \Delta z} = \sum_j m_{\Delta x_i \Delta y_j} m_{\Delta y_j \Delta z} \tag{3.10}$$

where $m_{\Delta x_i \Delta z}$ is the contribution of neuron x_i to the output of neuron z divided by the difference in outputs for neuron x_i, Δx_i, given a hidden layer of neurons y_j in-between (see figure 3.8). The corresponding contribution $c_{\Delta x_i \Delta z}$ can be recovered from equation 3.10 as $c_{\Delta x_i \Delta z} = m_{\Delta x_i \Delta z} \Delta x$.

For a linear unit, the contribution of the inputs x_i to the output difference Δy is simply = $w_i \Delta x_i$. To avoid the issues of other methods regarding negative contributions, DeepLIFT treats separately positive and negative contributions, which leads to Δy and x_i being decomposed into its positive and negative components

$$\Delta y^+ = \sum_i 1\{w_i \Delta x_i > 0\} w_i (\Delta x_i^+ + \Delta x_i^-) \tag{3.11}$$

$$\Delta y^- = \sum_i 1\{w_i \Delta x_i < 0\} w_i (\Delta x_i^+ + \Delta x_i^-) \tag{3.12}$$

The contributions can be stated then as

$$c_{\Delta x_i^+ \Delta y^+} = \sum_i 1\{w_i \Delta x_i > 0\} w_i \Delta x_i^+ \tag{3.13}$$

$$c_{\Delta x_i^- \Delta y^+} = \sum_i 1\{w_i \Delta x_i > 0\} w_i \Delta x_i^- \tag{3.14}$$

$$c_{\Delta x_i^+ \Delta y^-} = \sum_i 1\{w_i \Delta x_i < 0\} w_i \Delta x_i^+ \tag{3.15}$$

$$c_{\Delta x_i^- \Delta y^-} = \sum_i 1\{w_i \Delta x_i < 0\} w_i \Delta x_i^- \tag{3.16}$$

For non-linear operations with a single input (e.g. ReLU activations), DeepLIFT proposes the so-called RevealCancel rule, which is able to better uncover non-linear dynamics (Shrikumar et al., 2017). For this case, Δy decomposes as

$$\Delta y^+ = \frac{1}{2}(f(x_0 + \Delta x^+) - f(x_0)) + \frac{1}{2}(f(x_0 + \Delta x^- + \Delta x^+) - f(x_0 + \Delta x^-)) \tag{3.17}$$

$$\Delta y^- = \frac{1}{2}(f(x_0 + \Delta x^-) - f(x_0)) + \frac{1}{2}(f(x_0 + \Delta x^+ + \Delta x^-) - f(x_0 + \Delta x^+)) \tag{3.18}$$

And to satisfy 3.9 we have that $\Delta y^+ = c_{\Delta x_i^+ y^+}$ and $\Delta y^- = c_{\Delta x_i^- y^-}$. Given this, the multipliers for the RevealCancel rule are

$$m_{\Delta x^+ y^+} = \frac{c_{\Delta x_i^+ y^+}}{\Delta y^+} = \frac{\Delta y^+}{\Delta y^+} \tag{3.19}$$

$$m_{\Delta x^- y^-} = \frac{c_{\Delta x_i^- y^-}}{\Delta x^-} = \frac{\Delta y^-}{\Delta y^-} \tag{3.20}$$

What makes DeepLIFT especially relevant is that it has been shown by Lundberg et al (Lundberg & Lee, 2017a) that DeepLIFT can be understood as fast approximation to the real Shapley Values when the feature reference values are set to their expected values. It can be seen that the RevealCancel rule computes the Shapley Values of the positive and negative contributions at the non-linear operations, and the successive application

of the chain rule proposed by DeepLIFT allows to propagate the approximate Shapley Values back to the inputs.

3.7 Conclusion

We have described our dataset, built from MIMIC-III data. We also stated an introduction to Convolutional Neural Networks, the kind of model we have used in our research. Finally we described Shapley Values and the algorithm we used to approximate them, DeepLIFT. In the next chapter we will describe our experimental settings and the characteristics of our Deep Learning architectures, along with our results.

Chapter 4

Experimental results and analysis

4.1 Introduction

As previously mentioned, our research deals with the use of ConvNets for ICU mortality prediction using physiological time series data and free-text medical note data as inputs. In this chapter we will give the details about model architecture, hyper parameter selection, and training/validation results.

4.2 Model development

We built our models using Keras (Chollet, 2015) and Tensorflow (USENIX Association. et al., 2015). Keras provides an easy to use high-level API that allows rapid prototyping. At the same time, it is possible to access the Tensorflow lower-level APIs for increased flexibility, which together with its model serialization and de-serialization capabilities made it a perfect fit for our research goals.

4.2.1 Physiological time series model

Development of models in machine learning tends to be an iterative process. It is quite common to set up test runs in which a reduced dataset (e.g. using fewer training examples or fewer input variables) is used for debugging purposes. Reason being that, as multiple libraries and are tools involved, it is likely that errors are introduced in the process of coding a full Machine Learning data pipeline. In this case it was decided to use a preexisting SQL query as a starting point for the entire pipeline. We modified this query, available at the MIMIC-III companion repository [1] to generate a dataset that included only common blood labs and routine measurements (see table 4.1) as input variables, using the same inclusion criteria defined from the beginning of this research. This dataset was put together initially as a proof of concept and to investigate the predictive power of common labs tests with little to no pre-processing other than standardization ($\mu = 0, \sigma \approx 1$). Finally all missing values were imputed to zero (equivalent to imputing the mean) instead of using more sophisticated imputation approaches.

Our chosen input representation places each feature as an image channel instead of stacking them as a 2-D input (the latter is natural for ConvNets in Computer Vision), similar to (Suresh et al., 2017). This allows us to use 1-D temporal convolutions no matter how many input series we use.

A common trend in deep learning when it comes to search for an appropriate architecture, is to start with an over-parameterized model (i.e. larger than necessary) and reduce the number of parameters and/or use regularization, guided by cross-validation (Goodfellow et al., 2016). Following suit we started with a deep model comprised by four convolutional and three fully connected layers (Figure 4.1), for a total of 10M parameters. This model was a standard ConvNet with dropout and batch normalization,

[1] `https://doi.org/10.5281/zenodo.821872`.

Feature	MIMIC-III Table
Albumin	LABEVENTS
Anion gap	LABEVENTS
Bicarbonate	LABEVENTS
Bilirubin	LABEVENTS
Blood Urea Nitrogen (BUN)	LABEVENTS
Chloride	LABEVENTS
Creatinine	LABEVENTS
Glucose	LABEVENTS
Hematocrit	LABEVENTS
Hemoglobin	LABEVENTS
International Normalized Ratio (INR)	LABEVENTS
Lactate	LABEVENTS
Magnesium	LABEVENTS
Oxigen saturation (SPO_2)	LABEVENTS
Phosphate	LABEVENTS
Platelet	LABEVENTS
Potassium	LABEVENTS
Prothrombin Time (PT)	LABEVENTS
Partial Thromboplastin Time (PTT)	LABEVENTS
Sodium	LABEVENTS
White cell blood count	LABEVENTS
Diastolic blood pressure	CHARTEEVENTS
Heart rate	CHARTEVENTS
Respiratory rate	CHARTEVENTS
Systolic blood pressure	CHARTEEVENTS
Temperature	CHARTEVENTS
Urine output	OUTPUTEVENTS

Table 4.1: Variables used in the initial, test dataset.

with no multi-scale design. To quickly get results with fewer parameters to adjust, we decided to use the Adam optimizer (Kingma & Ba, 2014). Since our dataset is highly unbalanced with the positive class (death) representing just under 10% of training examples, we used a weighted logarithmic loss that gives more importance to positive examples. Performance evaluation was carried out by calculating the area under the Receiver Operating Characteristic curve (ROC AUC), a standard metric commonly used in the related literature (Lipton et al., 2015; Che, Purushotham, Khemani & Liu, 2016; Purushotham et al., 2018). ROC AUC results showed that the model was overfitting

the training set (0.99 ROC AUC), which was the expected behaviour. On a hold-out validation set the model performed quite badly (0.587 ROC AUC), which also was the expected result.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 1, 48, 28)	0
conv1 (Conv2D)	(None, 1, 48, 512)	100864
bn1 (BatchNormalization)	(None, 1, 48, 512)	2048
activation_1 (Activation)	(None, 1, 48, 512)	0
max_pool1 (MaxPooling2D)	(None, 1, 24, 512)	0
conv2 (Conv2D)	(None, 1, 24, 512)	786944
bn2 (BatchNormalization)	(None, 1, 24, 512)	2048
activation_2 (Activation)	(None, 1, 24, 512)	0
max_pool2 (MaxPooling2D)	(None, 1, 12, 512)	0
conv3 (Conv2D)	(None, 1, 12, 1024)	1573888
bn3 (BatchNormalization)	(None, 1, 12, 1024)	4096
activation_3 (Activation)	(None, 1, 12, 1024)	0
max_pool3 (MaxPooling2D)	(None, 1, 6, 1024)	0
conv4 (Conv2D)	(None, 1, 6, 2048)	6293504
bn4 (BatchNormalization)	(None, 1, 6, 2048)	8192
activation_4 (Activation)	(None, 1, 6, 2048)	0
max_pool4 (MaxPooling2D)	(None, 1, 3, 2048)	0
flatten_1 (Flatten)	(None, 6144)	0
fc1 (Dense)	(None, 256)	1573120
fc2 (Dense)	(None, 512)	131584
fc3 (Dense)	(None, 1)	513

Total params: 10,476,801
Trainable params: 10,468,609
Non-trainable params: 8,192

Figure 4.1: Initial test model architecture as given by Keras. The large number of trainable parameters explain the observed overfitting.

Hyperparameter search space

After this initial run we started searching for a more suitable (and smaller) architecture. Together with network size we defined a hyper-parameter search space that included training batch size, dropout probability, class weights, and number of training epochs. This can be justified as follows:

- Training batch size. In deep learning, most commonly used optimization algorithms are all based on mini-batch stochastic gradient descent (Goodfellow et al., 2016). The main idea behind it is to introduce randomness into the gradient signal after forward propagation, by using only a subset of the training set for its calculation. In practice it has been found that this help to speed up convergence (Goodfellow et al., 2016). When selecting the batch size there is a trade-off: smaller batches introduce more noise and as a result the chance of getting stuck in bad quality local minima is lower, at the expense of longer training time. On the other hand a large batch size speeds up convergence but the benefits of a noisy gradient estimate are largely lost. For this parameter we considered a few popular values in the literature (16, 32, 64).

- Dropout probability (Srivastava et al., 2014). Dropout is an effective way to combat overfitting specially in large, overparameterized models (Goodfellow et al., 2016). It involves zeroing out neural connections randomly during the training process and compensating for it at inference time. Dropout can be understood as a way of encouraging the model to be more robust against noise and/or as a cheap way to create an exponentially large ensemble of models. We considered several values but in practice we tended to stay close to the originally recommended value of 0.5.

- Class weights. Setting different misclassification costs for our cost function was the way to manage our heavily unbalanced dataset. In practice this became much more of a constant, as after the initial test runs it was apparent that a 1:10 ratio was more than adequate to get the models to converge. This was heuristically determined after the fact that the minority class makes about 10% of our dataset.

- Number of training epochs. Setting this parameter involves a tradeoff insofar as if its value is too low, underfitting the dataset is a possibility, whereas choosing

a large enough value almost invariably means to overfit. We experimented with values from 30 to 100 epochs.

With our search space defined this way and using our initial model and setup, subsequent experimental runs recorded a 0.873 ROC AUC over the training set and a 0.606 ROC AUC over the validation set, but it was clear that hyperparameter search alone was not going to be able to improve our results significantly.

Multi-scale architectures

Our input representation is comprised by several physiological time series, some of them very different from the rest, and the corresponding time scales for events of interest are consequently different as well. On the other hand, in ConvNets the convolution kernels act as local feature detectors for which size determines the scale in which patterns contained in the input features are most relevant. In light of this it is sensible to posit that by using different kernel sizes at the same layer, the network would be able to pick up different patterns on different time scales of relevance (similar rationale has been applied before in related domains with some success (Razavian & Sontag, 2015; Suresh et al., 2017)).

After switching to a smaller model with a multi-scale architecture, performance improved considerably. At this point our best model was a multi-scale ConvNet using convolution kernels with different sizes that concatenated the resulting feature maps into a single layer output tensor (Figure 4.3). This is followed by ReLU activations, average-pooling with a window size of 3 and dropout (Srivastava et al., 2014) plus Batch Normalization (Ioffe & Szegedy, 2015), all of this performed after the concatenation operation. In this layer we use three temporal scales: Three hours, six hours, and twelve hours; each represented by a stack of convolution kernels with dimensions 3x1, 6x1, and 12x1, respectively. This model has 115.601 trainable parameters, two orders of

magnitude fewer parameters than our original model.

The convolutional layer is followed by a fully connected layer with ReLU activations, Dropout, Batch Normalization and a final one-neuron layer with logistic activation.

We decided to switch to 5-fold cross validation for a more reliable performance estimate, and to be able to report a measure of confidence of said performance. Also we standardized the dataset ($\mu = 0, \sigma \approx 1$) calculating fold statistics independently to avoid data leakage. We opted for heuristically chosen values for the most important hyper-parameters (dropout probability 0.5, and a batch size of 32). We also switched our optimizer to Stochastic Gradient Descent with Nesterov Momentum (0.9) (Nesterov, 1983), and learning rate of 0.01 with a $1e-7$ decay. We ran 50 training epochs for each fold, and picked the best model from each fold to calculate our ensemble performance metrics. After training, this multi-scale architecture was able to reach a ROC AUC of 0.8070 (± 0.0179) for the training set, and 0.7845 (± 0.0151) ROC AUC for the cross validation set.

SAPS-II dataset Our motivation for choosing our main dataset was two-fold: The variables largely coincide with those used by the SAPS-II risk score (Gall et al., 1993), which give us a better starting point with a set of predictors that have shown good correlation with our endpoint of interest. Furthermore, related works use SAPS-II inputs for the training and validation of deep models, making any comparison between models easier. Guided by this we created our new SAPS-II based dataset and decided to train a multi-scale architecture on it. Also, it is worth recalling that we did not perform any fancy imputation procedures and instead restricted ourselves to basic forward/backward filling, mean imputation, and in the case of the Urine variable, zero imputation.

Initial results immediately validated one of our assumptions, namely the fact that the SAPS-II input set is much more predictive of mortality than our initial blood panel, as it

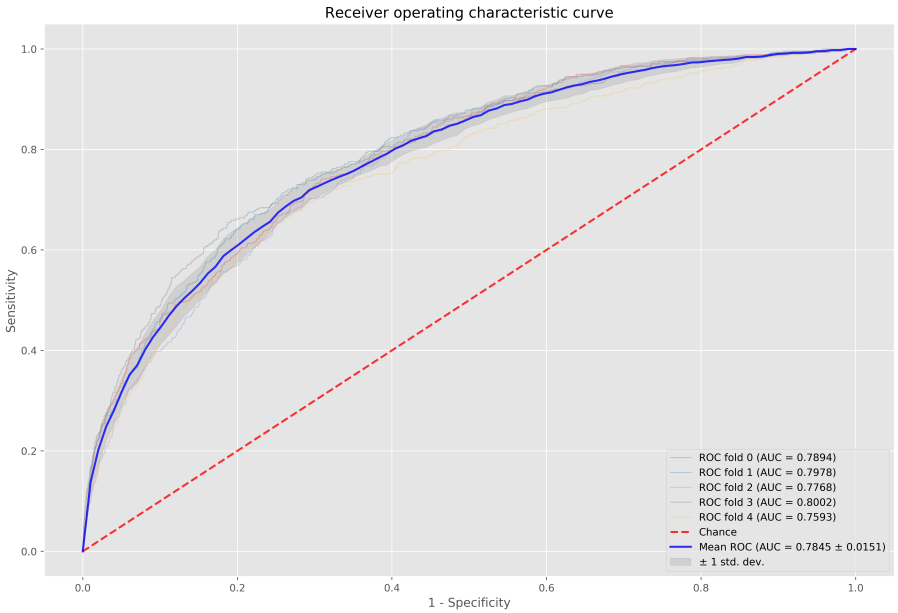


Figure 4.2: 27-predictor original model 5-fold cross validated ROC AUC.

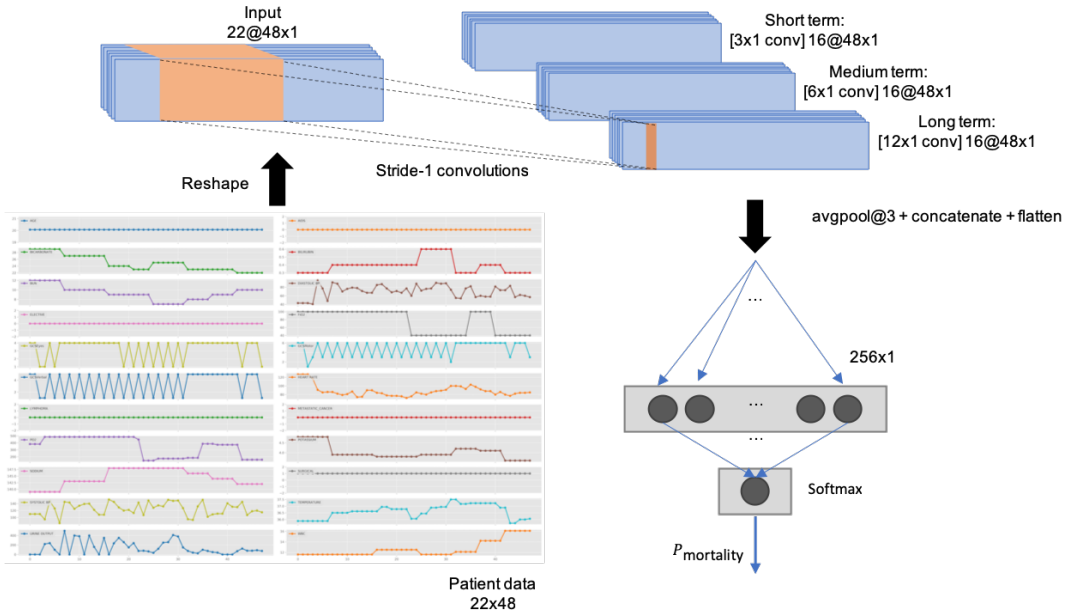


Figure 4.3: Multi-scale convolutional architecture for mortality prediction. ReLU activations, BatchNorm, and Dropout layers, have been omitted for clarity and brevity.

```
____________________________________________________________________________________________________
Layer (type)                     Output Shape          Param #     Connected to
====================================================================================================
input (InputLayer)               (None, 1, 22, 48)     0
____________________________________________________________________________________________________
3h_conv (Conv2D)                 (None, 1, 22, 16)     2320        input[0][0]
____________________________________________________________________________________________________
6h_conv (Conv2D)                 (None, 1, 22, 16)     4624        input[0][0]
____________________________________________________________________________________________________
12h_conv (Conv2D)                (None, 1, 22, 16)     9232        input[0][0]
____________________________________________________________________________________________________
activation_13 (Activation)       (None, 1, 22, 16)     0           3h_conv[0][0]
____________________________________________________________________________________________________
activation_14 (Activation)       (None, 1, 22, 16)     0           6h_conv[0][0]
____________________________________________________________________________________________________
activation_15 (Activation)       (None, 1, 22, 16)     0           12h_conv[0][0]
____________________________________________________________________________________________________
3h_pooled (AveragePooling2D)     (None, 1, 8, 16)      0           activation_13[0][0]
____________________________________________________________________________________________________
6h_pooled (AveragePooling2D)     (None, 1, 8, 16)      0           activation_14[0][0]
____________________________________________________________________________________________________
12h_pooled (AveragePooling2D)    (None, 1, 8, 16)      0           activation_15[0][0]
____________________________________________________________________________________________________
concatenate_5 (Concatenate)      (None, 1, 8, 48)      0           3h_pooled[0][0]
                                                                   6h_pooled[0][0]
                                                                   12h_pooled[0][0]
____________________________________________________________________________________________________
dropout1 (Dropout)               (None, 1, 8, 48)      0           concatenate_5[0][0]
____________________________________________________________________________________________________
bn1 (BatchNormalization)         (None, 1, 8, 48)      192         dropout1[0][0]
____________________________________________________________________________________________________
flatten_5 (Flatten)              (None, 384)           0           bn1[0][0]
____________________________________________________________________________________________________
fc1 (Dense)                      (None, 256)           98560       flatten_5[0][0]
____________________________________________________________________________________________________
dropout3 (Dropout)               (None, 256)           0           fc1[0][0]
____________________________________________________________________________________________________
bn3 (BatchNormalization)         (None, 256)           1024        dropout3[0][0]
____________________________________________________________________________________________________
fc2 (Dense)                      (None, 1)             257         bn3[0][0]
====================================================================================================
Total params: 116,209
Trainable params: 115,601
Non-trainable params: 608
____________________________________________________________________________________________________
```

Figure 4.4: SAPS-II trained multi-scale model parameters and architecture summary as given by Keras.

represents a great deal of knowledge acquired through continued research and medical practice. Even though our dataset variables are not exactly the ones used in SAPS-II (there are hand-engineered features in SAPS-II, while our approach is based on raw data) its general spirit is kept as most SAPS-II inputs are indeed present. After carrying our training and optimization procedures, our final model hyper-parameters did not deviate much from the ones used in our previous experiment (dropout probability 0.45, and a batch size of 32), and we also used Stochastic Gradient Descent with Nesterov Momentum (0.9), and learning rate of 0.01 with a $1e-7$ decay. Just as before, we ran 50 training epochs for each fold, and picked the best model from each fold to calculate our ensemble performance metrics. The model contains 115.601 trainable parameters

(Figure 4.4), and results show validation figures tracking closely the training ones.

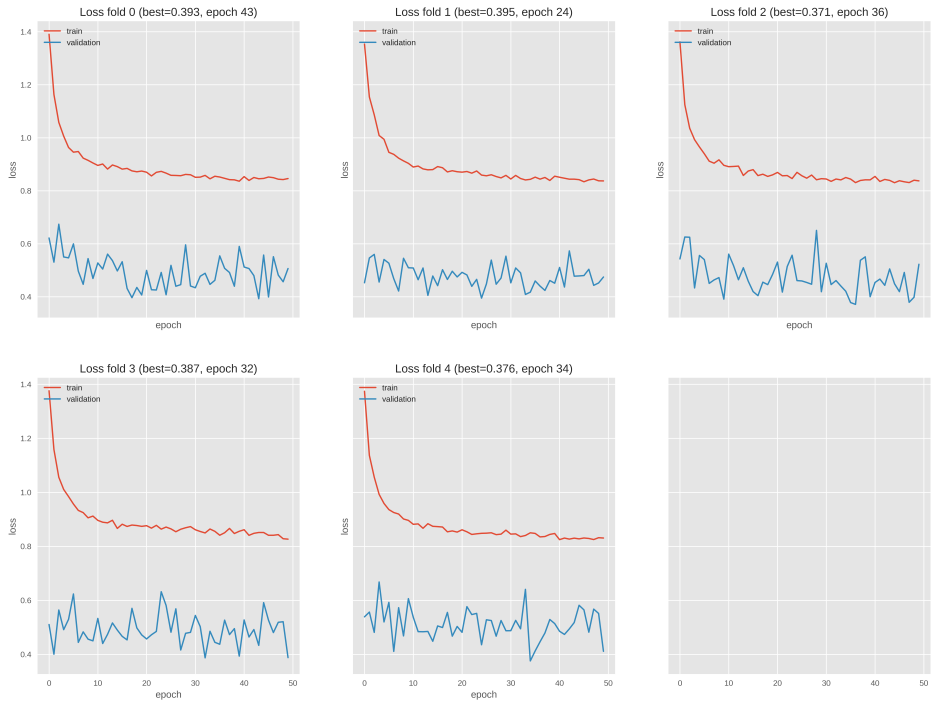


Figure 4.5: 5-Fold training loss history of the main 48-hour model.

Model performance Using this training configuration we obtained a cross validated Receiver Operating Characteristic Area Under the Curve (ROC AUC) of 0.8933 (± 0.0032) for the training set, and 0.8735 (± 0.0025) ROC AUC for the cross validation set (Figure 4.6). Using a 0.5 decision threshold, the model reaches 75.423% sensitivity at 82.776% specificity. Figure 4.5 shows the loss curves of each cross validation fold.

SAPS-II results as baseline SAPS-II uses data from the first 24 hours of ICU stay to calculate a score, which in turn is converted into a mortality probability. In order to compare our approach with SAPS-II predictions and performance, we trained our convolutional architecture using data from the first 24 hours using similar parameters to the ones we already used for the 48 hour version (same values except dropout probability which we set to 0.5). For the comparison we used the SAPS-II implementation provided

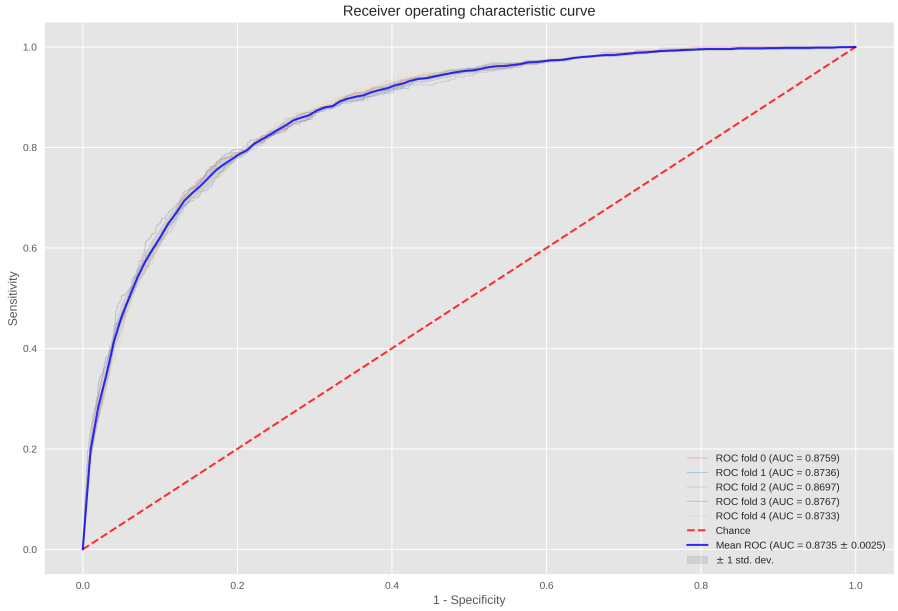


Figure 4.6: ConvNet (48h) 5-Fold cross validated ROC AUC.

by the authors of the MIMIC-III code repository (A. E. Johnson, Stone, Celi & Pollard, 2018). The 24 hour version of our model obtained a 0.8223 (± 0.0075) ROC AUC 5-fold cross-validation score, against 0.7372 (± 0.0091) for the SAPS-II model. We also retrained the original SAPS-II model which yielded a 0.7390 (± 0.0199) ROC AUC 5-fold cross-validation score. Finally, we used the SAPS-II individual scores to train a logistic regression classifier, for a 0.7633 (± 0.0220) ROC AUC 5-fold cross-validation score. Figures 4.7 and 4.8 show the corresponding ROC plots for the the 24 hour ConvNet and the SAPS-II score. Figures 4.9 and 4.10 show the ROC plots for the retrained SAPS-II versions.

Comparison with related works

This ConvNet model, which we have named ISeeU (Caicedo-Torres & Gutierrez, 2019) shows strong performance on the MIMIC-III SAPS-II based dataset with low

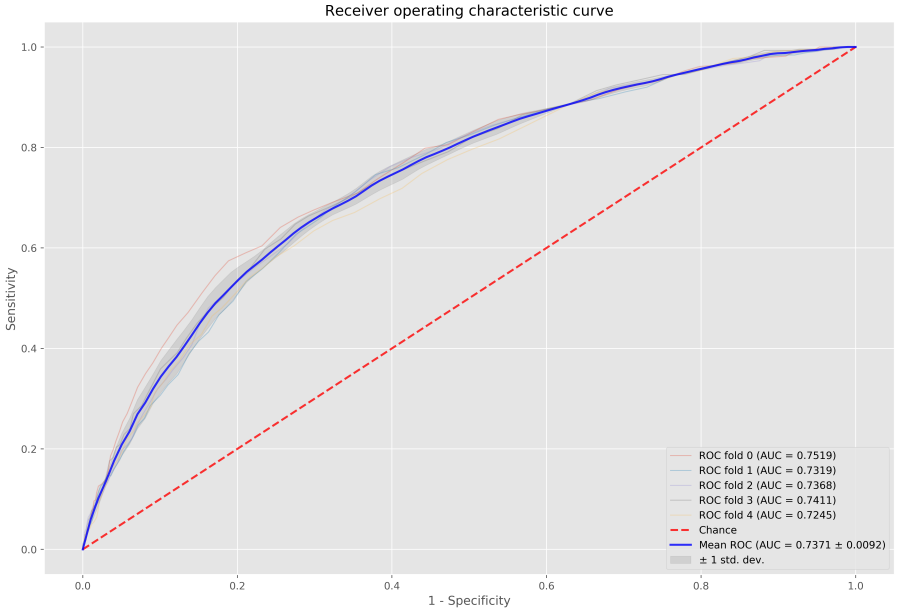


Figure 4.7: SAPS-II model 5-fold cross validated ROC AUC.

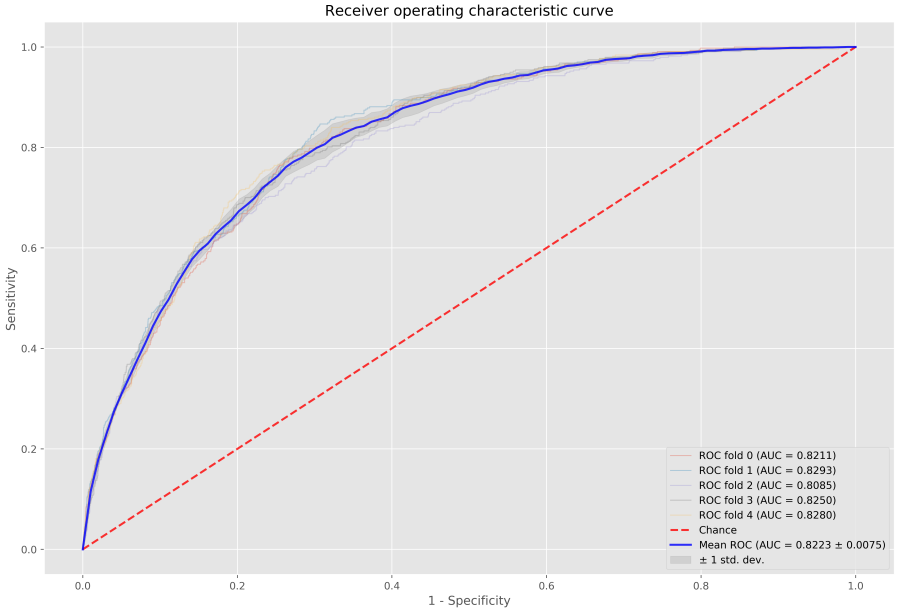


Figure 4.8: ConvNet (24h) 5-fold cross validated ROC AUC.

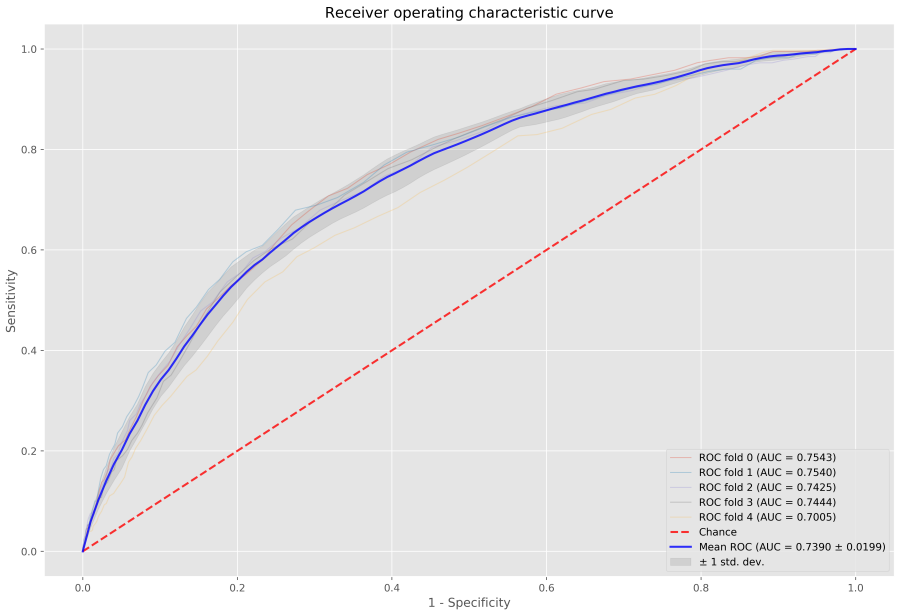


Figure 4.9: Validation ROC AUC for the retrained version of SAPS-II.

variability across folds. Also performance over training and validation data are close, evidencing that our model exhibits signs of good generalization properties, as there is no serious overfitting occurring (0.8933 (± 0.0032) training ROC AUC vs 0.8735 (± 0.0025) validation ROC AUC).

Validation performance reaches the state of the art for mortality prediction on MIMIC-III data and a comparable feature set (95% CI [0.870396, 0.876604] against a 95% CI [0.873706, 0.882894] corresponding to the performance reported by (Purushotham et al., 2018)). Our results show that a single convolutional architecture is able to handle both temporal and static MIMIC-III features using simple time replication for the static inputs, instead of using a recurrent/feedforward hybrid architecture as in (Purushotham et al., 2018). Table 4.5 shows a comparison with related works in the relevant literature.

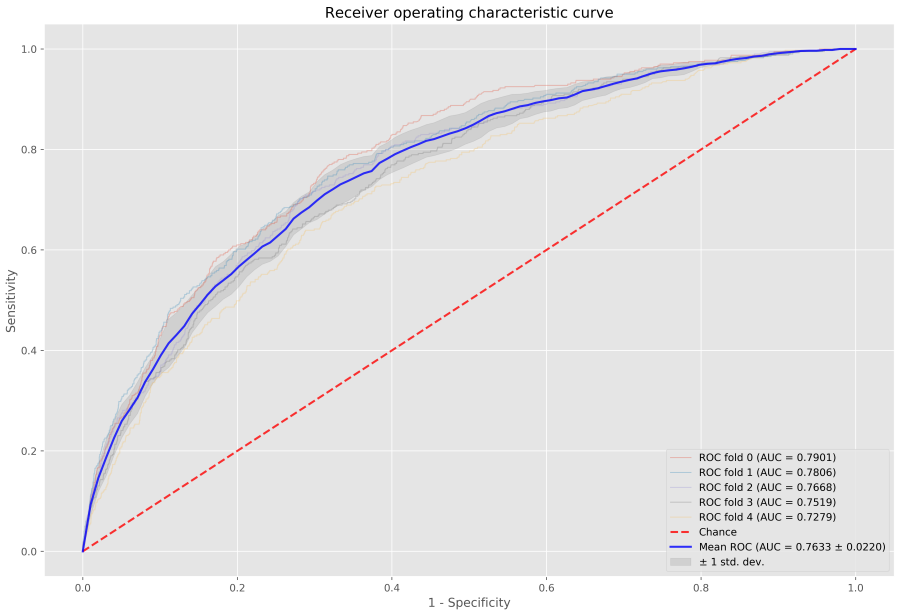


Figure 4.10: Validation ROC AUC for the logistic regression classifier trained on top of individual SAPS-II scores.

Issues observed in the training process

Training a deep neural network is not an easy task. There is still much to be known about the intricacies of minimizing a high dimensional, non-convex error surface with multiple critical points and large valleys. However some of our experiences in this research project proved themselves illuminating and validated some of the research findings others have put forth. In other cases our findings deviated somewhat of the expected or were counter-intuitive.

The context in which these issues were observed is related to data leakage. Data leakage occurs when information from the training process “leaks" to the validation set, so the validation results are optimistically biased. This can happen in several ways such as using samples coming from the same experimental unit in both training and validation/test sets, or by using the whole dataset to compute a Principal Component

Model	Type	Dataset	Task	ROC AUC
GRU-D (Che, Purushotham, Cho et al., 2016)	Recurrent	MIMIC-III (99 features)	ICU mortality	0.8527 ± 0.003
MMDL (Purushotham et al., 2018)	Hybrid	MIMIC-III (20 features)	ICU mortality	0.8783 ± 0.0037
GBTmimic (Ba & Caruana, 2014)	Hybrid + Gradient Boosted Trees	LA Children's Hospital PICU (48 features)	60-day mortality	0.7898 ± 0.030
ISeeU (Caicedo-Torres & Gutierrez, 2019)	ConvNet	MIMIC-III (22 features)	ICU mortality	0.8735 ± 0.0025

Table 4.2: Our results and results reported by related works (all use 48 hour data). ROC AUC results are mean and standard deviation from a 5-Fold cross validation run, except for the GBTmimic model (Ba & Caruana, 2014), which averages 5 different 5-Fold CV runs.

Analysis basis for dimensionality reduction. In our case data leakage came about because we used the whole dataset to calculate the mean and variance for standardization purposes. In this way the network was able to improperly leverage information from the training set to predict the correct class at inference time on the validation set. To resolve this, we modified the relevant code to calculate the mean and variance using the training set only, and since we were using 5-Fold cross-validation this implied calculating mean and variance of each training fold and using them to standardize the remaining folds.

After solving data leakage and re-training the model, performance was drastically reduced (from 0.88 ROC AUC down to 0.82 ROC AUC), highlighting the dangers of data leakage. Our efforts to restore the original performance of our model were successful and allowed us to attain the results reported in the last section. These efforts included changing the architecture from a two convolutional layer ConvNet to a single convolutional layer model, but with significantly more parameters on the dense layers. In the end we were able to go back to the original performance, and during this process we observed interesting things, some of which are summarized next.

Mean imputation of missing Urine Output worked best As explained before, one important aspect of this work is that we rely on simple imputation techniques like mean imputation and forward/backward filling, in lieu of more mathematically involved

options as adaptive imputation (Che, Purushotham, Cho et al., 2016), Multivariate Gaussian Processes (Razavian & Sontag, 2015), or adding a binary imputation mask to the inputs so the network can know which inputs have been imputed (Lipton et al., 2016).

As part of our strategy we decided that a sensible way to handle missing inputs for the Urine Output variable was to impute them to zero, under the reasoning that missing observations could correspond to no urination. It also seemed to make more sense compared to mean imputation, since it worried us that using mean imputation would lead to abnormally high Urine Outputs that were not physiologically correct.

We subscribed to that hypothesis for the majority of our experiments, but we decided to explore mean imputation as part of our efforts to improve model performance. When switching to mean imputation we observed a gain of approximately 1% in cross validation ROC AUC (from 0.82 to 0.83), which was significant in the pursue of our final results. The exact reasons for this improvement are unclear, but we hypothesize that since not all patients had Urine Output readings the addition of mean Urine Output instead of zeros could have helped the ConvNet to more adequately identify more plausible normal values for that feature.

Stochastic Gradient Descent (SGD) generalized better than adaptive methods At the beginning of our experimental runs, Adam (Wilson, Roelofs, Stern, Srebro & Recht, 2017) was chosen as our optimization algorithm given that its use allows to automatically and adaptively set the learning rate, thus freeing the experimenter from having to find a suitable value for it (and reducing the hyperparameter search space, which is always an attractive proposition). We kept using Adam well into our experimental runs even after identifying the data leakage issue. At some point it was apparent that merely tuning the hyperparameters was not going to be enough to recover our original performance and we started looking for alternatives.

Research has shown that adaptive learning rate optimization algorithms as RMSProp (Tieleman, Hinton, Srivastava & Swersky, 2012) and Adam (Kingma & Ba, 2014) find minima with different characteristics compared to minima found by SGD. In particular it's been observed that Adam finds "sharp minima" (i.e. narrow valleys) while SGD prefers flatter ones (Wilson et al., 2017), which in turn affect the generalization properties of the corresponding models as those sharper minima generalize worse than the flatter ones (Keskar, Mudigere, Nocedal, Smelyanskiy & Tang, 2017; Hochreiter & Schmidhuber, 1997a).

Given this we switched from Adam to SGD with Nesterov momentum. The addition of Nesterov momentum (Nesterov, 1983) allows for a more stable training process. Subsequent runs offered an improvement of approximately 3% taking our mean cross validation ROC AUC from 0.83 to 0.85, while the mean training ROC AUC came from 0.90 to 0.89. This is even more remarkable as we did not perform any optimization on the learning rate value, but instead we used somewhat standard values for the parameters. This supports the earlier findings about the better generalization properties of minima found by SGD, this time not in carefully constructed examples as in (Wilson et al., 2017) but in real-world datasets, used in a real-world application.

Average Pooling worked better than Max Pooling Dimensionality reduction in the form of pooling is a key component of convolutional architectures. It allows the network to discard useless and/or redundant information, which in turn helps to attain better performance and reduces computational cost (Goodfellow et al., 2016). Pooling is often realized through Max Pooling layers, which downsample the receptive fields generated by a convolutional layer and create a lower dimensional representation that preserves the most salient and important features. Pooling usually computes some statistical measure inside pre-defined neighborhoods in the input, taking the name of Max Pooling when the max function is used as said statistical measure. Unsurprisingly, when the

average is used it is called Average Pooling.

Seemingly, Max Pooling tends to be more popular than Average Pooling in the research literature, especially in Computer Vision, and many times it is the default choice for ConvNets (some research has been undertaken about which option is best, with no clear winner (Boureau, Ponce & LeCun, 2010)). As such we used Max Pooling since the beginning of the experiments, as we did with our chosen optimizer. Given the issues mentioned earlier we also decided to replace the Max Pooling layers for Average Pooling layers, which resulted in an approximate gain of 2% ROC AUC (0.85 to 0.87, roughly). The take-home message in this case is that both pooling techniques should be tested as part of the model development process.

Model interpretability

Having covered the most remarkable aspects of our training runs and our model performance, now we turn to the other main component of this work, interpretability. At this point we had good performance (compared to the relevant literature) and we focused on making the models more interpretable.

As discussed before we leveraged a fast approximation of the Shapley Values for interpretability. We used the DeepLIFT implementation provided by its authors to compute our input feature importances from a model trained on one cross validation fold. We selected zero (empirical mean) as the reference value. We also computed importances for individual patients and at the dataset level, and created a series of visualizations to offer explanations for the predictions of the model.

Visualizations are the crux of our approach to interpretability. In our literature review it was found that an important issue for the adoption of Deep Learning models was the absence of a sense of feature importance, which is readily available in models like logistic regression, for example. As deep neural networks are highly non-linear and hierarchical, there is no simple way to offer something similar. However as we

have discussed, Shapley Values offer the marginal importance of the contribution of every input, and through a fast approximation we are able to compute them and use them as a form of evidence for/against the model's decisions. Armed with these values we created a series of visualizations that show the feature importances at various levels of aggregation, so the end user can clearly establish which inputs the neural network considers important, and the nature of their importance (positive or negative).

The proposed visualizations are designed to combine patient features with their importance towards the predicted probability of death. A crucial feature of our approach is that these visualizations constitute a form of *post hoc interpretability* (Lipton, 2016) insofar as they try to convey how the model regards the inputs in terms of their impact on the predicted probability of death, without having to explain the internal mechanisms of our neural network, nor sacrificing predictive performance by using a surrogate model for predictions.

To highlight the benefits of our approach, we were particularly interested in cases in which we can compare usual medical expectations about the interpretation of certain features and their importance, with the feature importances computed by our model. Because of the cross-validation procedure, the dataset was randomly shuffled and partitioned into 5 folds using stratified random sampling to preserve class ratios. We inspected some of the first patients in the last fold (as the model was trained on the first four folds) and we found an interesting case of a 20 years old patient (mean age of the cohort is 63.828, with a standard deviation of 15.576). Given our initial motivation, we wanted to see if the model used age in a way consistent with medical expertise.

We present four types of visualization: Aggregate predictor importance, predictor importance by hour, positive and negative predictor importance, and dataset-level feature importance. We will describe the details of each visualization type using a real patient from MIMIC-III as example.

Aggregate predictor importance Here we treated the patient tensor representation as an image and we grouped feature importance attribution semantically (i.e. observations belong to a particular predictor, as pixels on an image belong to an object) to find net contributions per predictor. Figure 4.11 shows the feature importances computed for a single patient (predicted probability of mortality: 0.5764, observed mortality: 1), summed over 48 hours for each individual predictor and then normalized over the predictor set. As mentioned this visualization shows the importance of each predictor as a whole, highlighting with red those predictors that contribute to a positive (death) prediction, and with blue those that contribute to a negative (survival) prediction. Since hourly importances can be either positive or negative in sign, it is possible that the total importance might be close to zero (gray background), even if the individual importances are not. We can clearly see that the network is assigning high positive importance to the components of the Glasgow Comma Scale - GCS, and high negative importance to the age of the patient. These are interesting because GCS values are shown to be abnormal, and the patient is very young (20 years old), and it is plausible that a young age is negatively correlated with mortality in the ICU.

Predictor importance by hour In this visualization we further de-aggregate importance and show the individual approximate Shapley Values for each input value and hour (Figure 4.12). We can see evidence for the non-linear dynamics the network has learned, as values from the same predictor have different importance across the temporal axis.

Positive and negative predictor importance Here we can treat positive and negative importances separately to have a better sense of how each predictor affects the final prediction and how it compares to the rest. Figure 4.13 shows a barplot with positive and negative importance grouped by predictor.

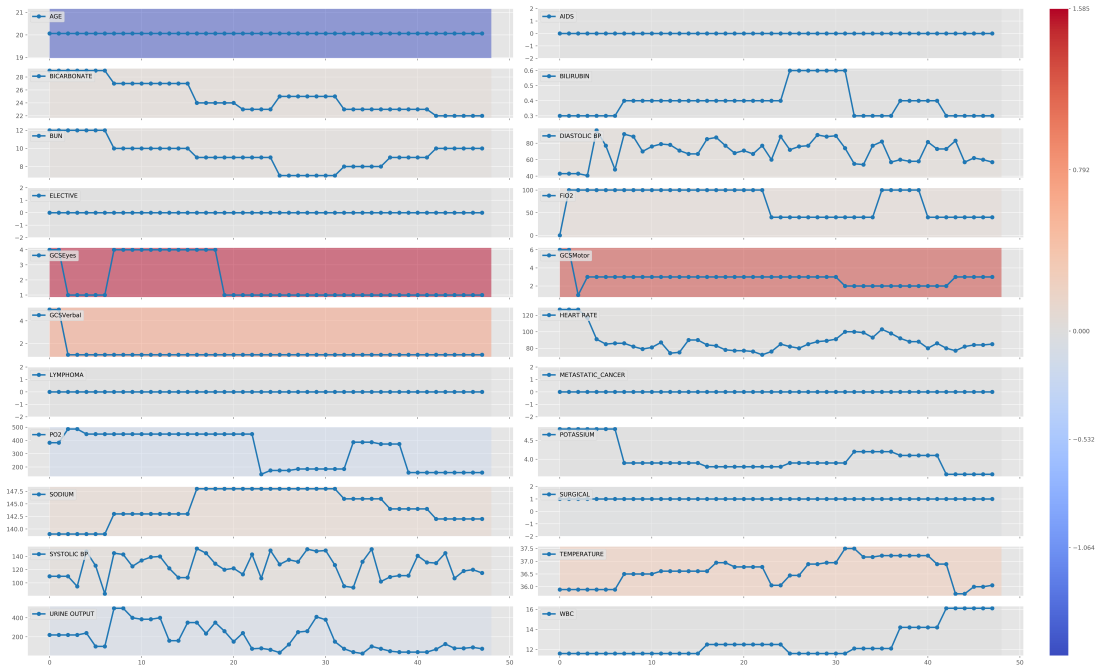


Figure 4.11: Marginal (total) predictor importance for a single patient.

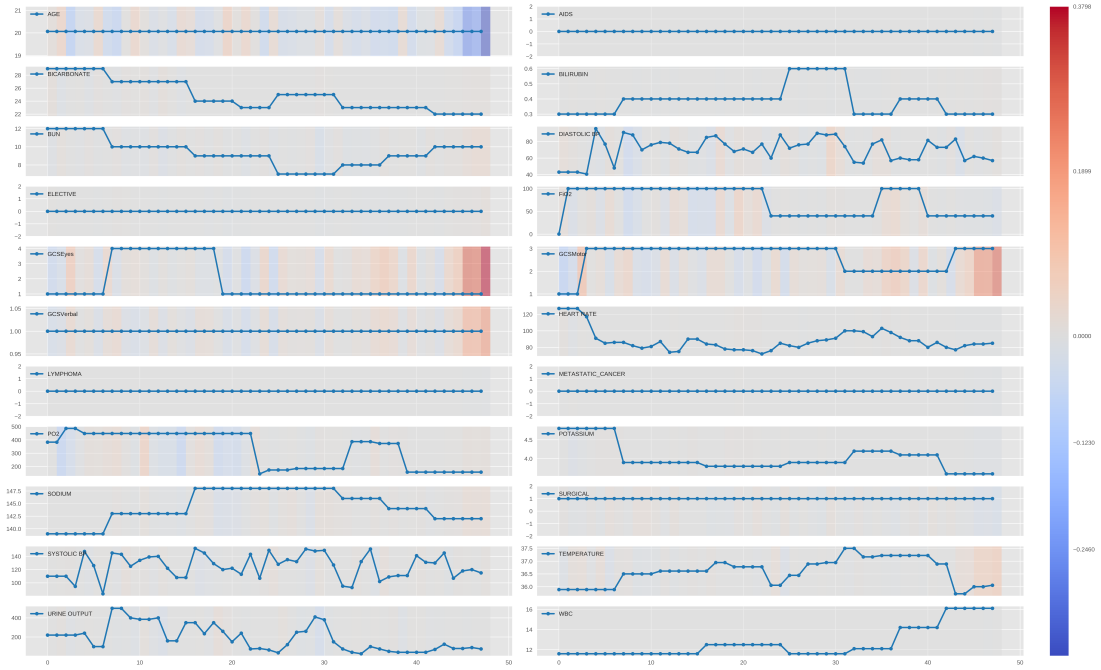


Figure 4.12: Predictor importance by hour for a single patient.

Dataset-level feature importances Additionally we computed importances for the validation set to offer interpretability at the dataset level. Table 4.3 and Figures 4.14 and

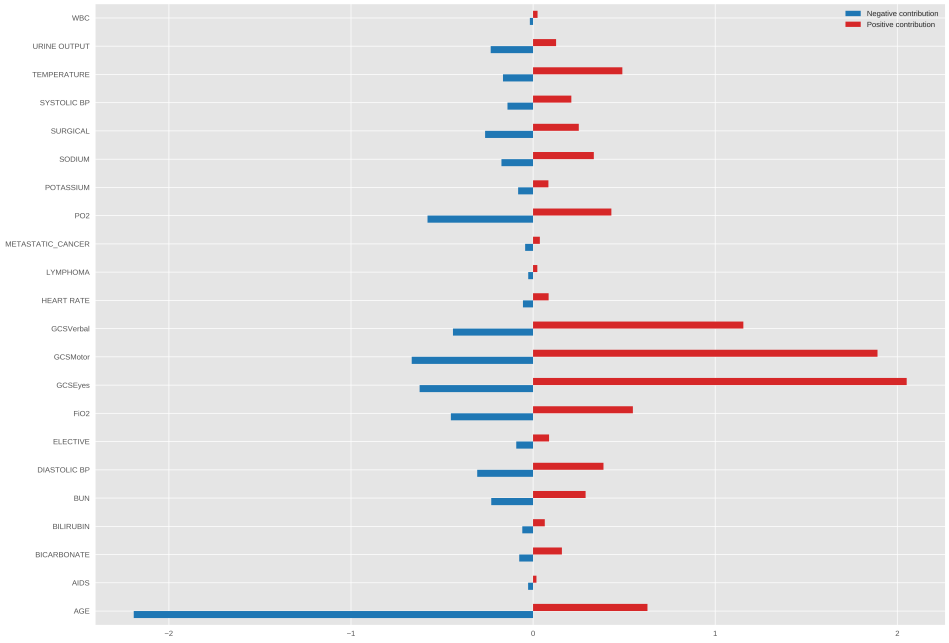


Figure 4.13: Negative and positive importance of each predictor for a single patient. The x axis corresponds to total feature importance.

4.15 show dataset-level statistics for the normalized positive and negative importance of each predictor.

Discussion

For this particular patient the predictor marginal importance visualization shows that the model is attending to sensible features to predict mortality. As mentioned previously, the model is attending to the components of the GCS scale which show abnormal values and assigns them a positive contribution to mortality. PO2, FiO2, blood sodium and temperature are also regarded, to various degrees, as evidence favoring predicting mortality. On the other hand the patient age is regarded by the model as strong evidence

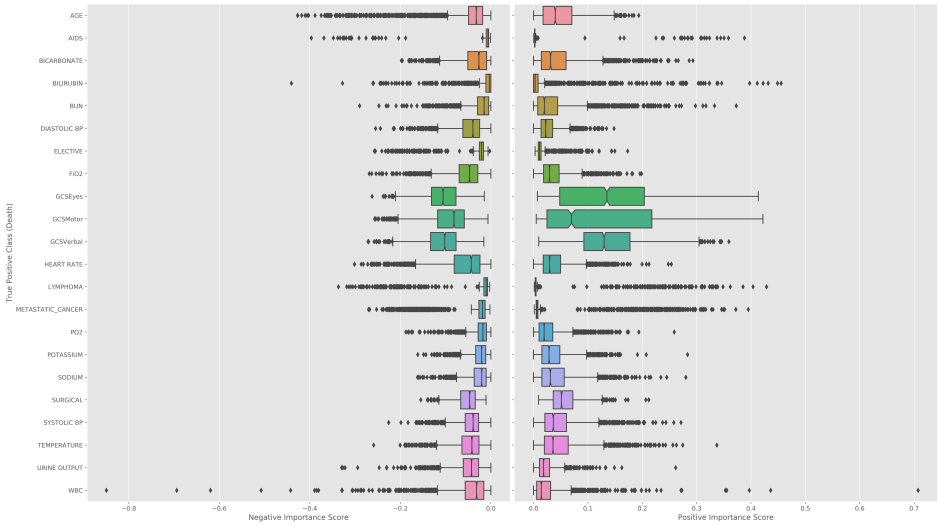


Figure 4.14: Boxplots for dataset-level negative and positive predictor importance for the positive class.

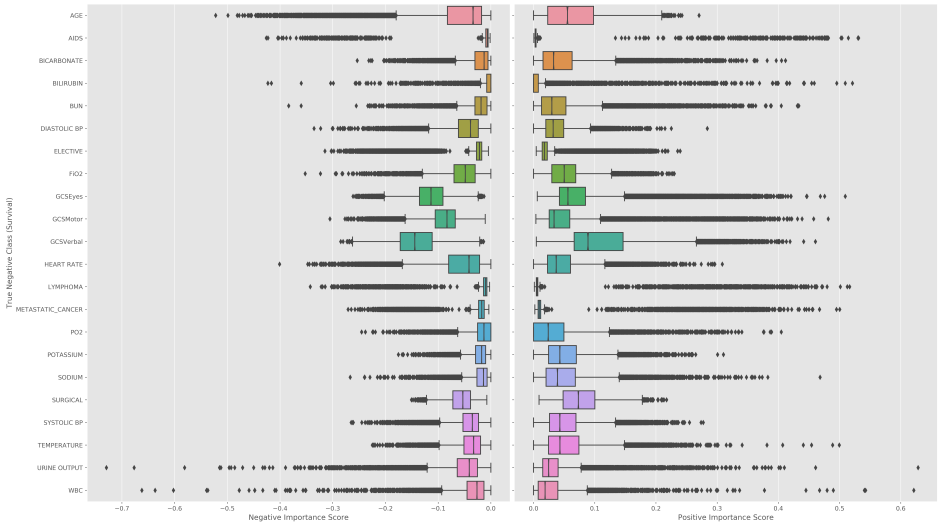


Figure 4.15: Boxplots for dataset-level negative and positive predictor importance for the negative class.

	Negative Importance Score								Positive Importance Score							
	n	Mean	Std	Min	Q1	Q2	Q3	Max	n	Mean	Std	Min	Q1	Q2	Q3	Max
Feature																
Age	22413	-0.067	0.082	-0.522	-0.078	-0.034	-0.018	-0.000	22413	0.062	0.047	0.000	0.022	0.054	0.095	0.270
AIDS	22413	-0.010	0.028	-0.426	-0.010	-0.007	-0.005	-0.001	22413	0.007	0.036	0.000	0.002	0.003	0.004	0.531
Bicarbonate	22413	-0.024	0.026	-0.254	-0.033	-0.014	-0.006	0.000	22413	0.046	0.044	0.000	0.015	0.033	0.063	0.412
Bilirubin	22413	-0.005	0.016	-0.441	-0.008	0.000	0.000	0.000	22413	0.008	0.029	0.000	0.000	0.000	0.008	0.521
BUN	22413	-0.023	0.023	-0.384	-0.030	-0.018	-0.007	0.000	22413	0.041	0.044	0.000	0.012	0.029	0.052	0.433
Diastolic BP	22413	-0.048	0.034	-0.336	-0.062	-0.039	-0.024	0.000	22413	0.037	0.024	0.000	0.019	0.031	0.048	0.284
Elective	22413	-0.042	0.054	-0.315	-0.027	-0.022	-0.017	-0.003	22413	0.029	0.033	0.003	0.013	0.017	0.022	0.240
FiO_2	22413	-0.053	0.032	-0.352	-0.070	-0.048	-0.030	-0.000	22413	0.050	0.029	0.000	0.028	0.048	0.068	0.230
GCSEyes	22413	-0.113	0.036	-0.263	-0.135	-0.113	-0.090	-0.013	22413	0.085	0.068	0.006	0.042	0.058	0.096	0.509
GCSMotor	22413	-0.090	0.038	-0.305	-0.107	-0.083	-0.067	-0.006	22413	0.063	0.070	0.004	0.026	0.034	0.067	0.481
GCSVerbal	22413	-0.138	0.043	-0.285	-0.170	-0.141	-0.106	-0.014	22413	0.116	0.071	0.005	0.067	0.092	0.153	0.460
Heart rate	22413	-0.059	0.053	-0.401	-0.080	-0.042	-0.022	0.000	22413	0.046	0.034	0.000	0.022	0.036	0.059	0.308
Lymphoma	22413	-0.016	0.034	-0.343	-0.014	-0.010	-0.008	-0.002	22413	0.014	0.050	0.001	0.004	0.005	0.007	0.516
Metastatic cancer	22413	-0.026	0.035	-0.271	-0.024	-0.017	-0.013	-0.002	22413	0.024	0.061	0.001	0.007	0.009	0.011	0.500
PO_2	22413	-0.018	0.021	-0.245	-0.026	-0.014	0.000	0.000	22413	0.033	0.038	0.000	0.000	0.023	0.048	0.405
Potassium	22413	-0.022	0.018	-0.176	-0.029	-0.018	-0.010	0.000	22413	0.050	0.036	0.000	0.023	0.041	0.068	0.310
Sodium	22413	-0.021	0.021	-0.267	-0.027	-0.015	-0.007	0.000	22413	0.049	0.041	0.000	0.020	0.038	0.067	0.468
Surgical	22413	-0.056	0.023	-0.155	-0.072	-0.053	-0.038	-0.008	22413	0.074	0.033	0.009	0.047	0.071	0.098	0.217
Systolic BP	22413	-0.042	0.028	-0.265	-0.053	-0.036	-0.024	0.000	22413	0.052	0.036	0.000	0.026	0.042	0.068	0.277
Temperature	22413	-0.039	0.027	-0.259	-0.052	-0.034	-0.020	0.000	22413	0.054	0.042	0.000	0.024	0.042	0.073	0.500
Urine output	22413	-0.051	0.042	-0.730	-0.063	-0.041	-0.026	0.000	22413	0.032	0.032	0.000	0.015	0.024	0.039	0.628
WBC	22413	-0.035	0.036	-0.849	-0.046	-0.027	-0.013	0.000	22413	0.028	0.032	0.000	0.007	0.018	0.039	0.707

Table 4.3: Statistics of predictor importances at the dataset level. GCSEyes, GCSMotor, and GCSVerbal stand for Glasgow coma scale (eyes), Glasgow coma scale (motor), and Glasgow coma scale (verbal) respectively. WBC stands for white cell blood count. FiO_2 and PO_2 stand for fraction of inspired oxygen and oxygen pressure in blood, respectively.

against mortality, followed by urine output.

The marginal importance visualization allows us to see something interesting: the model assigns a negative net contribution to the fact that the patient was admitted after surgery, this is, the model regards the surgical admission as evidence for survival (however at the dataset level, median positive importance for surgical admissions are greater across classes than their negative counterpart, i.e. the model tends to see surgical admission as evidence for mortality). This could be due to correlations present in the underlying dataset, or higher order interactions between predictors. The latter is attested by the predictor plus hour visualization, which shows that for static predictors, different observations across time of the same predictor are assigned different contributions, sometimes with different sign. It is also worth noticing that while the patient's surgical admission is regarded as evidence for survival, the fact that the surgery was not an elective surgery is considered as evidence for mortality, which is sensible. However both input features must not be analyzed separately (i.e. they correspond to a single concept

in SAPS-II score (Gall et al., 1993)). This is the kind of insight that interpretability efforts can reveal about black boxes, which is also absent in the majority of related works (Purushotham et al., 2018) (See table 4.4 for a comparison).

Dataset-level analysis of feature importance shows a high variance in attributed importance, both negative and positive. GCS components tend to be the features with the most importance attributed (especially positive importance for patients that eventually died), followed by age. On the other hand, there are a number of features with both low positive and low negative mean importance. Presence of AIDS or lymphoma, are deemed by the ConvNet as not carrying much weight for predicting either survival or death. Also some of the other predictors have modest mean importances. This could signal a possibility to simplify the input feature set and get better predictive performance.

Model	Dataset	Task	Interpretable?
GRU-D (Che, Purushotham, Cho et al., 2016)	MIMIC-III (99 features)	ICU mortality	No
MMDL (Purushotham et al., 2018)	MIMIC-III (20 features)	ICU mortality	No
GBTmimic (Ba & Caruana, 2014)	LA Children's Hospital PICU (48 features)	60-day mortality	Yes (dataset level)
ISeeU (Caicedo-Torres & Gutierrez, 2019)	MIMIC-III (22 features)	ICU mortality	Yes (patient and dataset level)

Table 4.4: Comparison between ours and commonly cited works in terms of patient-level and dataset-level interpretability.

4.2.2 Free-text medical notes model

After our examination of the usage of our physiological time series dataset for the creation of ISeeU, we will focus on the free-text medical notes dataset and the construction of a Natural Language Processing based mortality prediction model. Our second prediction model, which we have named ISeeU2, is also a convolutional neural network.

In this case we used Tensorflow v2 as it had been just released and its new capabilities were worth the switch. The specific architecture of ISeeU2 (figure 4.16) includes a text embedding layer to convert a bag of words text representation into 10-dimensional dense word vectors. The output of the embedding layer is then fed to a convolutional layer with 32 channels and a kernel size of $5x10$ (stride 1), followed by ReLU activations and a max-pooling layer with a pool size of $1x3$ (stride 1). The obtained representation is then fed to a $50x1$ dense layer with ReLU activations connected to a one-neuron final layer with sigmoid activation, which computes the mortality probability.

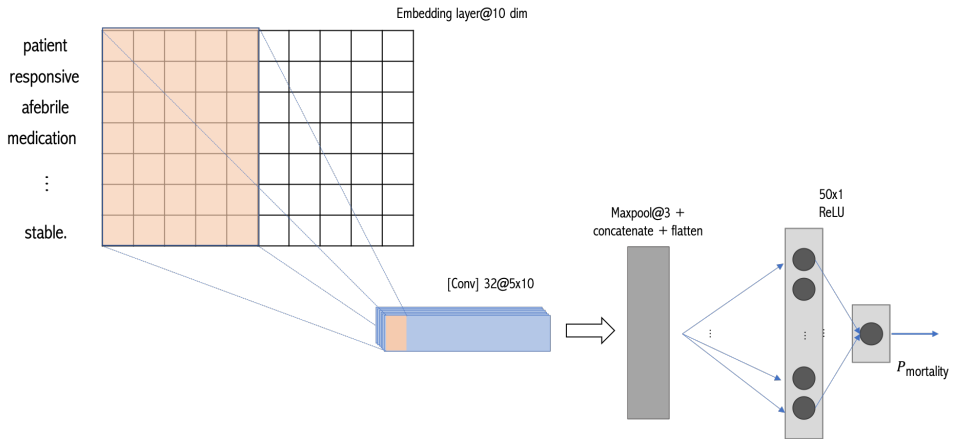


Figure 4.16: Deep learning model architecture.

As mentioned before, our dataset is highly unbalanced (negative outcomes represent just 9.78% of training examples), reason why we used again a weighted logarithmic loss assigning more importance to the positive class, i.e. patients that died in the ICU. We used 5-fold cross-validation to assess the model performance and place a confidence estimate on it. We did not perform any substantial hyperparameter optimization other than conservatively varying the number of channels of the convolutional layer and the number of neurons of the first fully connected layer of the network. Our choice of optimizer in this case was Adam (Kingma & Ba, 2014) with default Tensorflow-provided parameters. One important factor that influenced our optimizer choice was the

observed instability of Stochastic Gradient Descent (manifested in the frequent collapse of the model into trivial solutions, i.e. always predicting survival or always predicting death) and the difficulty to find a suitable value for the learning rate. Our model was trained for three epochs per training fold, and we kept the lowest loss model of each run.

Text pre-processing

One of our goals was to develop deep learning models that need little to no input pre-processing in order for it to be as widely applicable as possible. Keeping with that we used the NLTK library (Loper & Bird, 2002) to remove English stop-words and the *Tensorflow.keras* default tokenizer to vectorize the text notes, keeping the 100k most frequent words; and no further pre-processing was attempted. The tokenizer was fitted only on the training folds to avoid data leakage. Finally, we set the maximum note length to 500, so notes with a larger word count were truncated at the beginning and those with a smaller word count were padded with zeroes, at the beginning as well.

Model performance and comparison with baselines

Using this configuration we obtained a 5-fold cross validation Receiver Operating Characteristic Area Under the Curve (ROC AUC) of 0.8629 (± 0.0058) as seen in figure 4.17. Using a 0.5 decision threshold, the model reaches 72% sensitivity at 83% specificity. It is worth noting that in this case MaxPooling offered better performance than Average Pooling, supporting our conclusion that there doesn't seem to be a strictly superior choice between the two and rather they should be compared on a per-case basis. Again, we provide some baseline models to compare with our text-based model to better assess its performance. Concretely, we have included results for the SAPS-II risk score and a recurrent neural network.

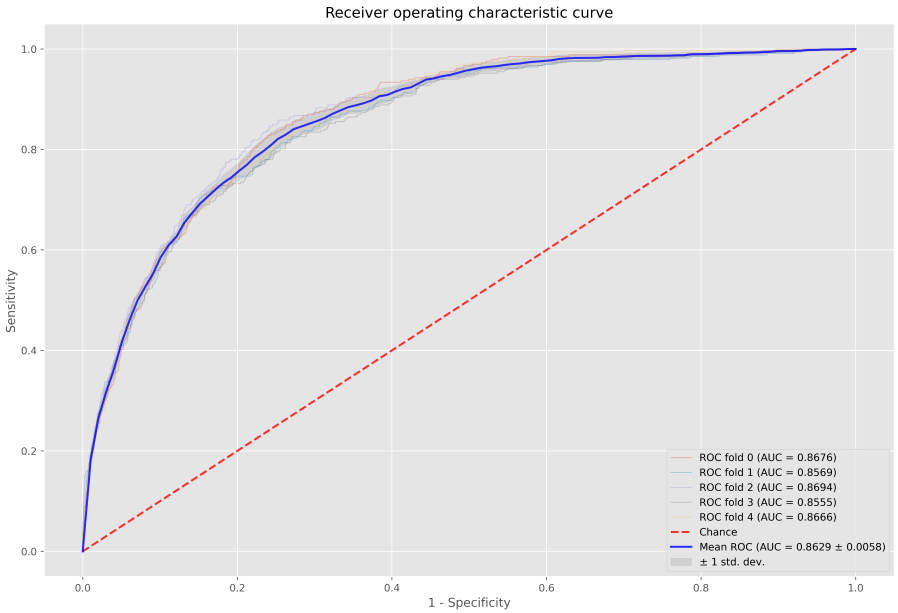


Figure 4.17: ConvNet 5-fold cross validated ROC AUC.

SAPS-II For SAPS-II, we again used the SAPS-II implementation provided by the authors of the MIMIC-III code repository (A. E. Johnson et al., 2018) (Given the results we obtained earlier showing that SAPS-II and its retrained variants were so similar in terms of ROC AUC performance, we don't include any retrained variant in this comparison). On the other hand we kept the parameters of our original model intact, but only used the first 24 hours worth of patient nursing notes for its training. The 24 hour version of our text model obtained a 0.8155 ($\pm$0.0102) ROC AUC 5-fold cross-validation score, against 0.7448 ($\pm$0.0117) for the SAPS-II model. Figures 4.18 and 4.19 show the corresponding ROC plots for the two models.

LSTM Our second baseline is a recurrent neural network based on the Long Short Term Memory (LSTM). LSTM is a neural network model designed to handle sequential input data with temporal dependencies (Hochreiter & Schmidhuber, 1997b), and it has been used extensively in Natural Language Processing tasks. We trained a deep

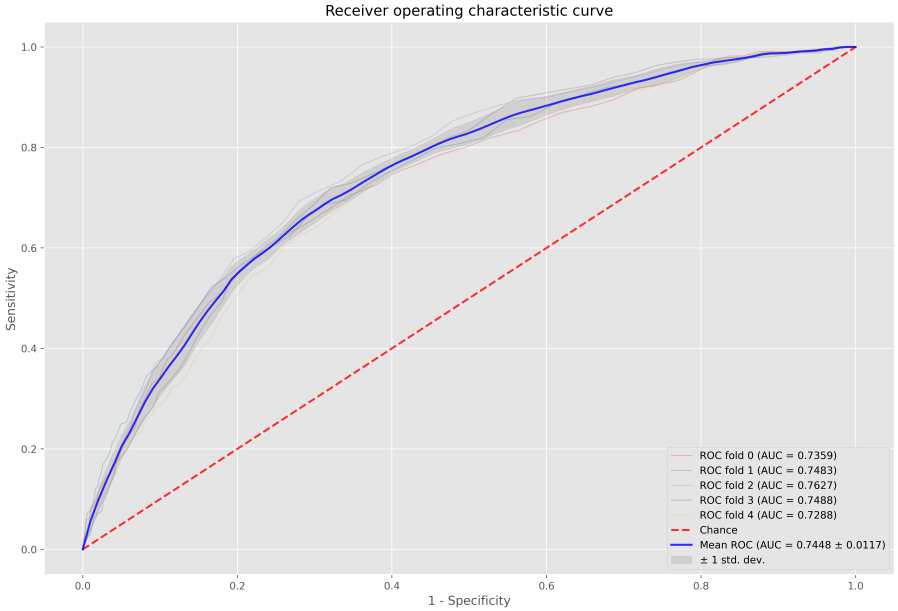


Figure 4.18: SAPS-II model 5-fold cross validated ROC AUC.

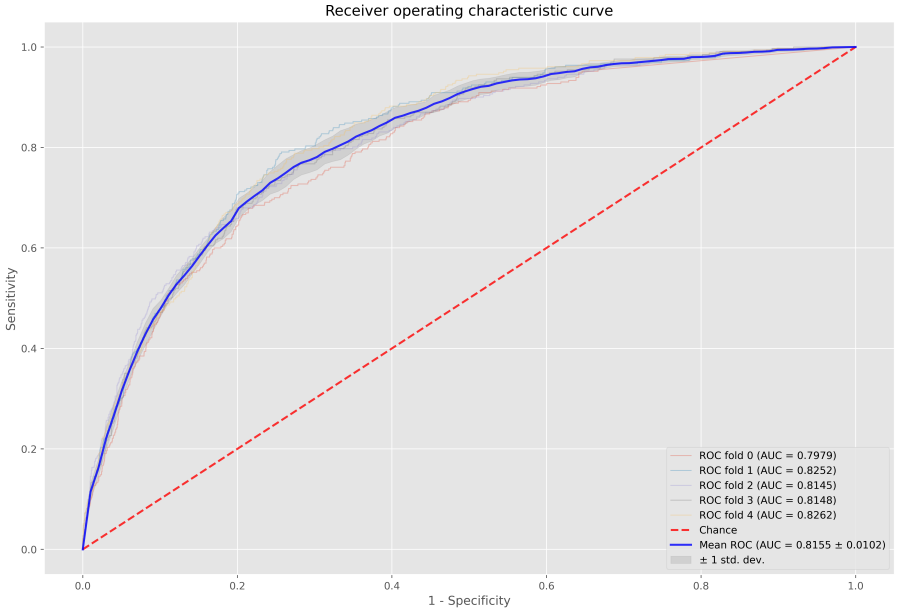


Figure 4.19: ConvNet (24h) 5-fold cross validated ROC AUC.

neural network with a bidirectional LSTM layer with 100 units, followed by an extra 100-unit LSTM layer, a 50-unit dense layer ReLU activation, and a final sigmoid layer. As we did with our original convolutional model, an embedding layer was used to create 10-dimensional dense vectors to feed the initial layer of the LSTM and the same text preprocessing pipeline was used (save for a now 1000-word maximum note length). Finally dropout with probability 0.5 was applied to control overfitting. With this particular architecture we were able to obtain a 0.7839 (± 0.0076) ROC AUC 5-fold cross-validation score (Figure 4.20).

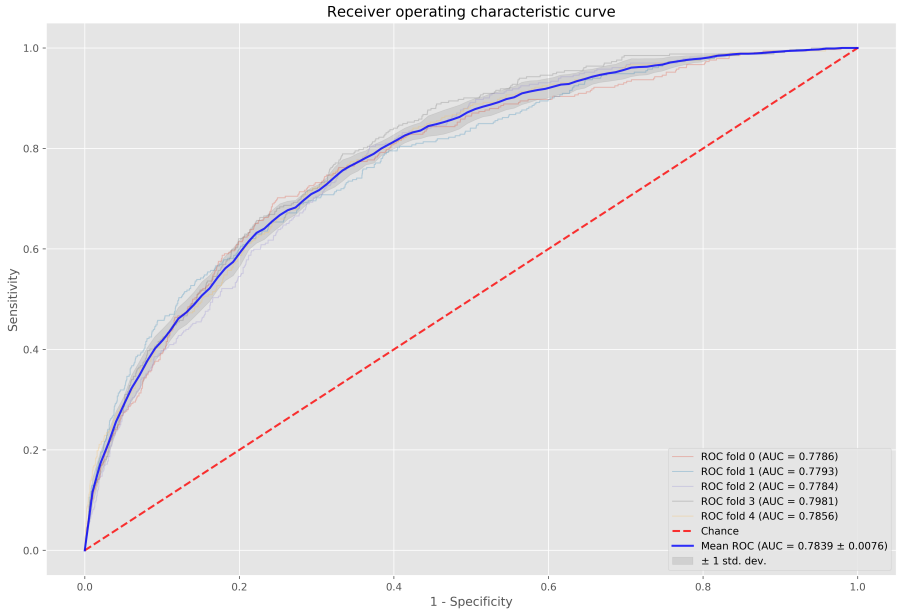
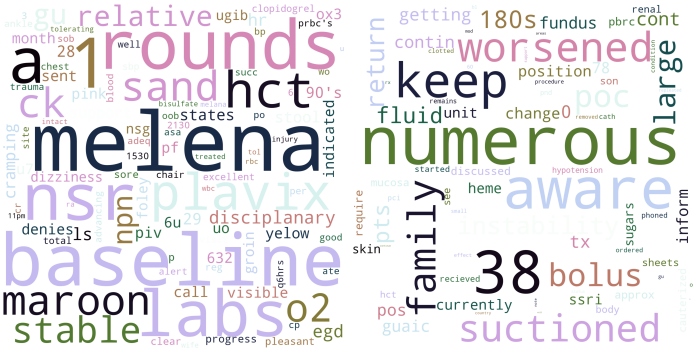


Figure 4.20: Deep LSTM 5-fold cross validated ROC AUC.

Model interpretability

Using the DeepLIFT implementation provided by (Lundberg & Lee, 2017b) which works appropriately with Tensorflow 2 models, we calculated word importances for our model, using the empirical mean of the input embedding vectors as reference value.

Using these values we designed and built visualizations to show the importance of each word in the original nursing note used as input. We have selected some examples at random from both the training set and the validation set of the last cross-validation run to show the behavior of the model and the way it regards certain words in the input notes. Our proposed visualizations include word clouds and text heatmaps (Figures 4.21 and 4.22).



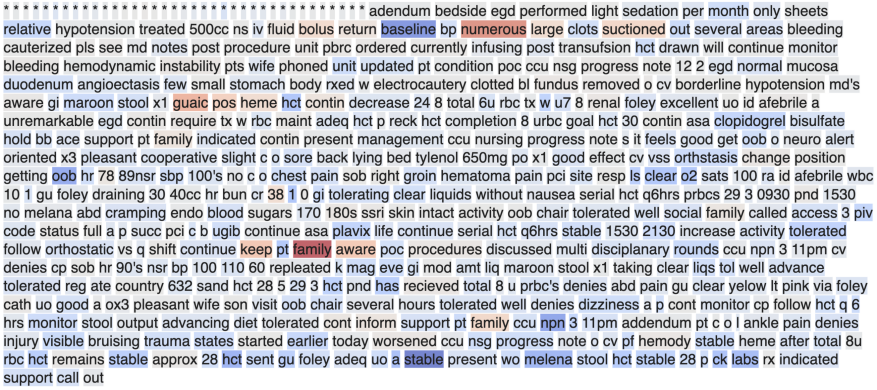


Figure 4.21: Top: Word clouds generated for one specific patient in the training set show the words deemed as most important for both survival (left) and death (right) prediction. Bottom: Text heatmap showing words, their importance and their context in sentences, generated for one specific patient in the training set. Red color denotes evidence for death, and blue color represents evidence for survival. Words with a gray background are not considered important for the prediction task by the network. Padding characters are represented by asterisks.

Word clouds are an interesting way to visualize words and their importance at the

same time, but they don't capture the context in which words live, potentially leading to erroneous interpretations. For example, the survival word cloud in Figure 4.21 shows *melena* as associated with survival, which is not readily understandable. However, when the word cloud is combined with the note heatmap, the reason becomes apparent, given the context of the word (*stable present wo melena stool*). We also observe that certain phrases and words are flagged intuitively, e.g. *guaic pos heme*, and also the fact that for this particular patient occurrences of Plavix/Clopidogrel in the note are flagged as evidence for survival. There are other instances in which results are not intuitive and may point to statistical flukes rather than strong causal features. As an example we can point to the phrases *return baseline bp numerous large clots suctioned*, and *continue keep pt family aware*, in which the words *clot* and *pt* seem to be flagged incorrectly.

Annotation smoothing In order to help ameliorate the sharp changes and inconsistencies observed at the sentence level we used a convolution filter to take into account the effect of the Shapley Values of all words in a particular sentence when generating the heatmap annotations, and provide a smoother and more intuitive result. Note that this is approximated since we are intent on using a basic pre-processing pipeline, without any advanced capabilities (i.e. sentence segmentation). A 5×1 convolution filter $[0.1, 0.2, 0.4, 0.2, 0.1]$ allows us to spread out the feature importance of individual words and to fade out weak importances that are due possibly to noise, while still keeping the most salient features. Figure 4.22 show our previous training set and a new validation set note with and without the convolution filter applied.

Note length and mortality probability High capacity machine learning models such as deep neural networks have the ability to leverage subtle correlations and patterns to attain very low training error in learning tasks. As shown in Table 3.7 and Figure 3.5, there is a difference in our sample between mean length of patients who survived

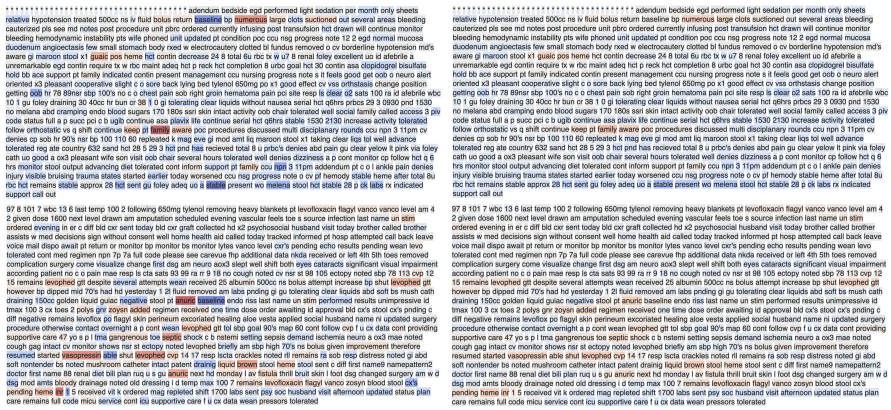


Figure 4.22: Text heatmaps with (right) and without (left) convolution smoothing. Bottom row corresponds to a nursing note from the validation set.

and those who had a negative outcome. A Mann-Whitney U test supplied further evidence, as we were able to reject the null hypothesis, i.e. distributions of the length of nursery notes are the same (p = 0.000), in favor of our alternative hypothesis, i.e. notes for patients that do not survive are longer. A possible explanation for the observed difference in length is that patients who are more sick are monitored more constantly and the written descriptions of their status need more detailed precisely because of their delicate condition, compared to less sick patients.

Having established that, we decided to investigate if our text model was attending somehow to that difference in distributions. For this purpose we inspected the importance score of the padding characters used by our pre-processing pipeline, with most of them being regarded as evidence for survival, which is consistent with our original conjecture that the model considers that shorter notes are correlated with a survival outcome (shorter notes have more padding characters). Figure 4.23 shows the distribution of approximate Shapley Values for padding characters.

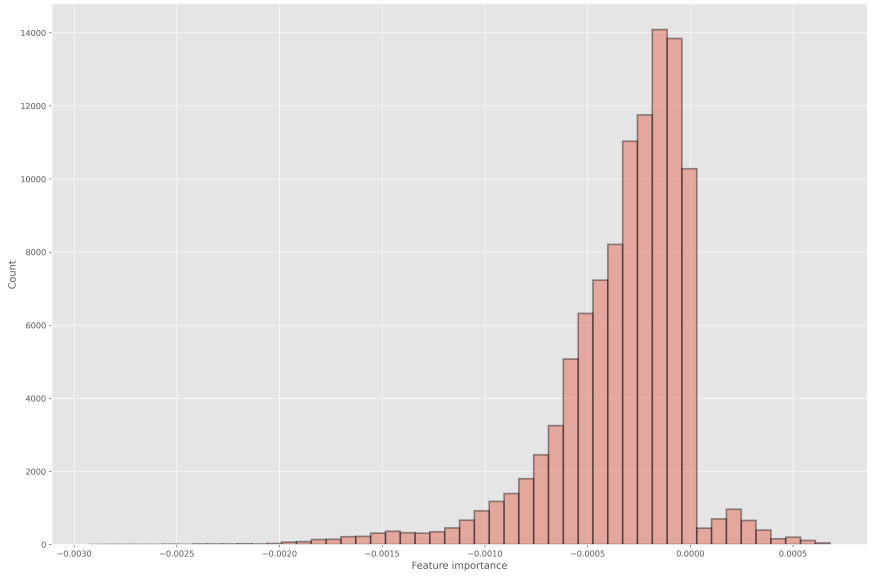


Figure 4.23: Distribution of approximate Shapley Values for padding characters. The histogram shows that most padding characters are deemed as evidence for survival by our model.

Discussion

Our convolutional text model shows interesting performance on the MIMIC-III dataset, with consistent results across validation folds, showing evidence for good generalization. Validation ROC AUC (95% CI [0.855689, 0.867888]) is competitive with published results in a comprehensive benchmark by (Purushotham et al., 2018) (95% CI [0.873706, 0.882894] ROC AUC) and our physiological time series model (95% CI [0.870396, 0.876604] ROC AUC). Moreover, our free-text model bypass some of the most important difficulties associated with the usage of physiological time series, i.e. inconsistent sampling times and missing values. On the other hand, the 24-hour version of our model still manages to surpass comfortably the SAPS-II baseline.

Our results are not directly comparable to those published by Grnarova el al (Grnarova et al., 2016) given that we restricted our input window to the first 48 hours of patient stay, instead of using all available notes up until the time of discharge. Results

published by Jo et al (Combining LSTM and Latent Topic Modeling for Mortality PredictionJo et al., 2017) show their models performing under 0.84 ROC AUC for mortality prediction using MIMIC-III data (48 hour mark), which is well below our results here. On the other hand, the model Vital + EntityEmb reported by (Jin et al., 2018) uses physiological data and a substantial text preprocessing pipeline that involves a second neural network for Named Entity Recognition. Table 4.5 shows reported performance results for relevant models, compared with the performance of our model.

Model	Type	ROC AUC
Physiological models		
GRU-D (Che, Purushotham, Cho et al., 2016)	Recurrent	0.8527 ± 0.003
MMDL (Purushotham et al., 2018)	Hybrid	0.8783 ± 0.0037
ISeeU	ConvNet	0.8735 ± 0.0025
NLP models		
LSTM+E+T+D (Combining LSTM and Latent Topic Modeling for Mortality PredictionJo et al., 2017)	Recurrent	< 0.84
Vital + EntityEmb (Jin et al., 2018)	Hybrid (text & physiological inputs)	0.8734 ± 0.0019
ISeeU2 (our work)	ConvNet	0.8629 ± 0.0058

Table 4.5: Our results and results reported by related works. ROC AUC results are mean and standard deviation from a 5-fold cross validation run, except LSTM+E+T+D and Vital + EntityEmb, which report a single result over the test set.

By using text notes as input we are using not the raw physiological data but health-care workers perceptions and judgement in the form of free-text notes, giving us access to higher level concepts not present in said physiological data. On the other hand, MIMIC-III notes are very noisy, with frequent misspellings, typos and a lack of standardized naming (i.e. writing *vancomycin* vs *vanc*), which makes them not an optimal learning substrate. However, we have been able to show that deep learning models are able to separate useful signals from such noise, by keeping our pre-processing pipeline very basic. Another interesting take on the usage of free text notes is that deep models can leverage meta-data such as note length, as our evidence suggest. This phenomena is comparable to observations made in (Lipton et al., 2016; Razavian & Sontag, 2015) regarding the ability of deep neural networks to exploit patterns of missingness in physiological patient data to attain better predictive performance: Certain physiological measurements are taken more or less frequently according to the state of the patient,

providing additional and useful metadata. This kind of flexibility and power is outside the reach of more traditional statistical modeling techniques as the ones behind risk scores as SAPS-II.

Our visualization approach allows to easily locate the parts of the notes the deep learning model is attending to, which can then be compared to clinical expectations. In this way the potential users can be certain that the reasons behind the predictions are sound and align properly with medical knowledge, as opposed to being evidence of statistical artifacts being leveraged by the model. It is interesting to note that our results suggest our model and text heatmap visualization could be used to annotate medical notes for, at some point, easier handling by ICU staff. Finally, it's worth noticing that our usage of Shapley Values was instrumental to discover how the network regarded the padding introduced in shorter nursing notes.

4.3 Software implementation

As part of our objectives we planned a software implementation that leveraged a deep learning model trained on MIMIC-III for interpretable mortality prediction in the ICU, and this led to the creation of ISeeU and ISeeU2. These are open source Python packages available on the Python Package index (PyPi) [2] [3], that provide the functionality developed in this research. As such, they can be installed freely on any machine running Python 3.6+ through the pip command (`pip install iseeu`, `pip install iseeu2`). Sources for the packages are available on GitHub [4] [5].

ISeeU exposes an object-oriented API comprised by the following methods:

[2] `https://pypi.org/project/iseeu/`
[3] `https://pypi.org/project/iseeu2/`
[4] `https://github.com/williamcaicedo/ISeeU`
[5] `https://github.com/williamcaicedo/ISeeU2`

- `predict(patient_tensor)` takes a `(1, 22, 48)` Numpy tensor as input and returns a mortality prediction between zero and one, and a `(22, 48)` Numpy matrix that contains the associated feature importances (approximated Shapley Values).

- `visualize_patient_scores(patient_tensor,...)` generates a Matplolib visualization similar to Figure 4.12, in which importances are shown for each individual input at different hours. If no `importance_scores` are provided, the method returns a plot of the patient data alone.

- `visualize_evidence(importance_scores=scores)` returns a barplot similar to Figure 4.13. This visualization is intended to show the total importance of each input feature group, plotting their positive and negative contributions separately.

The ISeeU implementation is completely extensible and since it makes the actual importance scores available to the end user, further visualizations and explanations can be created. ISeeU can be used in IPython Notebooks or as a part of another application. Screenshots of its use inside a Jupyter Notebook can be seen in Figures 4.24, 4.25, and 4.26

ISeeU2 also exposes an object-oriented API, with the following public methods:

- `preprocess_notes(notes=patient_notes)` takes a Pandas dataframe with a column named "text" that shall contain the medical notes to be used for prediction. This method transforms the actual note into a `(1, 500)` Numpy tensor that can be used by the `predict` method.

- `predict(patient_note_sequences=sequences)` takes a `(1, 500)` Numpy tensor as input and returns a mortality prediction between zero and one,

Figure 4.24: Visualization of patient inputs using ISeeU.

Figure 4.25: Prediction of mortality probability and visualization of hourly feature importances using ISeeU.

and a list of `(1, 500)` Numpy matrices that contain the associated word importances (approximated Shapley Values) for each provided note.

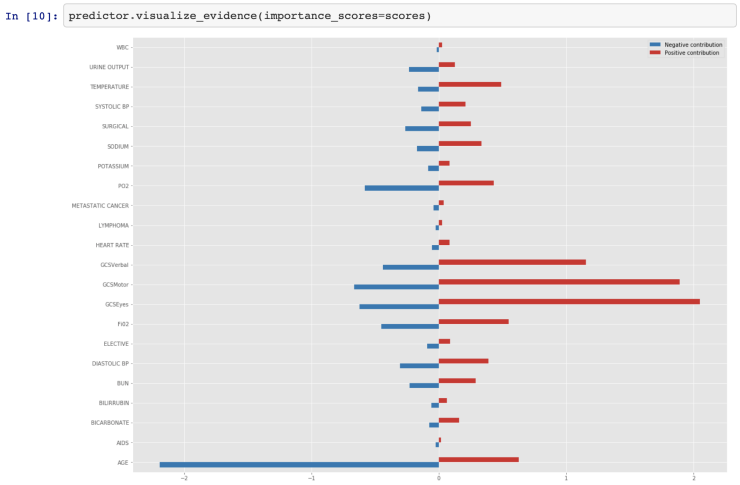


Figure 4.26: Generating the explanation of a prediction by visualizing marginal feature importances using ISeeU.

- `get_word_clouds(patient_note_sequence=sequence,...)` generates a word clouds similar to those in Figure 4.21.

- `get_note_heatmap(patient_note_sequence=sequence,...)` returns a note heatmap similar to the one in Figure 4.21. This visualization shows the importance of words in notes and allows to perform convolution smoothing over word importances for enhanced note annotation.

ISeeU2 also provides the approximated Shapley Values back for further use, and can be employed inside iPython notebooks or as part of other applications as well. Screenshots of its use inside a Jupyter Notebook can be seen in Figures 4.27, 4.28.

4.4 Conclusion

Using MIMIC-III data we have been able to train a physiological time series-based Multi-scale ConvNet and a free-text medical note-based ConvNet to predict patient mortality inside the Intensive Care Unit. Model performance shows stability across validation folds and the physiological time series model reaches state of the art performance for

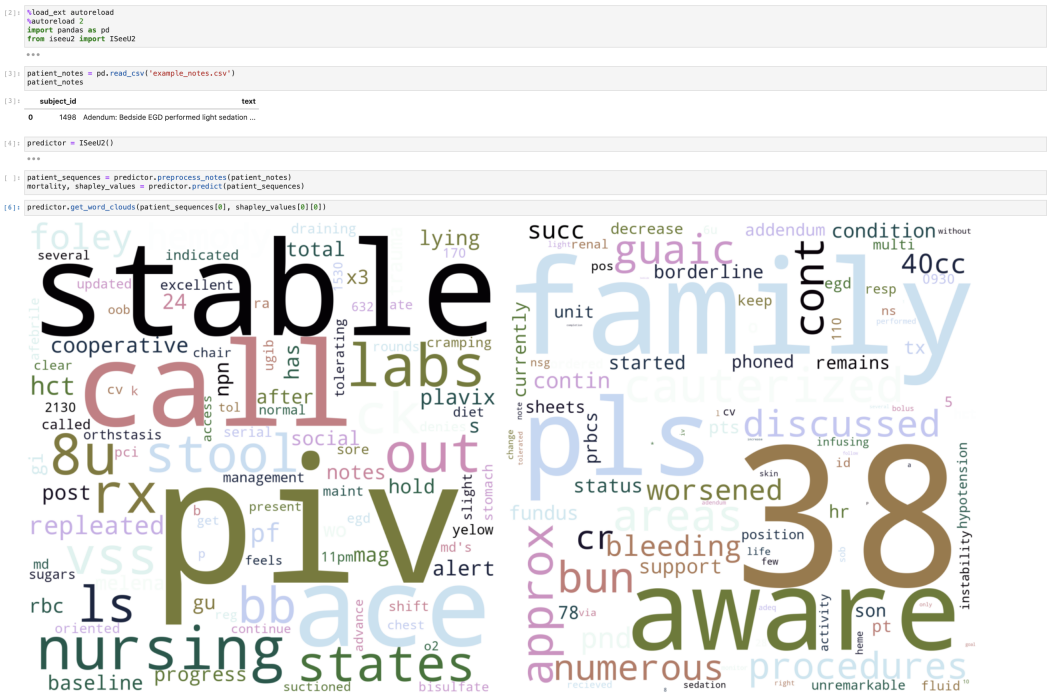


Figure 4.27: Use of ISeeU2 and generating word clouds for word importance visualization.

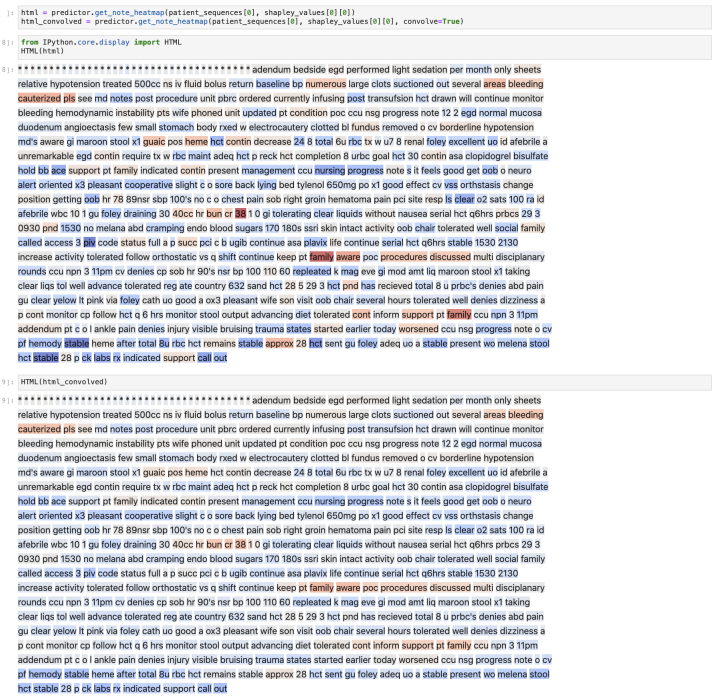


Figure 4.28: Generating note heatmaps with ISeeU2.

ICU mortality prediction using MIMIC-III and comparable feature sets. Both ConvNets have the added benefit of post-hoc interpretability without having to resort to a surrogate, interpretable model. During the development process we were able to discover evidence that corroborates certain findings by the literature, allowing us to obtain our reported performances.

In the next chapter we will consolidate our conclusions and we will discuss the implications of our findings and how they relate to the more general picture of the use of Machine Learning for mortality prediction.

Chapter 5

Conclusions and future work

5.1 Research conclusions

Quantification of the likelihood of negative outcomes in patients in Intensive Care Units is necessary for the evaluation of the effectiveness of treatments and clinical practices. To this end statistical scores have been created, with various degrees of success. These scores exhibit certain shortcomings associated to the specific choice of their underlying models, and to evolving clinical practice. In response to this, the use of more expressive models has been studied, with interesting results. Amongst alternatives, Deep Learning shows promise due to its ability to learn complex, non-linear patterns, from a wide variety of structured and non-structured data. However adoption of Deep Learning for medical tasks including mortality prediction still faces hurdles due to its reduced interpretability, which makes it difficult for doctors to trust Deep Learning models in clinical settings.

The motivation for this work was to answer a set of research questions that have been posed in order to shed light on the matter of whether it is possible to build Deep Learning models able to predict mortality inside the ICU, subject to certain constraints,

namely state-of-the-art predictive performance and the offering of explanations for the predictions of the model. For completeness, we will state the aforementioned research questions next. Our questions were:

- How can we develop a Deep Learning model able to predict, with state-of-the-art performance, survival-related outcomes in a critical care setting?
- How can we provide useful explanations for the model decisions using feasible time and computing resources, without resorting to auxiliary classification models?

Our research findings allowed to formulate adequate answers to these questions, in the form of a novel multi-scale convolutional network model for interpretable mortality prediction inside the ICU, trained on Physiological Time Series (PTS) data; and a novel convolutional network trained on free-text nursing notes. We showed that our PTS model offers state of the art performance, and our free-text model offers performance comparable to that of PTS models and overperforms similar Natural Language Processing approaches. Both models accomplish the aforementioned goals while offering visual explanations based on a concept from coalitional game theory, that express how important the inputs features are for the model's output. Such explanations are offered at the individual patient level (PTS and free-text models) with different levels of de-aggregation, and at the dataset level (PTS model), allowing for a more complete statistical understanding of how our models regard input predictors, compared to what related works have provided so far, and without resorting to auxiliary or surrogate models. We were able to show that

- Deep Learning offers predictive performance superior to that of traditional statistical techniques for mortality prediction inside the ICU, as attested by results on MIMIC-III data.

- A convolutional model can handle both temporal and static features at the same time without having to resort to recurrent and/or hybrid neural architectures, while at the same time offering comparable performance with specialized recurrent architectures.

- Simple imputation techniques offer competitive performance for ICU mortality prediction with Deep Learning, without incurring in the computational costs associated with more complex approaches.

- A convolutional model trained on free-text notes from MIMIC-III offers competitive performance to that from PTS models.

- The usage of nursing notes alone offers good predictive performance, compared to approaches that use the full gamut of medical notes available on MIMIC-III.

Additionally, our experimental framework allowed us to uncover interesting insights, as the following:

- Adaptive optimization algorithms like ADAM tend to be more stable and easier to use, but they are still not able to match the performance of Stochastic Gradient Descent (SGD) (given that SGD maintains stability). Even if the training loss achieved by both is similar, the generalization gap is smaller for SGD. Our results support the usual practice of using adaptive methods first to rapidly iterate over predictors and architectures, and using SGD later (if possible) to get the best possible performance.

- Even if Max Pooling is the dominant approach for subsampling, it is a good idea to try Average Pooling as well and compare results. In our experiments, we found a performance increase of approximately 2% in ROC AUC by switching

from Max Pooling to Average Pooling. As with many things in Machine Learning, particularities of datasets end up determining the performance of modeling techniques and algorithms on a case-by-case basis.

- Deep Learning models are able to leverage training meta-data. In particular, the length of nursing notes was found to be a feature used by our convolutional network in order to predict mortality. Padding characters used at the beginning of the notes after preprocessing are regarded by our model as evidence for survival, as shown by their associated Shapley Values.

Finally it is worth noting that, with the models developed as part of our research we have created ISeeU and ISeeU2, two open source programming libraries freely available for researchers, meant as a stepping stone to kick-start efforts that could eventually lead to the creation of software solutions able to be used by ICU medical practitioners.

5.2 Limitations

Limitations of our study include the fact that we do not have access to pre-admission data, and that we are using a retrospective, single center cohort. Also given the moderate size of our dataset we are only reporting cross-validation results without a proper test set result. An additional limitation is that although we have strived to include common parameters and variables, it is still possible that some of them are not routinely recorded for a non-trivial group of patients, and our approach would suffer from elevated rates of missing data. An additional limitation is that high-quality nursing notes may not be available for a substantial number of patients in other critical care settings, which could hurt the performance of our Natural Language Processing model. Finally, the common misspellings and other noise present in the medical notes used to train our free-text model may affect the quality of our explanations, giving rise to counterintuitive results.

Despite these limitations, the results presented in this thesis support the feasibility of using Deep Learning for ICU mortality.

5.2.1 Threats to Validity

Following the Campbell tradition (Matthay & Glymour, 2020), we identify the following threats to validity of the work we have presented in this thesis:

- Internal Threats to Validity: Selection bias as a product of our patient inclusion criteria and choice of dataset (selection), and software defects in the computational tools developed in this thesis (instrumentation).
- External Threats to Validity: Results on MIMIC-III cohort may not be generalizable to other ICU populations due to unknown confounders or differing patient distributions (interaction of the causal relationship with units).

To manage the internal threats we have followed best-practices and a methodology that is in line with the relevant literature. In the case of the external threat, more research involving different ICU datasets is needed to validate our findings.

5.3 Future work

This work sets the stage for a series of interesting questions/projects. Some of them are

- One interesting question is whether we can leverage transfer learning in the ICU domain, i.e. use MIMIC-III pre-trained models and supervised fine-tuning to predict mortality in institutions with limited data. Since transfer learning has been used in the Computer Vision and Natural Language Processing domains with great success, it is reasonable to assume similar success can be enjoyed in

ICU mortality prediction, but this needs to be properly studied and validated in multi-center, prospective studies.

- The use of Shapley Values for feature selection is another interesting research direction this thesis sets up. Our results indicate that certain variables from the SAPS-II set are not being attended by our model to predict mortality, across the entire dataset. This is remarkable because SAPS-II (and similar scores) represent the domain expertise of physicians and health professionals. What does this mean? Can we use Shapley Values to inform traditional scores as well as to select features and improve our model's performance? Also, due to our interpretability work we have found that note length is helpful to predict mortality. Does this hold in different centers/settings? Studying this correlation more in depth can provide an interesting direction for future work.

- One particular strength of Deep Learning is the ability of Neural Network to leverage different modalities of data for predictive purposes, and our work is an example of that. An interesting question to be tackled by future work is whether PTS and free-text data can be used efficiently by a single model as inputs, and whether this hybrid model could over-perform the individual models or an ensemble of the two. Can we incorporate other modes of data, such as diagnostic images, as well?

- Other works have shown the regularization effects of multi-task learning. Evaluating the impact in terms of validation/test performance of jointly training our models to predict not only mortality but also Length of Stay would be a very interesting proposition. Accurate prediction of Length of Stay is an open problem and a pressing need for the efficient planning and management of ICU capacity.

- The question of algorithmic fairness in Machine Learning for healthcare starts

to become a major concern. Are our predictive models offering consistent performance across diverse ethnicities, demographics and cultures? Using our current research and development practices, is society at large reaping the benefits equally? are our datasets diverse enough? if not, what solutions can we devise?

- Finally, the dilemma of more accurate vs more timely predictions deserves to be studied more in depth. Our experimental results show that the usage of 48-hour data leads to better predictions, however predictions at the 24-hour mark are more useful. How is this trade-off relevant for individual institutions and medical practice, and what can we do to improve results at the 24-hour mark are questions worth exploring. The question of whether it is feasible to convert our approach into an early warning system, able to provide real-time mortality prediction is worth exploring as well.

References

Andreasen, N. C. (1984). Scale for the Assessment of Positive Symptoms (SAPS). *The British Journal of Psychiatry. Supplement*.

Ba, L. J. & Caruana, R. (2014). Do Deep Nets Really Need to Be Deep? In *Proceedings of the 27th international conference on neural information processing systems* (pp. 2654–2662). Cambridge, MA, USA: MIT Press. Retrieved from http://dl.acm.org/citation.cfm?id=2969033.2969123

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K. R. & Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*. doi: 10.1371/journal.pone.0130140

Bahdanau, D., Cho, K. & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. In *Iclr 2015*.

Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends{®} in Machine Learning*, *2*(1), 1–127. Retrieved from http://dx.doi.org/10.1561/2200000006 doi: 10.1561/2200000006

Bengio, Y., Lamblin, P., Popovici, D. & Larochelle, H. (2006). Greedy Layer-wise Training of Deep Networks. In *Proceedings of the 19th international conference on neural information processing systems* (pp. 153–160). Cambridge, MA, USA: MIT Press. Retrieved from http://dl.acm.org/citation.cfm?id=2976456.2976476

Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993–1022.

Boureau, Y.-L., Ponce, J. & LeCun, Y. (2010). A Theoretical Analysis of Feature Pooling in Visual Recognition. In *The 27th international conference on machine learning*. doi: 10.1.1.170.864

Caicedo-Torres, W. & Gutierrez, J. (2019, oct). ISeeU: Visually interpretable deep learning for mortality prediction inside the ICU. *Journal of Biomedical Informatics*, *98*, 103269. Retrieved from https://www.sciencedirect.com/science/article/pii/S1532046419301881?dgcid=author doi: 10.1016/J.JBI.2019.103269

Calvert, J., Mao, Q., Hoffman, J. L., Jay, M., Desautels, T., Mohamadlou, H., … Das, R. (2016). Using electronic health record collected clinical variables to predict medical intensive care unit mortality. *Annals of Medicine and Surgery*. doi: 10.1016/j.amsu.2016.09.002

Che, Z., Kale, D., Li, W., Bahadori, M. T. & Liu, Y. (2015). Deep Computational

Phenotyping. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining* (pp. 507–516). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/2783258.2783365 doi: 10.1145/2783258.2783365

Che, Z., Purushotham, S., Cho, K., Sontag, D. & Liu, Y. (2016). Recurrent Neural Networks for Multivariate Time Series with Missing Values. *CoRR*, *abs/1606.0*. Retrieved from http://arxiv.org/abs/1606.01865

Che, Z., Purushotham, S., Khemani, R. & Liu, Y. (2016). Interpretable Deep Models for ICU Outcome Prediction. *AMIA Annual Symposium Proceedings*, *2016*, 371–380. Retrieved from http://www.ncbi.nlm.nih.gov/pmc/articles/PMC5333206/

Cho, K., Courville, A. & Bengio, Y. (2015, nov). Describing Multimedia Content Using Attention-Based Encoder-Decoder Networks. *IEEE Transactions on Multimedia*, *17*(11), 1875–1886. doi: 10.1109/TMM.2015.2477044

Cho, K., van Merrienboer, B., Bahdanau, D. & Bengio, Y. (2014). On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *CoRR*, *abs/1409.1*. Retrieved from http://arxiv.org/abs/1409.1259

Choi, E., Bahadori, M. T., Schuetz, A., Stewart, W. F. & Sun, J. (2016). Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine learning for healthcare conference* (pp. 301–318).

Choi, E., Bahadori, M. T., Song, L., Stewart, W. F. & Sun, J. (2016). {GRAM:} Graph-based Attention Model for Healthcare Representation Learning. *CoRR*, *abs/1611.0*. Retrieved from http://arxiv.org/abs/1611.07012

Choi, E., Bahadori, M. T., Sun, J., Kulas, J., Schuetz, A. & Stewart, W. (2016). RETAIN: An Interpretable Predictive Model for Healthcare using Reverse Time Attention Mechanism. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon & R. Garnett (Eds.), *Advances in neural information processing systems 29* (pp. 3504–3512). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/6321-retain-an-interpretable-predictive-model-for-healthcare-using-reverse-time-attention-mechanism.pdf

Chollet, F. (2015). *Keras*. doi: 10.1016/j.it.2007.05.003

Chung, J., Gülçehre, Ç., Cho, K. & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR*, *abs/1412.3*.

Combining LSTM and Latent Topic Modeling for Mortality PredictionJo, Y., Lee, L. & Palaskar, S. (2017). Combining LSTM and Latent Topic Modeling for Mortality Prediction. *ArXiv*, *abs/1709.0*.

Cooper, G. F., Aliferis, C. F., Ambrosino, R., Aronis, J., Buchanan, B. G., Caruana, R., ... Spirtes, P. (1997). An evaluation of machine-learning methods for predicting pneumonia mortality. *Artificial Intelligence in Medicine*. doi: 10.1016/S0933-3657(96)00367-3

Crevier, D. (1993). *AI: The Tumultuous History of the Search for Artificial Intelligence*. New York, NY, USA: Basic Books, Inc.

El Hihi, S. & Bengio, Y. (1995). Hierarchical Recurrent Neural Networks for Long-term Dependencies. In *Proceedings of the 8th international conference on neural information processing systems* (pp. 493–499). Cambridge, MA, USA: MIT Press. Retrieved from http://dl.acm.org/citation.cfm?id=2998828.2998898

Elkan, C. & Greiner, R. (1993). Building large knowledge-based systems: Representation and inference in the cyc project. *Artificial Intelligence*, *61*(1), 41–52. Retrieved from http://www.sciencedirect.com/science/article/pii/000437029390092P doi: http://dx.doi.org/10.1016/0004-3702(93)90092-P

Esteban, C., Staeck, O., Baier, S., Yang, Y. & Tresp, V. (2016). Predicting clinical events by combining static and dynamic information using recurrent neural networks. In *Healthcare informatics (ichi), 2016 ieee international conference on* (pp. 93–101).

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Gall, J. R., Lemeshow, S. & Saulnier, F. (1993). A New Simplified Acute Physiology Score (SAPS II) Based on a European/North American Multicenter Study. *JAMA: The Journal of the American Medical Association*. doi: 10.1001/jama.1993.03510240069035

Ge, W., Huh, J. W., Park, Y. R., Lee, J. H., Kim, Y. H. & Turchin, A. (2018). An Interpretable ICU Mortality Prediction Model Based on Logistic Regression and Recurrent Neural Networks with LSTM units. *AMIA ... Annual Symposium proceedings. AMIA Symposium*.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning*. MIT Press.

Grnarova, P., Schmidt, F., Hyland, S. L. & Eickhoff, C. (2016). Neural Document Embeddings for Intensive Care Patient Mortality Prediction. *CoRR*, *abs/1612.0*.

He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. doi: 10.1109/CVPR.2016.90

Hinton, G. E., Osindero, S. & Teh, Y.-W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, *18*(7), 1527–1554. Retrieved from http://dx.doi.org/10.1162/neco.2006.18.7.1527 doi: 10.1162/neco.2006.18.7.1527

Hochreiter, S. & Schmidhuber, J. (1997a). Flat minima. *Neural Computation*. doi: 10.1162/neco.1997.9.1.1

Hochreiter, S. & Schmidhuber, J. (1997b, nov). Long Short-Term Memory. *Neural Comput.*, *9*(8), 1735–1780. Retrieved from http://dx.doi.org/10.1162/neco.1997.9.8.1735 doi: 10.1162/neco.1997.9.8.1735

Huang, Y. C., Chang, K. Y., Lin, S. P., Chen, K., Chan, K. H. & Chang, P. (2013). Development of a daily mortality probability prediction model from Intensive Care Unit patients using a discrete-time event history analysis. *Computer Methods and Programs in Biomedicine*, *111*(2), 280–289. Retrieved from http://www.sciencedirect.com/science/article/

pii/S0169260713001107 doi: http://dx.doi.org/10.1016/j.cmpb.2013.03.018

Hyvärinen, A. & Smith, S. M. (2013, jan). Pairwise Likelihood Ratios for Estimation of non-Gaussian Structural Equation Models. *J. Mach. Learn. Res.*, *14*(1), 111–152. Retrieved from http://dl.acm.org/citation.cfm?id=2502581.2502585

Ioffe, S. & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Arxiv*, *abs/1502.0*, 1–11. Retrieved from http://arxiv.org/abs/1502.03167 doi: 10.1007/s13398-014-0173-7.2

Jin, M., Bahadori, M. T., Colak, A., Bhatia, P., Celikkaya, B., Bhakta, R., ... Kass-hout, T. (2018). *Improving Hospital Mortality Prediction with Medical Named Entities and Multimodal Learning*.

Johnson, A. E., Stone, D. J., Celi, L. A. & Pollard, T. J. (2018). The MIMIC Code Repository: Enabling reproducibility in critical care research. *Journal of the American Medical Informatics Association*. doi: 10.1093/jamia/ocx084

Johnson, A. E. W., Ghassemi, M. M., Nemati, S., Niehaus, K. E., Clifton, D. & Clifford, G. D. (2016). Machine Learning and Decision Support in Critical Care. *Proceedings of the IEEE*, *104*(2), 444–466. Retrieved from http://ieeexplore.ieee.org/document/7390351/ doi: 10.1109/JPROC.2015.2501978

Johnson, A. E. W., Pollard, T. J., Shen, L., Lehman, L.-W. H., Feng, M., Ghassemi, M., ... Mark, R. G. (2016, may). MIMIC-III, a freely accessible critical care database. *Sci Data*, *3*, 160035. doi: 10.1038/sdata.2016.35

Kale, D. C., Che, Z., Bahadori, M. T., Li, W., Liu, Y. & Wetzel, R. (2015). Causal Phenotype Discovery via Deep Networks. *AMIA Annual Symposium Proceedings*, *2015*, 677–686. Retrieved from http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4765623/

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M. & Tang, P. T. P. (2017). On Large Batch Training for Deep Learning. In *Iclr*.

Khemani, R. G., Conti, D., Alonzo, T. A., Bart, R. D. r. & Newth, C. J. L. (2009, aug). Effect of tidal volume in children with acute hypoxemic respiratory failure. *Intensive Care Med*, *35*(8), 1428–1437. doi: 10.1007/s00134-009-1527-z

Kindermans, P.-J., Schütt, K., Müller, K.-R. & Dähne, S. (2016, nov). Investigating the influence of noise and distractors on the interpretation of neural networks. *ArXiv e-prints*.

Kingma, D. & Ba, J. (2014, dec). Adam: A Method for Stochastic Optimization. *ArXiv e-prints*.

Knaus, W. A., Draper, E. A., Wagner, D. P. & Zimmerman, J. E. (1985). APACHE II: A severity of disease classification system. *Critical Care Medicine*. doi: 10.1097/00003246-198510000-00009

Knaus, W. A., Wagner, D. P., Draper, E. A., Zimmerman, J. E., Bergner, M., Bastos, P. G., ... Harrell, F. E. (1991). The APACHE III prognostic system: Risk prediction of hospital mortality for critically III hospitalized adults. *Chest*. doi:

10.1378/chest.100.6.1619

Knaus, W. A., Zimmerman, J. E., Wagner, D. P., Draper, E. A. & Lawrence, D. E. (1981). APACHE-acute physiology and chronic health evaluation: a physiologically based classification system. *Critical care medicine*. doi: 10.1097/00003246-198108000-00008

Lasko, T. A., Denny, J. C. & Levy, M. A. (2013). Computational Phenotype Discovery Using Unsupervised Feature Learning over Noisy, Sparse, and Irregular Clinical Data. *PLoS ONE*, *8*(6), e66341. Retrieved from http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3691199/ doi: 10.1371/journal.pone.0066341

Lau, J. H. & Baldwin, T. (2016). An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation. *CoRR*, *abs/1607.0*. Retrieved from http://arxiv.org/abs/1607.05368

LeCun, Y., Bottou, L., Bengio, Y. & Haffner. (1998). Gradient-Based Learning Applied to Document Recognition. In *Proceedings of the ieee* (Vol. 86, pp. 2278–2324).

Liang, Z., Zhang, G., Huang, J. X. & Hu, Q. V. (2014). Deep learning for healthcare decision making with EMRs. *2014 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 556–559. doi: 10.1109/BIBM.2014.6999219

Lindsay, R. K., Buchanan, B. G., Feigenbaum, E. A. & Lederberg, J. (1993). DENDRAL: A case study of the first expert system for scientific hypothesis formation. *Artificial Intelligence*, *61*(2), 209–261. Retrieved from http://www.sciencedirect.com/science/article/pii/000437029390068M doi: http://dx.doi.org/10.1016/0004-3702(93)90068-M

Lipton, Z. C. (2016). The Mythos of Model Interpretability. *ICML Workshop on Human Interpretability in Machine Learning*, *abs/1606.0*, 96–100. Retrieved from http://arxiv.org/abs/1606.03490

Lipton, Z. C., Kale, D. & Wetzel, R. (2016). Directly Modeling Missing Data in Sequences with RNNs: Improved Classification of Clinical Time Series. In F. Doshi-Velez, J. Fackler, D. Kale, B. Wallace & J. Weins (Eds.), *Proceedings of the 1st machine learning for healthcare conference* (Vol. 56, pp. 253–270). Northeastern University, Boston, MA, USA: PMLR. Retrieved from http://proceedings.mlr.press/v56/Lipton16.html

Lipton, Z. C., Kale, D. C., Elkan, C. & Wetzell, R. (2015). Learning to Diagnose with LSTM Recurrent Neural Networks. In *International conference on learning representations (iclr 2016)*.

Loper, E. & Bird, S. (2002). NLTK.. doi: 10.3115/1118108.1118117

Lundberg, S. M. & Lee, S.-I. (2017a). A Unified Approach to Interpreting Model Predictions. In I. Guyon et al. (Eds.), *Advances in neural information processing systems 30* (pp. 4765–4774). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

Lundberg, S. M. & Lee, S. I. (2017b). A unified approach to interpreting model predictions. In *Advances in neural information processing systems*.

Matthay, E. C. & Glymour, M. M. (2020). A Graphical Catalog of Threats to Validity:

Linking Social Science with Epidemiology. *Epidemiology*. doi: 10.1097/EDE.0000000000001161

McCulloch, W. S. & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), 115–133.

Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *The Workshop Proceedings of the International Conference on Learning Representations (ICLR)*, *abs/1301.3*. Retrieved from http://arxiv.org/abs/1301.3781

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J., Chen, K., ... Chen, K. (2013). Distributed Representations of Words and Phrases and their Compositionality. In *Nips'14* (Vol. cs.CL, pp. 3111–3119). USA: Curran Associates Inc. Retrieved from http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf{%}5Cnhttp://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality doi: 10.1162/jmlr.2003.3.4-5.951

Minsky, M. L. & Papert, S. (1972). *Perceptrons: An Introduction to Computational Geometry*. Mit Press. Retrieved from https://books.google.co.nz/books?id=Ow1OAQAAIAAJ

Miotto, R., Li, L., Kidd, B. A. & Dudley, J. T. (2016). Deep Patient: An Unsupervised Representation to Predict the Future of Patients from the Electronic Health Records. *Scientific Reports*, *6*, 26094 EP –. Retrieved from http://dx.doi.org/10.1038/srep26094

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill Education. Retrieved from https://books.google.co.nz/books?id=xOGAngEACAAJ

Moreno, R. P., Metnitz, P. G., Almeida, E., Jordan, B., Bauer, P., Campos, R. A., ... Le Gall, J. R. (2005). SAPS 3 - From evaluation of the patient to evaluation of the intensive care unit. Part 2: Development of a prognostic model for hospital mortality at ICU admission. *Intensive Care Medicine*. doi: 10.1007/s00134-005-2763-5

Mozer, M. C. (1992). The Induction of Multiscale Temporal Structure. In J. E. Moody, S. J. Hanson & R. P. Lippmann (Eds.), *Advances in neural information processing systems, 4* (pp. 275–282). San Mateo, CA: Morgan Kaufmann.

Nadaraya, E. A. (1964). On Estimating Regression. *Theory of Probability & Its Applications*, *9*(1), 141–142. Retrieved from http://dx.doi.org/10.1137/1109020 doi: 10.1137/1109020

Nesterov, Y. E. (1983). A Method of Solving A Convex Programming Problem with Convergence Rate O(1/k2). *Soviet Mathematics Doklady*.

Nguyen, P., Tran, T., Wickramasinghe, N. & Venkatesh, S. (2017). Deepr: A Convolutional Net for Medical Records. *IEEE Journal of Biomedical and Health Informatics*, *21*(1), 22–30. doi: 10.1109/JBHI.2016.2633963

Paul, E., Bailey, M. & Pilcher, D. (2013). Risk prediction of hospital mortality for adult patients admitted to Australian and New Zealand intensive

care units: Development and validation of the Australian and New Zealand Risk of Death model. *Journal of Critical Care*, *28*(6), 935–941. Retrieved from http://www.sciencedirect.com/science/article/pii/S0883944113002566 doi: http://dx.doi.org/10.1016/j.jcrc.2013.07.058

Pham, T., Tran, T., Phung, D. & Venkatesh, S. (2016). DeepCare: A Deep Dynamic Memory Model for Predictive Medicine. In *Proceedings, part ii, of the 20th pacific-asia conference on advances in knowledge discovery and data mining - volume 9652* (pp. 30–41). New York, NY, USA: Springer-Verlag New York, Inc. Retrieved from http://dx.doi.org/10.1007/978-3-319-31750-2{_}3 doi: 10.1007/978-3-319-31750-2_3

Purushotham, S., Meng, C., Che, Z. & Liu, Y. (2018). Benchmarking Deep Learning Models on Large Healthcare Datasets. *Journal of Biomedical Informatics*. Retrieved from http://www.sciencedirect.com/science/article/pii/S1532046418300716 doi: https://doi.org/10.1016/j.jbi.2018.04.007

Rapsang, A. G. & Shyam, D. C. (2014). Scoring systems in the intensive care unit: A compendium. *Indian Journal of Critical Care Medicine : Peer-reviewed, Official Publication of Indian Society of Critical Care Medicine*, *18*(4), 220–228. Retrieved from http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4033855/ doi: 10.4103/0972-5229.130573

Rasmussen, C. E. (2006). Gaussian processes for machine learning. MIT Press.

Razavian, N., Marcus, J. & Sontag, D. (2016). Multi-task Prediction of Disease Onsets from Longitudinal Lab Tests. *Machine Learning in Health Care*, *abs/1608.0*, 1–27. Retrieved from http://arxiv.org/abs/1608.00647

Razavian, N. & Sontag, D. (2015). Temporal convolutional neural networks for diagnosis lab tests. *25 November*, *abs/1511.0*, 1–17. Retrieved from http://arxiv.org/abs/1511.07938 doi: 10.1051/0004-6361/201527329

Rosenblatt, F. (1957). *The Perceptron, a Perceiving and Recognizing Automaton Project Para*. Cornell Aeronautical Laboratory. Retrieved from https://books.google.co.nz/books?id=P{_}XGPgAACAAJ

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*. doi: 10.1038/323533a0

Russell, S. J., Norvig, P. & Davis, E. (2016). *Artificial Intelligence: A Modern Approach*. Pearson. Retrieved from https://books.google.co.nz/books?id=XS9CjwEACAAJ

Saeed, M., Villarroel, M., Reisner, A. T., Clifford, G., Lehman, L.-W., Moody, G., ... Mark, R. G. (2011). Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II): a public-access intensive care unit database. *Critical care medicine*, *39*(5), 952.

Shapiro, L. (2001). *Computer vision*. Upper Saddle River, NJ: Prentice Hall.

Shapley, L. S. (1953). A Value for n-Person Games. In H. W. Kuhn & A. W. Tucker (Eds.), *Contributions to the theory of games ii* (pp. 307–317). Princeton: Princeton University Press.

Shen, D., Wu, G. & Suk, H.-I. (2017). Deep Learning in Medical Image Analysis.

Annual Review of Biomedical Engineering, *19*(1), null. Retrieved from http://dx.doi.org/10.1146/annurev-bioeng-071516-044442 doi: 10.1146/annurev-bioeng-071516-044442

Shortliffe, E. H. (1977). Mycin: A Knowledge-Based Computer Program Applied to Infectious Diseases. *Proceedings of the Annual Symposium on Computer Application in Medical Care*, 66–69. Retrieved from http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2464549/

Shrikumar, A., Greenside, P. & Kundaje, A. (2017). Learning Important Features Through Propagating Activation Differences. *CoRR*, *abs/1704.0*. Retrieved from http://arxiv.org/abs/1704.02685

Si, Y. & Roberts, K. (2019). Deep Patient Representation of Clinical Notes via Multi-Task Learning for Mortality Prediction. *AMIA Joint Summits on Translational Science proceedings. AMIA Joint Summits on Translational Science*.

Silva, I., Moody, G., Scott, D. J., Celi, L. A. & Mark, R. G. (2012). Predicting In-Hospital Mortality of ICU Patients: The PhysioNet/Computing in Cardiology Challenge 2012. *Computing in cardiology*, *39*, 245–248. Retrieved from http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3965265/

Simonyan, K., Vedaldi, A. & Zisserman, A. (2013). Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *CoRR*, *abs/1312.6*. Retrieved from http://arxiv.org/abs/1312.6034

Springenberg, J. T., Dosovitskiy, A., Brox, T. & Riedmiller, M. A. (2014). Striving for Simplicity: The All Convolutional Net. *CoRR*, *abs/1412.6*. Retrieved from http://arxiv.org/abs/1412.6806

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014, jan). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, *15*(1), 1929–1958. Retrieved from http://dl.acm.org/citation.cfm?id=2627435.2670313

Strumbelj, E., Kononenko, I. & Wrobel, S. (2010). An Efficient Explanation of Individual Classifications using Game Theory. *Journal of Machine Learning Research*. doi: 10.1145/2858036.2858529

Suresh, H., Hunt, N., Johnson, A., Celi, L. A., Szolovits, P. & Ghassemi, M. (2017). Clinical Intervention Prediction and Understanding using Deep Networks. *arXiv preprint arXiv:1705.08498*.

Sushil, M., Šuster, S., Luyckx, K. & Daelemans, W. (2018). Patient representation learning and interpretable evaluation using clinical notes. *Journal of Biomedical Informatics*. doi: 10.1016/j.jbi.2018.06.016

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015). Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1–9. doi: 10.1109/CVPR.2015.7298594

Theano Development Team. (2016). Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, 19. Retrieved from http://arxiv.org/abs/1605.02688

Tieleman, T., Hinton, G. E., Srivastava, N. & Swersky, K. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA:*

Neural Networks for Machine Learning.

Urban, G., Geras, K. J., Kahou, S. E., Aslan, O., Wang, S., Caruana, R., ... Richardson, M. (2017). Do Deep Convolutional Nets Really Need to be Deep (Or Even Convolutional)? In *Iclr*.

USENIX Association., M., ACM SIGMOBILE., P., ACM Special Interest Group in Operating Systems., J., ACM Digital Library., Z., Davis, A., Dean, J., ... Xiaoqiang˜Zheng (2015). *{TensorFlow}: Large-Scale Machine Learning on Heterogeneous Systems*.

Van Den Oord, A., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., ... Hassabis, D. (2018). Parallel WaveNet: Fast high-fidelity speech synthesis. In *35th international conference on machine learning, icml 2018*.

van der Maaten, L. & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(Nov), 2579–2605.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*.

Vincent, J. L., Moreno, R., Takala, J., Willatts, S., De Mendonça, A., Bruining, H., ... Thijs, L. G. (1996). The SOFA (Sepsis-related Organ Failure Assessment) score to describe organ dysfunction/failure. *Intensive Care Medicine*. doi: 10.1007/BF01709751

Vincent, P., Larochelle, H., Bengio, Y. & Manzagol, P.-A. (2008). Extracting and Composing Robust Features with Denoising Autoencoders. In (pp. 1096–1103).

Watson, G. S. (1964). Smooth Regression Analysis. *Sankhy{\=a}: The Indian Journal of Statistics, Series A (1961-2002)*, *26*(4), 359–372. Retrieved from http://www.jstor.org/stable/25049340

Weinberger, K. Q., Sha, F., Zhu, Q. & Saul, L. K. (2007). Graph Laplacian regularization for large-scale semidefinite programming. *Advances in neural information processing systems*, *19*, 1489.

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N. & Recht, B. (2017). The Marginal Value of Adaptive Gradient Methods in Machine Learning arXiv : 1705 . 08292v2 [stat . ML] 22 May 2018. In *31st conference on neural information processing systems (nips*.

Zeiler, M. D. & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. In D. Fleet, T. Pajdla, B. Schiele & T. Tuytelaars (Eds.), *Computer vision –eccv 2014: 13th european conference, zurich, switzerland, september 6-12, 2014, proceedings, part i* (pp. 818–833). Cham: Springer International Publishing. Retrieved from http://dx.doi.org/10.1007/978-3-319-10590-1{_}53 doi: 10.1007/978-3-319-10590-1{_}53

Zhang, Z., Zou, Y. & Gan, C. (2018). Textual sentiment analysis via three different attention convolutional neural networks and cross-modality consistent regression. *Neurocomputing*. doi: 10.1016/j.neucom.2017.09.080

Zimmerman, J. E., Kramer, A. A., McNair, D. S. & Malila, F. M. (2006). Acute Physiology and Chronic Health Evaluation (APACHE) IV: Hospital mortality assessment for today's critically ill patients. *Critical Care Medicine*. doi: 10.1097/

01.CCM.0000215112.84523.F0

Appendix A

Glossary

Activation function In Neural Networks, a mathematical function that defines the output of a neuron.

AIDS Acquired Immuno-Deficiency Syndrome, a chronic viral disease that targets the human immune system.

Area Under the Receiver Operating Characteristic Curve A classification metric that represents the probability that a classifier will assign a higher score to an randomly chosen instance of the positive class than to a randomly chosen instance of the negative class.

Batch Normalization A Machine Learning technique that speeds up the training process by enforcing the normalization of the output of every hidden layer in a neural network.

Bicarbonate Byproduct of body metabolism, it is exhaled as carbon dioxide by the lungs.

Bilirubin Byproduct of the breakdown of red cells.

Bun Blood Urea Nitrogen, a bio-marker related to kidney.

Creatinine Waste product produced by the muscles of the body.

Cross-validation A resampling technique to estimate the out-of sample performance

of Machine Learning models when a test set is not available.

Dopamine A type of hormone and neurotransmitter.

Dropout A regularization technique widely used for Deep Neural Network fitting where connections inside the networks are zeroed randomly at training time.

Empirical distribution Probability distribution derived from an observed dataset.

Epinephrine Hormone and neurotransmitter produced by the adrenal glands.

Feed-forward architecture A type of Neural Network that does not have feedback loops.

First order logic An extension to propositional logic that admits the use of variables and quantifiers.

FiO_2 Fraction of inspired oxygen, the fraction of oxygen present in a gas being inspired.

Game theory Study and modeling of interactions between rational decision makers.

Glasgow Coma Scale A scoring system to quantify the level of consciousness of a person following traumatic brain injury.

ICD-9 International Classification of Diseases version 9, a system of assigning codes to diagnoses and medical procedures inside hospitals.

Intensive Care Unit Special hospital facility that provides critical care medicine.

Interpretability The degree to which the decisions made by a Machine Learning model can be made sense of by humans.

Jupyter Notebook Application that allows the creation of executable files called notebooks, that combine code, visualizations and narrative text.

Kernel Density Estimation Non-parametric probability density estimation procedure.

Length of Stay Time spent by a in-patient in a healthcare facility.

Lymphoma A type of cancer of the blood.

Matplotlib A Python visualization and graphics library.

Misspecification Situation in which a statistical model posited by a researcher does

not correspond to the 'true' model being sought.

Neural Network A class of Machine Learning loosely inspired by the central nervous system of vertebrates.

Norepinephrine A type of hormone and neurotransmitter.

Numpy A Python library for fast array manipulation.

Overfitting A detrimental state of Machine Learning models in which the training set has been fit excessively well, leading to bad performance in the training set.

Pandas A Python data analysis library.

Arterial pH Measure of acidity (amount of hydrogen ions) of the blood.

PO_2 Partial pressure of oxygen, the amount of oxygen gas dissolved in the blood.

Precision healthcare Approach to medicine in which the individual variability of each patient is taken into account to tailor treatment.

Principal Component Analysis A dimensionality reduction technique that transforms a dataset to a new coordinate system of fewer dimensions, while retaining most of the variability in the original data.

Prior Short for prior belief or prior probability distribution.

Propositional logic A branch of logic that deals with statements, the relationships between them and their truth values.

Rectified Linear Unit (ReLU) A type of non-linear activation function, widely used in Deep Learning models.

Recurrent Neural Network A specialized type of Neural Network that uses feedback loops to deal with temporal

Regularization A technique in Machine Learning to control overfitting.

Sensitivity Also known as Recall, the fraction of actual positives correctly identified by a classifier.

Specificity The fraction of actual negatives correctly identified by a classifier.

Underfitting A detrimental state of Machine Learning models in which the training

set has not been fit sufficiently well.

White cell Blood Count Concentration of white blood cells. White blood cells are specialized cells that help the body fight infections. dependencies in the input.

Appendix B

Library Usage

B.1 ISeeU

B.1.1 Visualize patient data

```
1 from iseeu import ISeeU
2 import numpy as np
3 #patient data must be stored in a numpy array
4 patient = np.load('patient.npy')
5 patient.shape
6 #(1, 22, 48)
7 predictor = ISeeU()
8 predictor.visualize_patient_scores(patient_tensor=patient
    ↪ )
9 #see figure B.1
```

B.1.2 Predict patient mortality

```
1 prediction, scores = predictor.predict(patient)
```

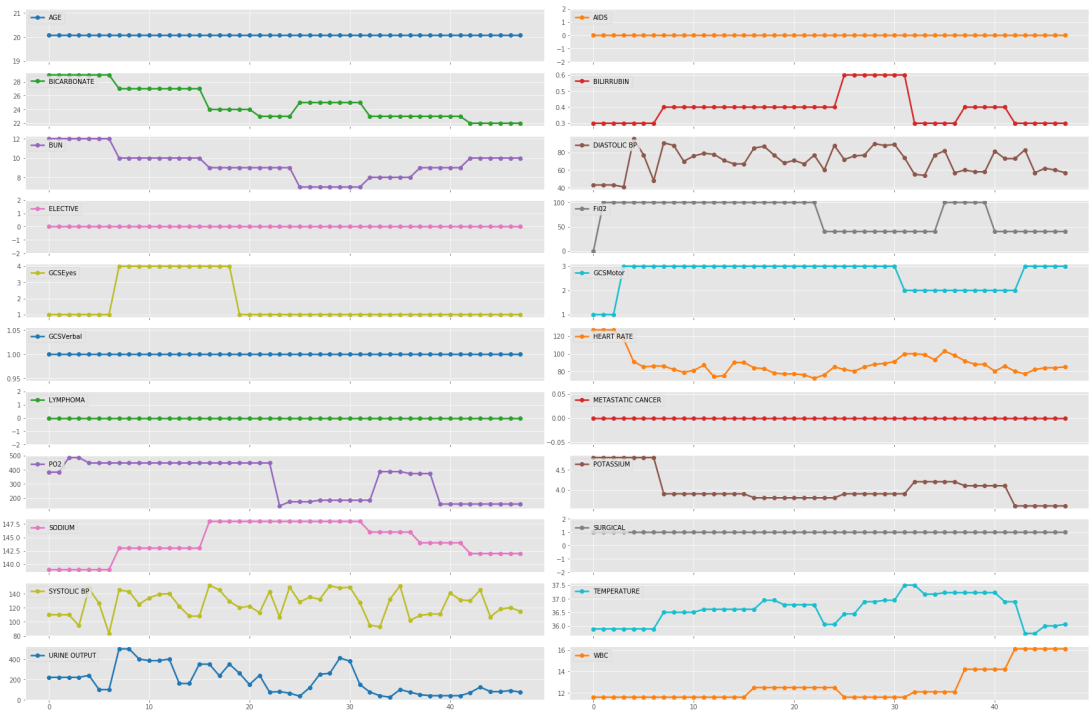


Figure B.1: Visualizing patient data with ISeeU.

```
2 print(prediction)
3 print(scores.shape)
4 #0.576449
5 #(22, 48)
```

B.1.3 Visualize feature importances

```
1 predictor.visualize_patient_scores(patient_tensor=patient
    ↪ , importance_scores=scores)
2 #see figure B.2
3 predictor.visualize_evidence(importance_scores=scores)
4 #see figure B.3
```

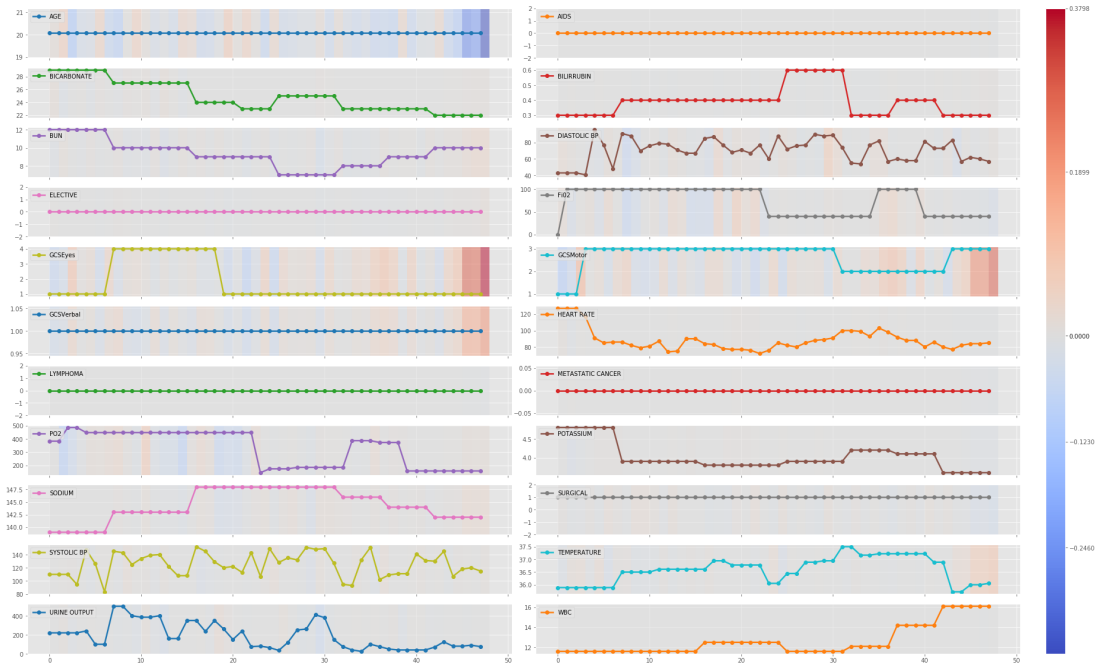


Figure B.2: Visualizing predictor evidence by hour with ISeeU.

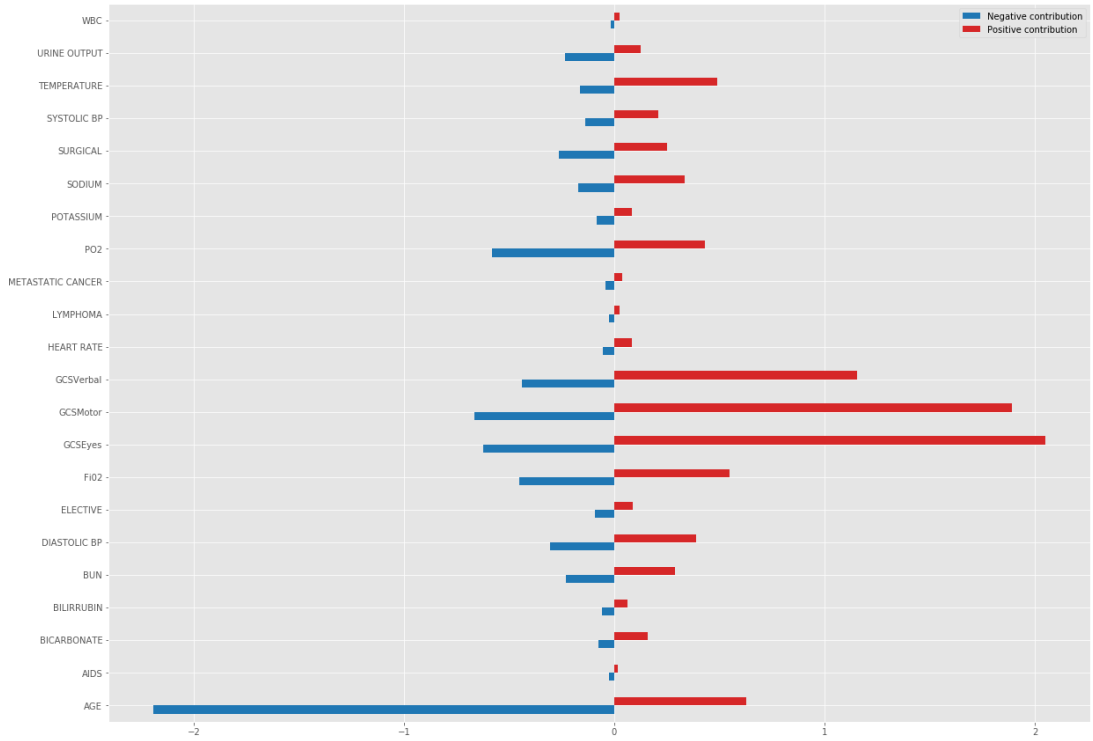


Figure B.3: Visualizing aggregated evidence with ISeeU.

B.2 ISeeU2

B.2.1 Predict patient mortality

```
1 from iseeu2 import ISeeU2
2 patient_sequences = predictor.preprocess_notes(
    ↪ patient_notes)
3 mortality, scores = predictor.predict(patient_sequences)
4 print(prediction)
5 print(scores[0].shape)
6 #0.09
7 #(1, 500)
```

B.2.2 Visualize feature importances

```
1 predictor.get_word_clouds(patient_sequences[0],
    ↪ shapley_values[0][0])
2 #see figure B.4
3 html_convolved = predictor.get_note_heatmap(
    ↪ patient_sequences[0], shapley_values[0][0], convolve
    ↪ =True)
4 from IPython.core.display import HTML
5 HTML(html)
6 #see figure B.5
```

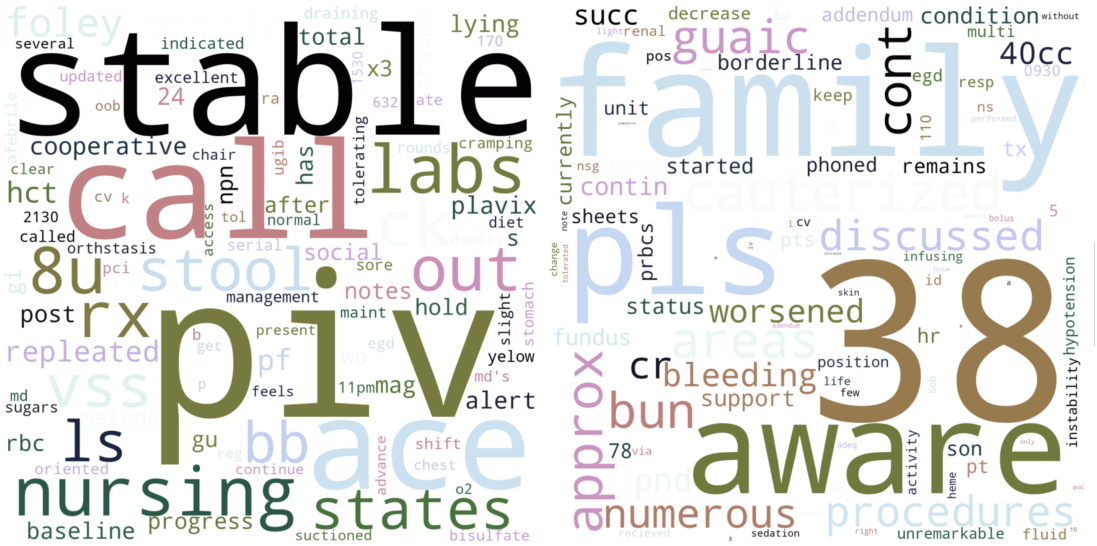


Figure B.4: Word clouds generated by ISeeU2.

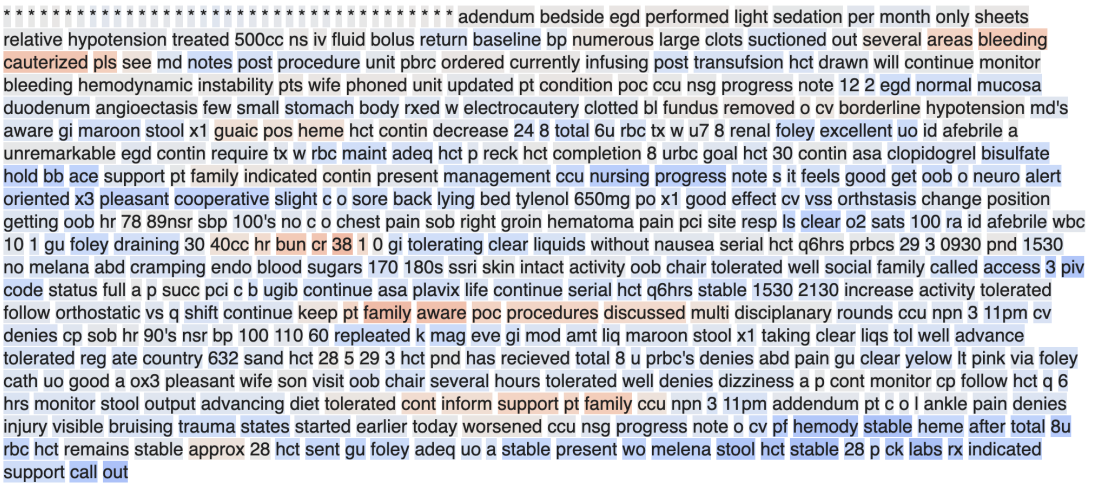


Figure B.5: Text heatmap generated by ISeeU2.