Machine Learning for the Improvement of Patient Flow

Rasheed El-Bouri

Worcester College

Supervised by
Professor David A. Clifton
Dr. Tingting Zhu

Trinity Term 2021

This thesis is submitted to the Department of Engineering Science, University of Oxford, in partial fulfilment of the requirements for the degree of Doctor of Philosophy

Declaration

I declare that the work contained in this thesis is entirely my own except where explicitly noted.

Rasheed El-Bouri
31st March 2021

Acknowledgements

Completing this thesis would not have been possible without the help and supervision of Professor David Clifton and Dr. Tingting Zhu. Their guidance and our discussions over exotic teas (and countless coffees) proved invaluable, and for that I am sincerely grateful. I'd like to thank Corinne for proofreading everything I wrote and greatly improving the quality of my work. I'd also like to thank all of my labmates; Tom, Chenyang, Heloise, Farah, Dani, Mohammed and all of the postdocs for the great discussions and always agreeing to take a coffee break with me.

I'd like to express my gratitude to the Worcester College crew; Adriana, Shannon, Inès, Nesrine, Johann, Matteo, Giulia, Lea, Ed and Julien for the dinners, the debates and the great memories. I'd also like to thank Doaa, Saleem, Hachem and Samah for the excellent company and helping me forget about work every once in a while.

I am especially grateful to Gogs, Theo and Sasha. Gogs for being one of the most humble and genuine friends, Theo for always getting in touch and asking how I'm doing (although I'm still working on actually answering my phone) and Sasha for always having more faith in me than myself. Thanks guys, I really appreciate it.

Last but certainly not least I'd like to thank my family. Thank you to my mother, Halima, and my father, Khalid, for their endless patience and support throughout the duration of my research. Thank you to my sister, Dana, for keeping me laughing when things got stressful. And thank you to my brother, Wahbi, who I now admit always gives great advice, even if I always choose to do the opposite. I'm indebted to you all.

Relevant Publications

- **El-Bouri, R.**, Zhu, T., Clifton, D. A. (2018). Curriculum Learning for Training Neural Networks on Medical Data. Machine Learning 4 Health Workshop, NeurIPS 2018.

- **El-Bouri, R.**, Eyre, D. W., Watkinson, P., Zhu, T., Clifton, D. A. (2020). Hospital admission location prediction via deep interpretable networks for the year-round improvement of emergency patient care. IEEE Journal of Biomedical and Health Informatics, 25(1), 289-300.

- **El-Bouri, R**., Eyre, D., Watkinson, P., Zhu, T.*, Clifton, D.* (2020, November). Student-teacher curriculum learning via reinforcement learning: predicting hospital inpatient admission location. In International Conference on Machine Learning (pp. 2848-2857). PMLR.

- Alsharid, M.[1], **El-Bouri, R.**[1], Sharma, H., Drukker, L., Papageorghiou, A. T., Noble, J. A. (2020). A Curriculum Learning Based Approach to Captioning Ultrasound Images. In Medical Ultrasound, and Preterm, Perinatal and Paediatric Image Analysis (pp. 75-84). Springer, Cham.

- **El-Bouri, R.**, Taylor, T., Youssef, A., Zhu, T., Clifton, D. A. (2021). Machine learning in patient flow: a review. Progress in Biomedical Engineering.

- **El-Bouri, R**., Eyre, D., Watkinson, P., Zhu, T.*, Clifton, D.* (2021). Towards Scheduling Federated Deep Learning using Meta-Gradients for Inter-Hospital Learning. arXiv preprint.

- Alsharid, M.[1], **El-Bouri, R.**[1], Sharma, H., Drukker, L., Papageorghiou, A. T., Noble, J. A. (2021). A Course-Focused Dual Curriculum for Image Captioning. IEEE International Symposium on Biomedical Imaging (ISBI) 2021.

- Youssef, A., Kouchaki, S., Shamout, F., Armstrong, J., **El-Bouri, R.**, Taylor, T., ... Eyre, D. W. (2020). Development and Validation of Early Warning Score Systems for COVID-19 Patients. medRxiv.

- Patent - Javed, H., **El-Bouri, R.**, Bishop, J., Zhu, T.T., Taylor, T., Watkinson, P.J., and Clifton, D.A.: Method and Apparatus for Monitoring a Patient GB 1817708.9, 2018

Contents

List of Figures

List of Tables

Chapter 1

Introduction

When a country's population and average age increase every year, it is inevitable that a strain is placed upon its healthcare system. This is due to the clinical attention that is generally required by older people and the increasing size of the ageing population. This is the current situation faced by the UK, as well as many Western European countries [6], [25], [39]. National media outlets can be particularly vocal about the performance of healthcare systems which makes the desire for a solution to poor efficiency in healthcare systems not only technically and economically desirable, but also politically important. The ability to cope with the demand for efficient healthcare has recently further been compromised due to austerity measures implemented by some EU governments as a reaction to the financial crisis of 2008 [72], [80]. Reductions in funding for healthcare have led to closures of many Emergency Departments (EDs) throughout the UK [121]. This in turn has concentrated the flow of ED patients to the remaining ED and has therefore increased the demand placed upon those hospitals. There is also the added complication that there is a governmental target for the National Health Service (NHS) that states that 95% of patients in the ED must be admitted, transferred or discharged within 4 hours. If this target is not met (termed "breaching"), a fine of £120 is levied upon the hospital for every patient waiting longer than four hours (the fine increasing drastically if the patient is not admitted, transferred or discharged within 12 hours)[195]. In 2016 this missed target

was one of the reasons the government withheld over £600 million from the NHS [103]. The government has since relaxed the fines administered upon NHS Trusts, however this has still not helped improve the efficiency or quality of service provided. Despite the focus of this thesis on hospitals within the UK, this is not a uniquely British problem and is faced by hospitals across the world [24]. The thesis has been framed in as general a way as possible, so that any hospital with electronic health record (EHR) systems implemented may also benefit from the findings.

1.1 What is Patient Flow?

Patient Flow is a term used within the NHS and other healthcare services to refer to the way in which patients are moved through a healthcare facility. It involves the medical care, resources, and internal systems needed to get patients from admission to discharge while maintaining a standard of quality of care and satisfaction for the patient [75]. In this thesis, I aim to show that patient flow is predictable using machine learning techniques. I aim to demonstrate that given data collected from a patient at point of entry to the ED, their immediate destination within the hospital can be predicted well ahead of their need to be transferred, and, as a result, clinical resources and bed space can be planned ahead of time. This in turn will improve the flow of patients out of the ED and into the hospital, which will reduce waiting times for patients as well as avoiding fines imposed due to breaching.

1.1.1 The Structure of the Hospital

The hospital being analysed in this work is the John Radcliffe Hospital, one of the five hospitals in the Oxford University Hospitals NHS Foundation Trust, Oxfordshire. This hospital consists of the following types of units:

- The emergency department (ED): This is where patients who need emergency medical attention or surgical intervention present to. There are also intensive

care beds for severe emergency presentations. Patients arrive either through their own means or by calling out an ambulance to take them to the hospital. The ED is generally seen as a separate entity to the rest of the hospital.

- The emergency assessment unit (EAU): This unit is located immediately next to the ED. This is classified as a ward where patients requiring emergency medical attention are admitted to after being processed through the ED. However due to the close proximity of the ED, this ward can be used as a buffer zone during busy periods to overcome the aforementioned four hour target. This renders the EAU a complex ward to classify due to its dynamic purpose.

- The surgical emergency unit (SEU): This ward is the equivalent of the EAU but for when patients require surgical intervention. Due to the SEU being further away from the ED physically, there are fewer patients admitted for convenience than is the case for the EAU.

- Wards: The wards are rooms which patients are allocated to typically based on the specialty of the ward. Each ward contains a set number of beds which defines the patient-treating capacity of the ward. The number of beds is not only decided by space but also by the manpower and equipment available to provide care for the patients. Wards can be stayed in overnight, but can only be discharged from between 9 a.m. and 8 p.m.

- Day Case Unit: The daycase units are ones where scheduled appointments are due to take place. These include appointments with medical specialists in clinic, or routine surgeries that allow the patient to be discharged the same day. Patients cannot stay overnight in daycase units.

1.1.2 The Admission Process

Patients can be admitted to hospital either through an emergency admission or as an elective admission. Elective admissions are those which are pre-planned and therefore are accounted for by the hospital staff. This can be for surgery or other routine treatment and may involve the patient staying in the hospital overnight or longer. However, as the procedure that the patient will undergo is already known, good estimates of the patient's length of stay are possible and this can be incorporated into the hospital plan.

Emergency admissions can either be delivered to the ED through their own means or by an ambulance, or straight to the relevant ward for their emergency. One example is patients who are clearly suffering from a heart-related problem being delivered straight to the cardiothoracic ward. Despite not being processed through the ED this will still be counted as an emergency admission. The process of emergency admissions through the ED is outlined in Figure 1.1.

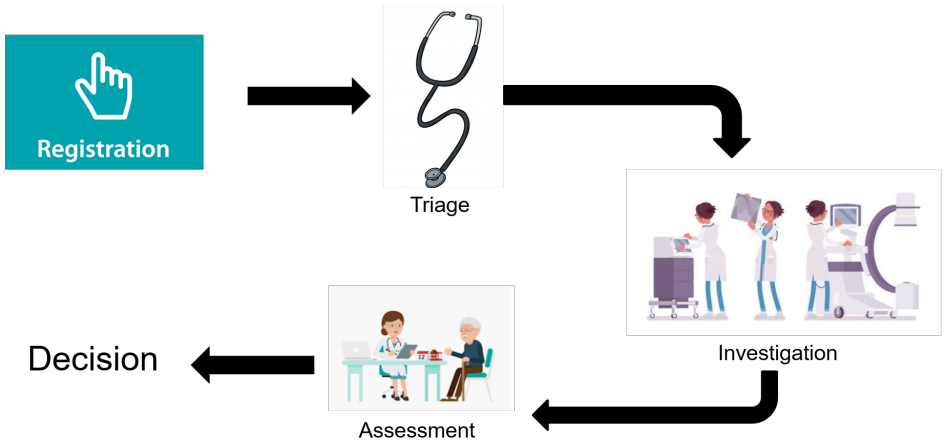


Figure 1.1: Process of admission to hospital through the ED as is described below.

For patients who are admitted to the ED the following process is followed:

1. Patient registers at the registration desk provided they are in a state to do so. Patients not in a state to do so are immediately taken in for care.

2. Upon registering their attendance a wrist band is printed creating their new entry

in the electronic health record (EHR) system. Any previous visits (admitted through the ED or otherwise) or diagnoses of this patient are immediately available upon the printing of this wrist band.

3. The patient is then seen by a triage nurse. The triage nurse will take vital sign measurements of the patient, understand their complaint, request any tests they believe might be needed and log a score of the severity of the patient.

4. The severity score is used as a prioritisation queuing system. Patients must wait in the waiting room to be seen by a clinician in the meantime.

5. The clinician will then see the patient and the information given by the triage nurse. After the consultation, the clinician will make a decision to either admit the patient to hospital, schedule a daycase appointment and discharge the patient or discharge the patient from the ED without an appointment. Often, a clinician will seek further advice from a specialist. Should a specialist not be available at the time of being seen in the ED, the patient may be admitted to the EAU until a specialist becomes available.

6. If the clinician decides to admit the patient, a handover of care needs to occur. This includes the communication of the case to the relevant team and a request for space within one of their wards.

The waiting time experienced after step 3 and before step 5 is usually the longest step in the process. The NHS target of being seen within four hours involves reaching step 5 within four hours with the timer starting from when the wristband is printed in step 2.

1.1.3 Key Decision Areas for Patient Flow

There are four predictions which can have a great impact on the flow of patients through the hospital. They are: the prediction of patient admissions to the ED, the

prediction of the movement of patients from the ED into the hospital as an inpatient, the prediction of movement within the hospital and the prediction of discharge from hospital. This thesis will primarily focus on the prediction of movement into hospital from the ED, however the other predictions will also be discussed in the wider context of patient flow. The first prediction - of patient admissions to the emergency department - provides us with an idea of how many patients will present to the ED on a given day which in turn guides the staffing levels of the ED. A model that performs well at this task will be useful in preventing a backlog of patients needing to be seen in the ED at it's busiest time.

Another prediction with potential for impact on the patient flow process is predicting where patients from the ED will be admitted to within the hospital. This boundary between the ED and the main hospital is called the ED-inpatient interface (EDii). This prediction also has the potential for reducing overcrowding of the ED. Current practice is for the patient to wait until they are seen, and if the patient requires hospitalisation the clinician then requests bed space. The bed space however can take hours to prepare and the patient will remain in the ED while it is prepared for them. Predicting where the patient will be admitted before the bed request is made will therefore provide a longer amount of time to prepare a bed prior to the request, and will help to move patients out of the ED.

The third prediction that can impact on patient flow outcomes is predicting how patients and resource will move within the hospital for admitted patients. This would provide some understanding of the staff and spaces required in order to treat the inpatient.

The final prediction which can have great impact on patient flow is the prediction of when a patient is ready to be discharged from the hospital. Currently there are many beds which are occupied by patients who are medically fit for discharge. A well-trained prediction model would allow clinicians to identify patients ready for discharge to empty beds in the wards and improve the flow from the ED and into the

hospital.

1.1.4 Why Patient Flow from the ED is Important

Patient flow is of great importance for patient safety and for pressures on clinical staff [219]. Predicting the outcome of the EDii is a known complex problem;- Staib et al. [135] emphasise the importance of this interface by discussing how there is significant evidence to show that the delayed transfer of emergency patients to wards is associated with a 20-30% relative increase in inpatient mortality [135]. They also describe why this problem is difficult to predict. This is primarily due to the EDii being poorly defined in terms of clinical ownership, as well as the fact that the unscheduled nature of emergency admissions disrupts scheduled activity within the hospital, thereby slowing the movement of patients out of the ED. This can lead to patients being admitted to wards that are not ideal for their treatment in order to release bed space in the ED, which can be hazardous [30]. By providing a prediction of the likely inpatient admission location, we seek to begin bridging the gap in patient flow between the ED and the inpatient wards.

In addition, a study by Guttman et al. [55] showed that for patients whose transfer out of the ED is delayed, the risk of an adverse event occurring increased with the mean length of stay in the ED. It has also been shown by Karakusevic et al. [108] that waiting times in the ED can increase due to a lack of available beds in the main hospital when there is a need to admit patients. As mentioned before, the ED is conventionally referred to as being distinct from"the hospital"; patients are thus "discharged from the ED" and "admitted to the hospital" in clinical parlance. A lack of bed space within a hospital could therefore be causing greater waiting times in ED, and in turn increase risk to patients who are waiting. There are also financial incentives for providing as efficient a service as possible [33].

1.1.5 Current Management of Patient Flow

An important briefing published by the Nuffield Trust in 2016 gave a detailed analysis of the current state of patient flow through UK hospitals [108]. The briefing aimed to explain why, given that there have been many reorganisational and structural changes to the function of the EDs over the years, the flow of patients through the ED has not yet improved. The briefing concludes that the current methods of measuring occupancy do not recognise the need for resources to move patients in and out of beds, which peak at different times to actual occupancy. There is a process that needs to be followed before admitting a patient to a new bed in a new ward: the new bed area needs to be cleaned, the patient must be transferred, there must be an official handover of the patient to the new ward, and this must be officially communicated between the hospital staff [75]. All of these steps require hospital staff time and resource. The study also recognises that small changes in the length of stay (LOS) of a patient within a ward can cause significant disruption to the flow of patients through a hospital. Karakusevic also recognises that a minority of long-staying patients ("bed-blockers") occupy the majority of bed use. The author concludes that there is therefore a need for new computerised management tools that can support clinicians in decision making. From this we see that if a clinician in the ED is able to make a prediction of where a patient will be admitted in the hospital after leaving the ED, a line of communication can be created (potentially hours in advance of actual admission) signalling the need for bed space. This would not only allow the resource to be planned and made available in advance, but would also prevent any increased LOS of a patient in the ED.

This briefing inspired many researchers to use modern techniques developed in machine learning and mathematical modelling to attempt to alleviate the patient flow problem. Due to the nature of these methods this led to the sub-categorisation of patient flow into simpler flow sub-problems to solve as opposed to creating a management process to jointly optimise the entire process.

Some argue that patient flow is simply a management problem and should be

resolved by restructuring or reorganisation of the entry point of a hospital. Miro et al. [24] measured the performance of a Spanish hospital before and after a structural reorganisation. They found that the ED performed significantly better at seeing patients after restructuring and that the average waiting time was reduced from 87 minutes to 24 minutes. While this is an impressive improvement, one of the strategies used in the paper was to add capacity to the ED. While this may be possible in smaller hospitals, large hospitals with high volumes of patient throughput such as the John Radcliffe Hospital in Oxford would struggle to implement this due to the limited space available, as well as the time and cost associated with such a change. There is also the concern that management techniques developed within one hospital might not be generalisable to others. Therefore, while the reorganisation of the ED worked for the hospital in this study, we have no guarantee of the restructuring being applicable or appropriate for any other hospital. The authors argue that a data-driven model is thus needed that can be generalised to all hospitals in a robust manner. [36] are also critical of current management approaches to managing patient flow in hospitals. They state that current management methods do not consider the entire patient journey from admission to discharge. Each ward unit is considered independently and their performances are measured individually. As a result, optimising the performance of an individual unit will not necessarily optimise the flow of patients throughout the hospital. [5] are also critical where they state that patient flow management decisions are made without the use of quantitative model-based analyses. Given the need for prediction and the difficulty of increasing capacity in a hospital due to capital costs, this thesis advocates for the development of a ward admission prediction tool. The aim of this tool is to aid ED clinicians in predicting where a patient will be admitted within the main hospital after discharge from the ED, allowing preparation of a bed well ahead of time. This should use information about a patient as soon as they are admitted to the ED to provide as much time as possible for bed preparation, and thus prevent the effects of flow disruption due to increased LOS of patients in the ED.

The tool should consider all patients, and not just patients with a specific condition: patients with specific conditions are more likely to use only specific wards within a hospital and so any optimisation of flow would only optimise over those wards and not the system as a whole.

1.2 Summary

This thesis aims to achieve three objectives. The first is to provide an understanding of what work has been published in machine learning for patient flow. Due to the private nature of healthcare data, most work published in this field is highly disjointed. As a result, the first objective of this thesis is to provide a comprehensive overview of existing research in this field and highlight the gaps. This is carried out in the literature review in Chapter 2.

The second objective was to develop machine learning methods to aid in improving the current state of patient flow in hospitals. Several methods have been developed to exploit a dataset extracted from hospital EHRs and maximise the efficacy of predictive patient flow models as can be seen in Chapters 4, 5 and 6.

The final objective is to provide insights to clinicians using these machine learning algorithms. These should guide clinicians in feature selection and prioritisation, define the data that they should sample from in their datasets and what datasets to use for training. These are also discussed in chapters 4,5 and 6.

A summary of the structure of this thesis is as follows: Chapter 2 contains a review of the work conducted in the field of machine learning for patient flow. This serves to provide a landscape of the work conducted thus far. From this it is concluded that a prediction of where in the hospital an emergency patient will be admitted is the most clinically useful and actionable. In Chapter 3, the dataset to be used for this problem is then analysed. It is then shown how a feature set for this problem can be created from the EHR. In Chapter 4, it is shown how curriculum learning can benefit the

training of neural networks for classification and a curriculum is designed specifically for EHR data. In Chapter 5, in order to maximise the performance of the classifier as early as possible in the ED admission, training the neural network is turned into a reinforcement learning problem. It is shown that a machine can develop teaching strategies using the curriculum. In Chapter 6, a method for learning from multiple data centres using meta-gradients is presented. It is also shown how this method can overcome learning from data sources with systematically recorded errors as can be the case in EHR systems. In Chapter 7, the thesis is concluded and summarised. It is hoped that the methods and insights that are presented in this thesis will be of use to the machine learning and medical communities.

Chapter 2

Literature Review

This chapter is a review of the ways in which machine learning has been used in order to plan, improve or aid the problem of moving patients through healthcare services. We decompose the patient flow problem into four subcategories: prediction of demand on a healthcare institution, prediction of the demand and resource required to transfer patients from the ED to the hospital, prediction of potential resource required for the treatment and movement of inpatients, and prediction of length of stay and discharge timing. The following sections investigate the machine learning studies and models that have been used to tackle these problems.

2.1 Motivation

Patient flow is a topic that has been studied extensively by various researchers of differing backgrounds. As a result, the literature associated with the improvement of patient flow is vast and a diverse range of techniques from different disciplines are employed in an attempt to tackle the problem. In this review, we will primarily focus on the history of how patient flow has been handled, as well as techniques that involve the use of machine learning methods. This is, however, by no means an exhaustive review of all methods used for the improvement of patient flow. It should also be noted that this review is not intended to summarise the machine learning methods

that have been applied to patient flow or the best performing models for each task (as seen in [198]) and so the performances of the models will not be included. Rather, it is to provide some structure to the field of machine learning applied to patient flow, to allow researchers to see how machine learning has already been applied to the patient flow problem and to identify where there are (to the best of our knowledge) gaps in the literature.

While some authors have attempted to tackle patient flow as a single system through a hospital, most researchers break the problem into smaller constituent problems to tackle. These constituent parts are usually associated with the key flow bottlenecks in hospitals and these are: (i) prediction of patient admissions and demand on EDs, (ii) prediction of flow through the EDii (i.e., handover from ED to the hospital), (iii) prediction of movement of patients (and associated resource) within the hospital and (iv) prediction of length of stay. In this review we will discuss the work published in all of these topics and how they have been used to improve patient flow through hospitals.

2.2 Outline

Figure 2.1 shows the process of hospitalisation for many hospitals with an ED. Hospital visits can be decomposed into two overarching types of admission: elective (planned) and emergency (unplanned). It is generally the unplanned emergency admissions which cause the greatest disruptions to patient flow through hospitals [46].

Elective admissions are planned prior to their admission. As a result, the resource for these patients has been planned and there is bed space should it be needed. Elective patients have also been shown to have consistent lengths of stay in hospitals, meaning that they cause minimal disruption to the flow of the hospital [64].

Due to each emergency case being different, there can be no estimate of the resource required or how long each patient will stay in hospital prior to their arrival. These

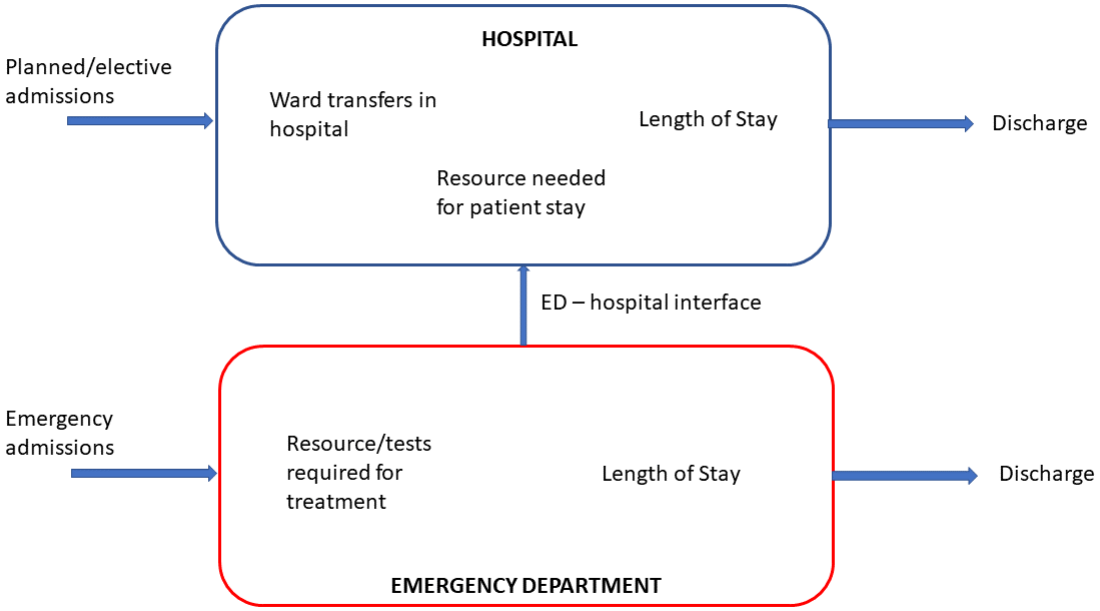


Figure 2.1: A visualisation of the process of hospitalisation and the main considerations at each stage from a patient flow perspective.

therefore have become popular topics for the use of machine learning for prediction. Should these patients need hospitalisation, there is again little warning and so adapting the planning of the hospital becomes difficult.

In the following sections we will look at the work that has been carried out in applying machine learning to all of these sections of the hospitalisation process, the techniques that have been used and where we believe researchers should focus their attention in the future to further improve patient flow.

2.3 Management of Patient Flow in Current Clinical Settings

The effect of poor resource management on patient flow within the hospital is well known. Conceptually, high patient flow can be achieved by the effective balance of supply and demand within the system. If the supply of beds, staff and equipment is

readily available to meet the needs of patients arriving at the door, then few perceivable barriers exist to prevent their immediate usage. However, studies of waiting lists have long shown that increasing supply in fact leads to a proportional stimulation of demand, highlighting the inadequacies of using relative need for services as the sole basis for resource provision [2]. If increasing supply cannot satiate demand, the optimisation of existing resources is an obvious and necessary strategy. Oredsson, Jonsson, Rognes, *et al.* [57] reviewed modern triage-based interventions designed to improve patient flow in emergency departments, demonstrating that the most significant improvements are observed through the use of fast-track and team triage approaches.

Current approaches to the management of patient flow in hospitals are typically driven by the need to report and improve upon key performance objectives. Within the NHS in the United Kingdom, the introduction of the Patient's Charter (1991) allowed providers greater flexibility to curate local operational policies, whilst imposing stricter performance and reporting structures across the system [90]. By specifying the metrics required to deliver an adequate level of care, the identification and treatment of bottlenecks in the system naturally become a focus of attention. Such metrics are often objective and time-based, such as the time taken for acute arrivals to be admitted or discharged. Perhaps the most significant of these targets introduced within the NHS was that of the 4-hour waiting limit for ED arrivals, stipulating the need to admit, transfer or discharge a patient within this timeframe [28]. The most widely used approach to fulfil this target in the UK is the use of the "See and Treat" framework, which encourages rapid, on-arrival assessment of the patient's needs by an individual clinician, and allows full autonomy to that clinician to decide the treatments, referrals and investigations necessary to facilitate their care, or be discharged as appropriate. Saint Lamont [32] discussed the benefits and limitations of this approach, including the barriers to adoption observed when additional resources or suitably trained staff are unavailable.

A lack of efficiency and poor patient flow is typically perceived to correlate with

a reduction in staff availability. This observation is particularly valid where patient satisfaction is concerned. A study by [12] showed that positive overall satisfaction was associated with the perception of short waiting times and accurate information delivery, rather than actual waiting times. Whilst increasing staff within the emergency department may improve turnaround times for rapid triage and discharge of non-urgent cases, it is less likely to result in an improvement for patients requiring admission, as shown by [27], indicating that the true bottlenecks exist further in the pathway beyond the emergency department. This fact has been clearly recognised in recent guidance, where the focus on enabling patient flow has shifted away from the performance of the ED and towards acute networks and support services [120]. At one end, Clinical Streaming has been introduced as the process by which patients are assigned to one of several parallel pathways, according to their care requirements, allowing for more structured and reliable coordination of support services within the hospital [199]. At the other, Discharge to Assess (D2A) models emphasise the need to address unnecessary delays in discharging clinically optimised patients from hospital, due to a lack of funding or support within the community [132], [203].

2.4 Machine Learning for Patient Attendances

2.4.1 Prediction of Emergency Attendances

The number of patient attendances to the hospital within a specified timeframe is arguably one of the most important aspects of patient flow. It determines the demand that is placed upon the hospital and therefore affects how many patients can be treated to a high standard of quality and care. The importance of predicting patient admissions is reflected by the number of publications in this area. However, with little information on patients prior to their arrival, it is also one of the most difficult areas of patient flow in which to create accurate predictions.

Some works predict the number of emergency attendances using multiple regression

[21], [40], [62]. They frame the problem such that they forecast for daily attendances as well as weekly and monthly attendances. The use of regression is for interpretability of the predictions as well as the development of a simple model to improve the chance of being able to generalise to other hospitals. As mentioned previously, due to limited information on these patients prior to arrival, the authors use the days of the week and national holidays as features.

Whereas the aforementioned studies approach the problem as a static prediction (i.e. using information from a snapshot in time to make predictions), [11], [47] and [45] treat the problem as a time-series. They use autoregressive models to account for the trajectory of the numbers of patients. This approach is more likely to be successful than a static approach due to the incorporation of data close to the event of interest. However, the benefit of a static approach (if the model is accurate) is that a prediction can be made at an early stage and action can be taken based on that prediction without needing to wait for the time-series to unfold. These time-series approaches also perform regressions to predict patient volumes in the coming days, weeks and months.

While the seasonal features such as weather and time of the year have been shown to be helpful with predicting patient numbers, they are not patient-specific and therefore are limited in their use for predicting when a patient will be admitted to hospital. As a result, [50] and [192] consider predicting patient readmissions or reattendances to the ED instead of predicting any given attendance. In doing so, they are able to utilise the wealth of data already recorded by the hospitals on individual patients and identify markers that indicate high risk of readmission in an emergency. [76] use Naive Bayes and a decision tree in order to classify patients who will be readmitted to hospital using their health records as features. These works generally pose the readmission problem as predicting readmission within the next 30 days as this has the most impact on the health and welfare of the patient, as well as the scheduling of the hospital [81].

A problem that can arise due to these readmission predictions is that patients can be readmitted for various issues (for example a patient who was hospitalised for cardiac issues might need rehospitalisation for breaking their leg). Considering this type of readmission does not generate useful information on the underlying clinical issue, as it is not indicative of the underlying condition and so the health records of the patient will not be useful for this prediction. To get around this issue, many authors have conditioned their prediction of attendance on subsets of patients with certain underlying conditions. A study by [134] uses a naive Bayes classifier to predict readmission and only considering a subset of patients with heart failure. They only consider a readmission to be valid if the patients are readmitted with heart failure within 30 days. Conditioning on a subset of patients is also seen in [151] where they use patients who have had a neurosurgical procedure carried out and compare the performance of their model (trained used gradient boosted trees) using features available during the patients stay versus features that were obtained after the patients discharge. Naturally the model with access to features after the patient discharge performed better, however it is very difficult in most situations to obtain features post-discharge. A similar study is carried out by [176] but they consider patients suffering from COPD. They investigate various machine learning methods and find that gradient boosted trees offer the best prediction of readmission accuracy for their dataset. They also utilise recurrent neural networks in order to treat the problem as a time-series problem but the performance is significantly worse. In fact there are very few works that treat the prediction of readmission as a time-series due to the difficulty of obtaining data on patients post-discharge [48]. All of the autoregressive models cited are autoregressive moving average models and so fit a linear model to differences in their time-series using ordinary least squares.

2.4.2 Scheduling Instead of Admissions

The ultimate aim of all of the works mentioned in Section 2.4.1 is to provide the hospital with an understanding of the volumes of patients that may be attending the ED. By forecasting this (and providing that the model is accurate) the hospitals may then plan the appropriate resource (including staff, tests and making equipment available) in order to be able to cope with the demand placed on them. For low numbers forecast, hospitals may also then reduce the required resource that is on standby which can lead to cost savings [37].

Some authors however approach the problem from the scheduling perspective. This is different in that whereas predicting admissions makes the assumption that resource can be altered to meet demand, the scheduling approach does not. With this approach authors assume that there is fixed resource and how it is used can be optimised with varying patient numbers.

In [182], the ED scheduling problem is defined as needing to satsify the following constraints: the schedule must minimise the risk of adverse consequences, minimise patient waiting time, minimise patient length of stay, minimise ED crowding and minimise interruption to caregivers. They use a mixture of health record data of the patients and data on the status of the ED to reconstruct the state of the ED when the patients were there. They next use a mixed integer linear program to optimise these scenarios, maximising throughput while being constrained by the aforementioned constraints. They then train a deep learning architecture on this optimised data and use it as a ranking system to predict the optimal patient-caregiver pair in the ED.

Some authors prefer to allow the machine learning algorithms to discover the optimal policies instead of optimising the problem themselves to learn from. This is seen in [207] where a deep Q network (a reinforcement learning algorithm) is used to learn the optimal policy of treating patients in the ED. In order to do this a simulation is made of the ED which will allow the agent to take exploratory moves essential for reinforcement learning. The state of the model is defined as the distribution of acuity

(sum of patients at each acuity level) within the ED as well as the distribution of needed treatment type. The action of the agent is to rank the next patient who needs to be seen, meaning it is also a patient priority-ranking system. A priority ranking system is also presented in [172] based on severity prediction, but the authors go as far as predicting whether patient presentations to the ED should be treated as elective visits given their low severity. They do this using the primary diagnosis code of the patient, however there may be difficulties in expanding this tool to other hospitals given that many hospitals assign diagnosis codes after the patient is discharged from hospital and not at admission.

The scheduling problem is approached slightly differently in [38] and [70] in that instead of ranking the priority patients in the ED, they aim to design the staffing schedules to optimise patient flow. They do this using genetic algorithms and allowing the staffing schedules to be updated and 'evolve' to a point where they are suitable for the demand placed on the ED. This approach makes the assumption that should the staffing level be predicted accurately, then there will be no need to prioritise patients in the ED as there will be enough staff (and resource) to process them.

Figure 2.2 shows the works that have been conducted so far on the prediction of admissions and scheduling in the ED. This is by no means an exhaustive summary but we aim to provide some structure to help other researchers understand what work has been conducted in the field of machine learning for patient flow through the ED. Table 2.1 further outlines the problems that have readily available datasets for prediction, and which models are popularly used to tackle the prediction problem in the literature. A lack of a readily available dataset for priority ranking is due to priority generally not being recorded in hospital EHRs. Readily available in this instance refers to existence in a typical hospital database and not that it is easily and openly accessible.

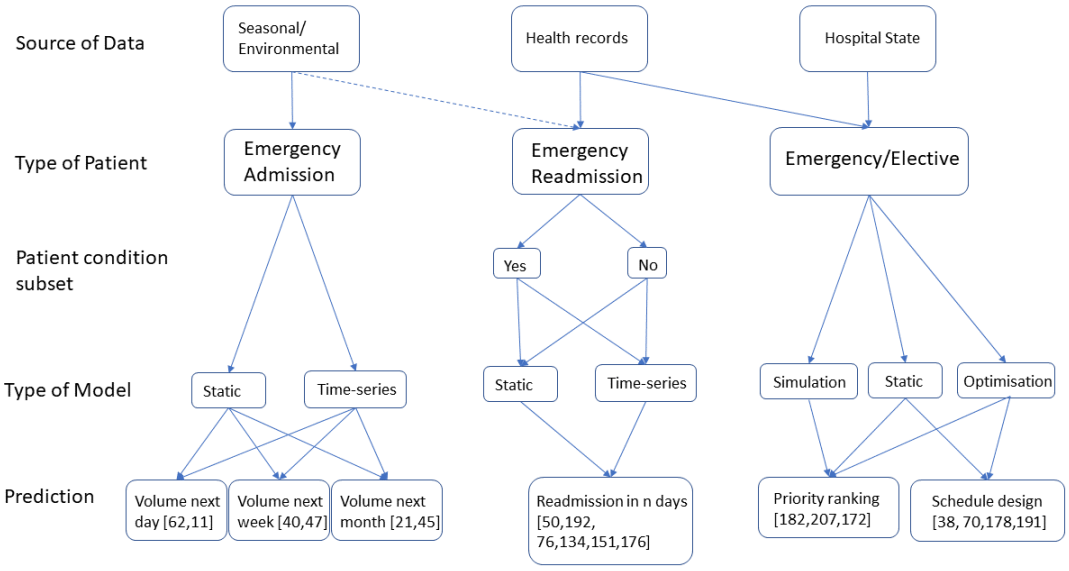


Figure 2.2: Visualisation of the studies that have been carried out regarding using machine learning to predict admissions and scheduling in the ED. Dashed lines indicate some studies opt to use these features.

2.4.3 Machine Learning in Elective Admissions

We have primarily focused on machine learning applied to emergency admissions as this is the larger body of research in the field. The stochastic nature of these admissions in terms of number and type of admission means that these are the most disruptive to patient flow in a hospital. Elective patients are generally planned for and so resource is available to treat them.

There are however studies that also apply machine learning to the admission of elective patients. In the study conducted by [178], the authors use machine learning to assess whether or not patients will actually attend their scheduled appointments in hospital. Despite the resource being prepared for these patients, a no-show will result in a waste of this resource and this work aims to provide a way to then re-direct that resource. The authors use information on the history of the patient with a gradient boosting machine to get a strong predictive accuracy. This same prediction of no-shows to elective appointments is carried out in [162], however they then leverage

Table 2.1: Table showing the popularity of different methods and data availability for each of the ED admission prediction problems. Data is considered readily available if most hospitals record the data required for the task.

	ED-admission problem			
	Vol. pred.	*Readmit Pred.*	*Priority*	*Schedule*
Labelled datatset readily available?	✓	✓	✗	✓
Regression methods popular?	✓	✓	✓	✗
Classification methods popular?	✓	✓	✓	✓
Genetic methods popular?	✗	✗	✗	✓

the risk of no-show in order to update the scheduling system of the hospital.

With many health systems providing long waiting times for appointments [73], [26], another important factor when it comes to elective patients is prioritising patients in the schedule. The approach in [191] is to first use a clustering algorithm to group patients into different priority categories. They then treat the schedule as a Markov decision process where waiting time for patients in the high priority clusters is to be minimised.

These approaches can be difficult to validate due to their direct impact on the scheduling of appointments. As a result, there is no chance to verify if the patients turn up or not once the schedule is changed. They also rely on historical behavioural data (such as how many times a patient has missed an appointment previously) which are not stationary distributions and therefore limit how successful supervised learning can be in this domain in the long term.

2.4.4 Interim Summary

Overall, the application of machine learning to predicting emergency patient admissions and scheduling is well-explored. Works are generally split between emergency and elective patients with further subdivisions according to the data used, the models used and what is being predicted (see Figure 2.2). Very few works validate their models in hospitals in real-time, most using a retrospective test-set to assess performance. Furthermore, some models are difficult to validate due to being designed to intervene

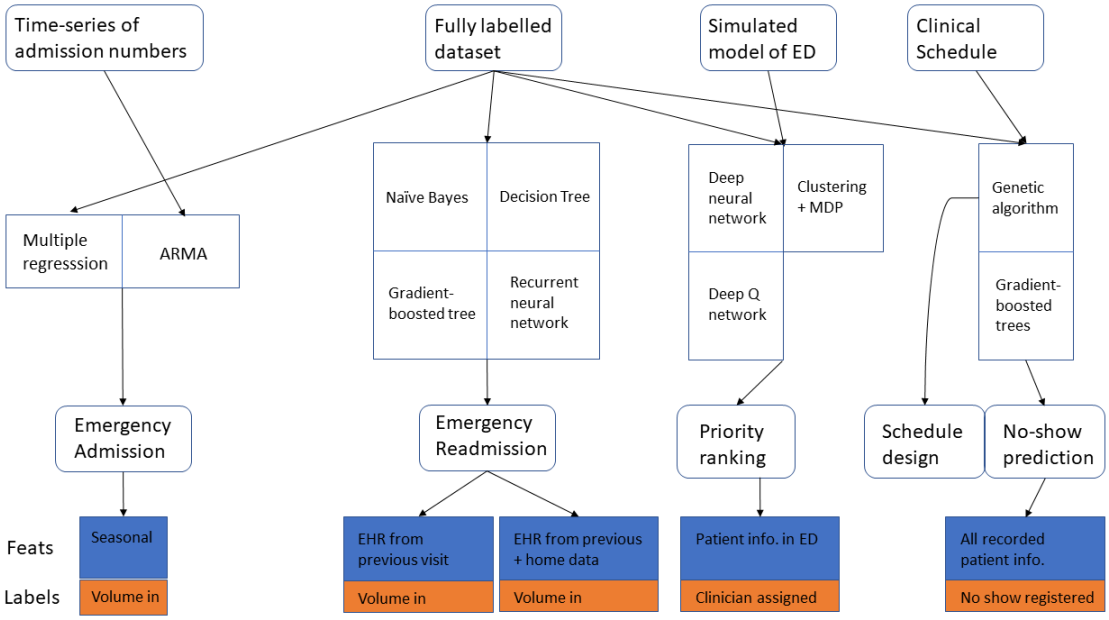


Figure 2.3: A lower-level visualisation of the studies conducted on ED admission. The top row indicates the pre-requisites for the model (what kind of data is needed). The row below shows the machine learning models used to tackle the prediction problems shown in the third row. The bottom two rows show the features and labels used for the predictions that require a fully-labelled dataset.

in the admission and scheduling process. A further difficulty is that very little data is known about patients when outside of the hospital unless they use wearable devices that can be tracked by the hospital. While this problem is difficult to predict, modelling of a poisson process representing stochastic attendances and integrating this into an overall model of patient flow through the hospital is likely to be beneficial for overall planning.

There is also very little connecting these studies. Most work is carried out with the data from the hospital that the authors are associated with and built around that. Due to hospitals being different from one another, that leaves little scope for building on previous work or developing models that can be used universally. A public dataset that could be held as the gold-standard for patient flow would aid in this significantly as a benchmark for experiments. In order to provide a more complete picture of the works that have been conducted in the space of machine learning in emergency admissions, Figure 2.3 provides a flow chart including the models and datasets that

have been used to make the predictions.

2.5 The Emergency-Inpatient Interface

The emergency-inpatient interface is an ill-defined area of many hospitals [135]. There is usually a lack of clarity on the ownership of this space of the hospital and who should manage the handover of patients from the ED to an inpatient setting. As a result of this lack of clarity, it should come as no surprise that there is much published on making predictions across this gap in the hospital. While it may seem like an obvious task to predict which patients need admission to hospital from the ED, it has been shown that this is not a trivial task [53]. Whereas the works discussed in Section 2.4 aim to provide predictions for planning (such as expected numbers or schedule planning), the predictions of the works found in this section are primarily designed for decision-support.

A fundamental question addressed in this section of the thesis is whether admission to the hospital from the ED can be predicted. [147] and [146] show this can be done using multiple machine learning models including a logistic regression, XGBoost and a deep fully-connected neural network. They show that this is possible using historic patient information as well as information from triage. This does however limit the potential use and application of the model to patients who already have electronic health records. This prediction is also carried out in [31] and [180] but using only a few variables that are measured early in the ED admission process and they show using a Bayesian network that admission to hospital can still be accurately predicted. This sentiment is echoed in [59], where they set up their classification such that the clinical staff may predict the risk of whether an inpatient bed is needed or not as soon as triage is complete in the ED. This prediction is then further augmented with the inclusion of the free-text written by the triage clinical staff as features to improve the performance of the model [143], [186].

As was the case for prediction of admissions in Section 2.4, many authors find it useful to consider certain demographics of patients. An example is in [154] where a multiple logistic regression is used to predict hospital admission from the ED for a cohort of patients over 70 years old and one of patients aged below 70 years. This is due to older patients generally being more at risk of admission and so by creating a model conditioned on age, they are able to better predict those most at risk of admission. In [210], elderly patients are considered to be those aged 75 and over, and they also show that accurate predictions of admission can be made for an elderly cohort of patients.

Another demographic that is often targeted for prediction is that of paediatric patients [29], [56]. In these studies, logistic regressions are used to predict whether a paediatric subset of patients will require admission to the hospital. Once again, by creating a separate cohort for these patients, they can make predictions comparing patients to other similar patients, rather than comparing with older patients who have different physiologies. This introduces a trade-off of improving model accuracy while reducing how generally the model can be applied.

Further subsets of paediatric patients have been made, for example, by considering those patients suffering from asthma exacerbation and predicting those most likely to be admitted to hospital for treatment [157].

To augment the performance of a model predicting paediatric admissions to hospital from the ED, the textual data recorded during triage can also be used as features [213]. Natural language processing techniques have been used in order to extract useful information which has been shown to improve predictability of admission.

2.5.1 Predicting Inpatient Resource Utilisation

Many of the studies that are created in predicting admission to hospital focus on subsets of patients with certain conditions. As these patients will require the same treatments and specialist staff to treat them, this can be seen as resource prediction

for patients being admitted to the hospital from the ED.

An example is in [67] where heart-rate variability in the ED is used alongside other demographic information on the patient as input features to a support vector machine. This is then used to create a score on the likeliness of cardiac arrest occurring in the next 72-hours. While this is not strictly framed for patient flow, this prediction allows clinicians to plan for resource in the cardiac department. Predicting whether or not a patient is septic is also important for patient flow in terms of resource planning. As a result, models predicting whether or not ED patients are suffering from sepsis have been developed [122], [167], [129]. The authors use a mixture of information available at triage, demographic information and free-text to make prediction of whether or not the patient is septic, which if accurate, could allow planning of their treatment before the patient becomes critically ill.

In fact, there have been many such studies predicting whether or not a patient is suffering with a certain condition in the ED which allows resource planning. These include predicting if a patient is suffering from acute kidney injury [209], requires intensive care [201], [202], is suffering from a urinary tract infection [163] or has bacterial infections [212]. In addition to these studies, [147] predict emergency hospitalisation of patients undergoing chemoradiation.

While these predictions are useful for planning patient flow, they are not explicit predictors of admission. A more explicit approach is seen in [175] where the classifier is trained to predict admission to hospital of patients suffering from bronchiolitis.

While predicting admission to hospital from the ED is useful, a greater level of granularity, such as which department in the hospital the patient will be admitted to, is more useful to clinical staff. An example is seen in [206] where rather than predicting admission, they predict the disposition of the admitted patient, choosing the destination unit from intensive care units, telemetry units, general practice units and observation units. As these 'ward types' tend to have separate resource, they are better able to adapt their resource according to the predictions made. In order

to make as general a model as possible, these studies of patient disposition do not consider subsets of patients but rather the entire population of the ED to replicate daily working conditions.

Figure 2.4 shows the general structure of works that have been conducted on predicting flow from the ED to hospital. It should be noted that as all of these works consider flow from the ED, all patients considered are emergency patients. Table 2.2 shows how readily available labelled datasets are for the EDii prediction problems and the popular approaches to tackling them. It should be noted that readily available here means data that would generally be saved on a hospital EHR and not data that would be easily accessible on a public dataset. In order to provide a more complete picture of the works that have been conducted in the space of machine learning in the EDii, Figure 2.5 provides a flow chart including the models and datasets that have been used to make the predictions.

2.5.2 Interim Summary

We have seen in this section that the emergency-inpatient interface in hospitals, while being ill-defined in practice, is well-researched using machine learning. Authors predict admission from the ED in order to provide information for clinical staff to prepare space should it be needed. To improve the performance of the classifiers, many authors condition their models on the demographics of the patients (e.g., elderly or young patients) or on the patient disease (patients suffering from the same ailment in the ED). In order to provide greater granularity on which resource will be used in the hospital, some authors also predict which 'ward type' will be used by the patient to be admitted to the hospital. Ideally, a model would integrate a stochastic model of admissions with a prediction of admissions across the EDii to make the operation of the ED less reactive.

However, once again there is little connecting these studies. None of the studies reviewed build from each other or use the same dataset for comparison. Furthermore,

the definitions used to categorise patients vary by paper. As was seen when categorising elderly patients some studies use an age threshold of 70 years and over and some use 75 years and over. Clearly it would be beneficial to have an agreed range to make models more comparable. This further emphasises the need for a shared, publicly available dataset for use when creating machine learning models for patient flow. All definitions of demographics should be included in the dataset so that researchers make valid comparisons to models. It will also be beneficial in allowing researchers to compare their methodologies and validate them on the same dataset as others as well as apply them to their own hospital's data. This will also make research more consistent, allowing researchers to build and improve upon each other's models instead of applying similar models to similar problems using different data.

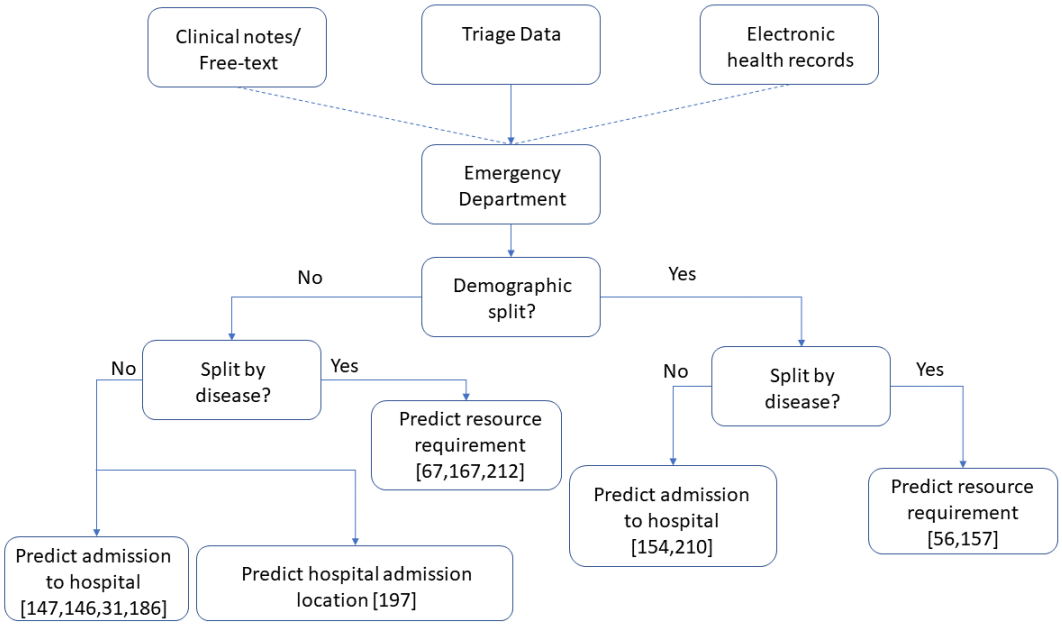


Figure 2.4: A decision tree showing how the studies that have been conducted on predicting movement from the ED to hospital are structured. Dashed lines indicate that these features are used in some works but not all.

Table 2.2: Table showing the popularity of different methods and data availability for each of the ED to inpatient interface prediction problems.

	EDii problem		
	Hosp. admit	*Hosp. admit loc.*	*Resource req'd*
Labelled dataset readily available?	✓	✓	✓
Regression methods popular?	✗	✗	✗
Classification methods popular?	✓	✓	✓
Bayesian methods popular?	✓	✗	✗

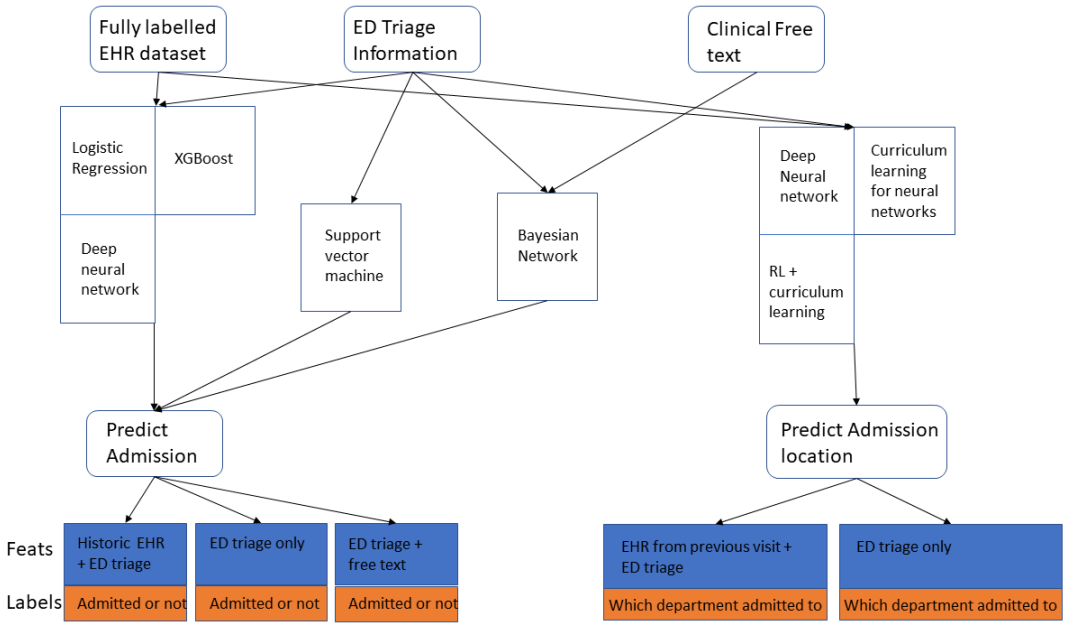


Figure 2.5: A flowchart showing the models that have been used for the separate prediction problems for predicting ED to inpatient admissions. The top row shows the sources of data used, the row below shows the models that have been used in various different works for the prediction problem, below this is the problem being tackled and below each of these problems is one of the datasets used in a study to train the model. RL is used as an abbreviation here for 'reinforcement learning'.

2.6 Intra-hospital Resource Management

Once patients have been admitted to hospital, there is yet another layer of resource flows that need to be considered. Patients can be transferred between wards, have tests carried out or be moved to diagnostic facilities in the hospital which house certain equipment such as MRI scanners. These all require staff to carry out the movements and therefore place a demand on the resource of the hospital. As this resource is part

of that needed to deliver the patients through hospital to discharge, it is relevant to patient flow.

2.6.1 Ward Transfer

The most common way in which machine learning is used to provide predictions for inpatient flow is through predicting if patients will be transferred to another ward. Note that while in Section 2.5.1 we considered studies which investigated patient degradation as a signal for resource preparation, we will not consider degradation for inpatients as a signal for resource prediction. This is due to hospitalised patients generally being admitted to wards that are capable of handling patients in their condition. It is also due to the fact that using machine learning for the monitoring of inpatients for degradation has a very rich literature and would require a review of it's own [89]. As a result, we only focus on works that explicitly predict admissions or patient transfer.

ICU Transfer

By far the most popular type of prediction to make in the inpatient setting is predicting admission of a patient to the ICU. This is due to the fact that the ICU is a resource-intensive area of the hospital and any way of informing the planning of this unit is beneficial to the running of the hospital [22].

A logistic regression is used in [139] to show that it can be predicted that a patient will need admission to the ICU 16 hours ahead of time. Furthermore, they demonstrate this using data from three separate institutions, in order to validate their model. A similar investigation is carried out in [116] in a tertiary care hospital but they consider readmissions to the ICU in 48 hours. This is also explored by [113] who develop a "Bayesian belief system" to predict admission to the ICU, but this time 9 hours before it is requested by the clinician in charge. An NLP approach has also been investigated in [170] where the online messages of doctors and nurses to each other are used in

order to predict patient transfer to ICU 3 days prior to the event taking place. It should be noted that for all of these studies, the outcome being predicted is different and so the studies cannot be compared.

Echoing the narrative presented in Section 2.4, many researchers have also considered predicting readmissions of inpatients to the ICU. This is seen in [159] where the authors investigate which patients, who were previously in the ICU, will be readmitted from their inpatient ward. To predict this they use a gradient boosting machine with features derived from the electronic health record of the patient as well as various blood tests that were taken. A time-series approach to this prediction was investigated by [174] where an LSTM was used and trained on the ICD-9 embeddings of the patients who had previously been admitted to the ICU, their demographics and the chart event features of the patients. They show a strong prediction accuracy when considering if a patient will be readmitted to the ICU within 30 days of their discharge.

Once again, conditioning the dataset on the demographic in question is utilised for the inpatient setting. In [160], they demonstrate, using adaptive and gradient-tree boosting, that they can predict the transfer of a child to the paediatric ICU 8 hours preceding the transfer. The prediction of transfer to paediatric ICU is also carried out in [86] where a logistic regression is used to predict their transfer within the first 24 hours of their inpatient status.

We again see works where the datasets (and therefore the models) are conditioned on the co-morbidities of the patients. In [173], they condition their dataset on patients who have undergone cardiac surgery and predict whether these patients will be readmitted to the ICU. They use a logistic regression with L1 regularisation to provide interpretability to their model, but also use a causal inference method to compare their findings. They find that there is little agreement between the two methods of feature importance ranking.

2.6.2 Resource Management

During a patient's stay in hospital, various tests may be requested to help clinicians gain a better understanding of the patient's condition. These tests are also an important part of the patient flow process and timely testing helps to improve flow through the hospital. An example is seen in [110] where the authors investigate whether or not they can predict if inpatients with traumatic brain injury require a CT scan using 'cost sensitive' random forests. In doing so, they aim to create a prioritisation system for scanning, which will allow faster treatment of patients and therefore a better patient throughput.

Another way in which resource management has been tackled with machine learning is in the scheduling of laboratory samples that need to be processed [189]. Again, by scheduling these samples in an efficient way, this allows patients to be treated more quickly in the hospital, and in some cases prevents the unnecessary hospitalisation of a patient.

These examples can be seen as assessing the risk of resource utilisation on a patient-by-patient basis. A more high-level view is used in [112] where all resource is pooled together (anything including staff or use of machinery). Random forests are used to perform regression on the expected resource use in the next 30 days. While this has limited use to clinical staff due to the lack of granularity, it may be useful for budgeting purposes.

2.6.3 Hospital-Wide Flow

There are very few works that seek to predict the full patient journey through a hospital using machine learning. This may be due to the fact that transfers of inpatients are generally quite rare due to most inpatients being admitted to a ward that is capable of providing the appropriate care for them. [142] treat the hospital journey as a point process. They use a generalised linear model to predict the next location a patient will

be transferred to as well as the dwell-time in that unit. They utilise the MIMIC-III dataset [107], which is an ICU-based dataset and so the transitions they predict are between various types of intensive care unit. However, in terms of predicting the inpatient journey, this is a promising direction. Expanding to the entire hospital, it is possible to predict movement of patients between wards as well as for the use of machinery. Also predicting the dwell-time will allow for better planning of the flow of patients.

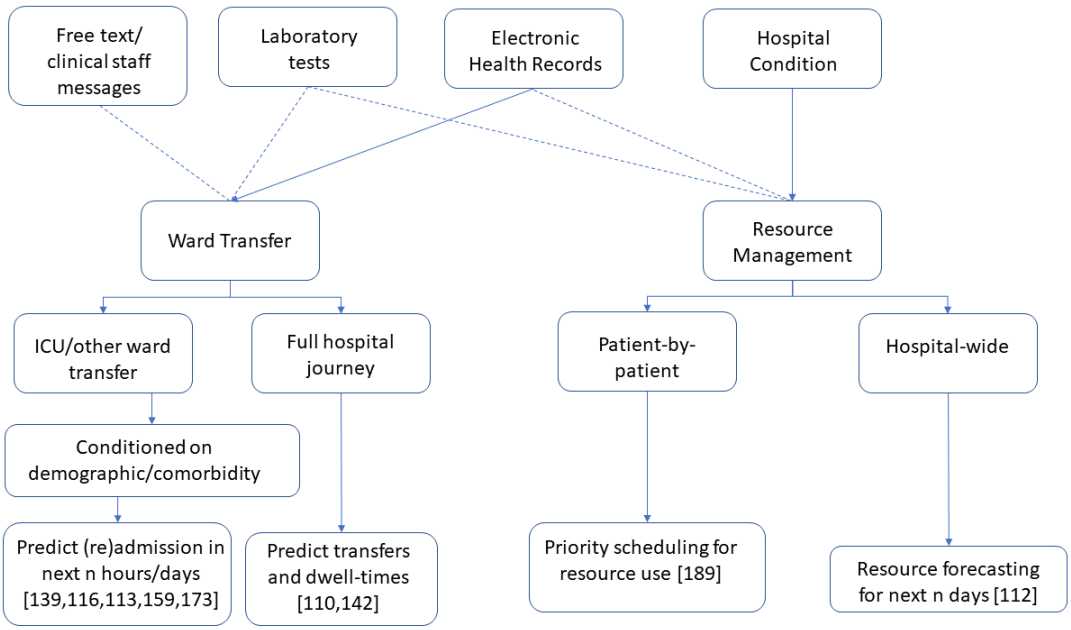


Figure 2.6: Visualisation of the studies carried out on using machine learning to aid in the inpatient journey. Dashed lines indicate some studies opt to use these features.

Table 2.3: Table showing the popularity of different methods and data availability for each of the intra-hospital transition/resource allocation problems.

	Intra-hospital prediction		
	Ward (re)admit	*Transfers*	*Resource forecast*
Labelled dataset readily available?	✓	✓	✓
Regression methods popular?	✓	✗	✓
Classification methods popular?	✗	✓	✓
Point processes popular?	✗	✓	✗
Bayesian methods popular?	✓	✗	✗

2.6.4 Interim Summary

Of the four parts of the hospitalisation process that we have defined, the inpatient setting is the one in which machine learning has been used the least. The majority of studies investigate patient transfer to the ICU due to the resource-intense nature of ICUs. There have also been limited attempts at utilising machine learning to predict the expected resource that will be required by a hospital, either as a whole, or on a patient-by-patient basis. Very few works again have attempted to predict the whole hospital journey using machine learning. A common inconsistency throughout the literature is the prediction lookahead time that is considered (the amount of time prior to predicting that an ICU bed will be needed). Standardising the lookahead time will allow studies to be more comparable and again, crucially, build upon previous work to further improve and integrate the field.

As the vast majority of studies are conducted with a clinical need in mind, this may reflect that the inpatient journey is not seen as a very important part of the patient flow problem. Figure 2.6 shows the structure of the studies that have been carried out in this area of patient flow. Table 2.3 shows the data availability for these prediction problems and popular methods used to tackle them. In order to provide a more complete picture of the works that have been conducted in the space of machine learning in the inpatient setting, Figure 2.7 provides a flow chart including the models and datasets that have been used to make the predictions.

2.7 Discharge Prediction

The importance of discharging patients in a timely fashion for patient flow cannot be over-stated. Long patient stays incur greater cost to the healthcare institution and reduce capacity for new patients to be admitted [41], [52]. As a result, a standard metric of the quality of care being provided is the patient length of stay (LOS) [35]. Patients who are admitted for long periods of time (either due to condition or due

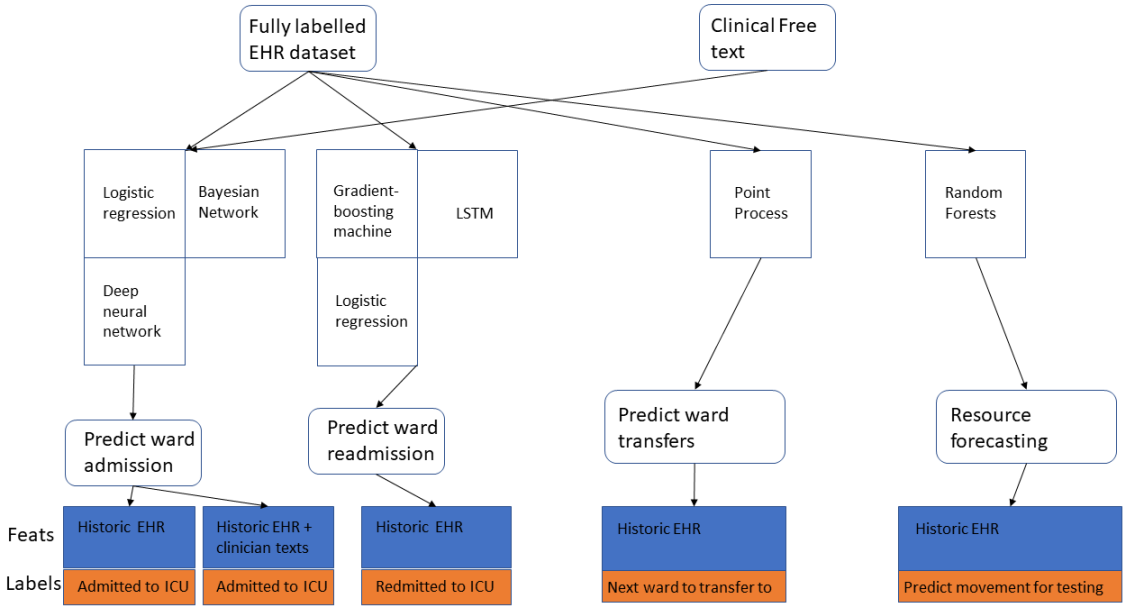


Figure 2.7: A flowchart showing the models that have been used for the separate prediction problems for predicting inpatient transfers. The top row shows the sources of data used, the row below shows the models that have been used in various different works for the prediction problem, below this is the problem being tackled and below each of these problems is one of the datasets used in a study to train the model.

to having no appropriate discharge destination) are commonly referred to as 'bed-blockers' and can constitute up to 20% of the hospital population [3], [10], [66]. Early recognition of the patients likely to have a long LOS should therefore allow for the planning of their treatment by the hospital, such as their admission to long-stay wards and beginning preparations for their discharge.

It should therefore be unsurprising that many researchers have sought to employ machine learning in order to predict the LOS of patients in order to provide hospitals with a better idea of how much resource will be required for patient stays. Note that the prediction of LOS or discharge are essentially the same as they both aim to predict when a patient is able to leave the hospital. We will refer to both types of prediction simply as 'discharge prediction'.

Discharge prediction can be separated into two separate subcategories for emergency and inpatient settings. In the emergency context, predicting the LOS of patients helps to understand whether the ED is at risk of overcrowding or not. In the inpatient

setting, predicting the LOS is useful for the planning of patient admissions and preparation of post-discharge care should it be needed.

2.7.1 Discharge in the Emergency Department

Discharge from ED has been treated as a classification as seen in [211]. The authors predict if a patient will be in the ED for longer than 4 hours or not. They use features that are available early in the ED process to train a decision tree binary classifier. This approach is mimicked in [183] where various learning algorithms are experimented with to classify patients according to their length of stay in the ED. The large imbalance there tends to be in LOS datasets is acknowledged in [87] (with far fewer patients having long LOS), and the authors present an ensemble method combined with multiple logistic regression to overcome this imbalance. However, in this work they define a long stay as patients in the ED for longer than 14 hours.

Rather than classify patients according to their likely LOS category, some authors prefer to use regression to predict each patient's LOS in the ED. Linear regression models are used in [79] to predict the likely LOS of each patient presented to the ED. Quantile regression is instead used in [49] but once again for the prediction of LOS in the ED. Feedforward neural networks have also been used for regressing the likely LOS of patients [91]. One advantage to this approach of regressing the probable LOS is that there are no longer inconsistencies between studies on what is defined as a long-stay. However, this approach is also more difficult to train and achieve an accurate model in practice.

2.7.2 The Inpatient Setting

Predicting the LOS of patients in the inpatient setting is significantly more popular as a research area than in the emergency setting. This may be due to a prediction of LOS in the ED being less actionable than in the hospital, where preparations can be

made to ready a patient for discharge.

A hospital-wide approach is adopted in [83] where a regression tree is used to predict the LOS of patients admitted to hospitals in Pennsylvania using data that is available at the time of admission. This approach is also applied in [60], this time using a feedforward neural network to regress the LOS. These predictions are carried out at the time of admission. An alternative approach is to implement a classifier every day before discharge and predict the patients who can be prioritised for discharge as seen in [102]. In framing the problem in this way, the authors exploit a static model for a dynamic problem by repeatedly applying the algorithm prior to discharge sessions at the hospital. They use a classification decision tree to prioritise patients ready for discharge.

Predicting discharge has also been approached as a time-series problem. In [156] an autoregressive integrated moving average model is used to incorporate a time-series of seasonal data to predict hospital discharge volume. They compare this with using Prophet [164], an additive regression model developed by Facebook Research for forecasting seasonal trends, for the same task. An NLP approach has also been used where the clinical notes from the ED are used in order to predict if a patient will be admitted to the hospital for more than 2 days [193].

As has been a common theme throughout this review, discharge predictions are also conditioned on patient demographics. In other sections this is primarily to improve predictive performance amongst patient subgroups. However, in discharge prediction this is due to certain patient subgroups being more likely to be 'bed-blockers' such as elderly patients [152]. To maximise clinical utility it is more effective to condition the training dataset on these subgroups and apply the algorithms to these patients only. An example is in [105] where a regression forest is used to predict the LOS of elderly patients in a hospital and a random forest is used to predict the location of discharge for these elderly patients. These predictions are used in conjunction with a discrete-event simulation in order to simulate the flow through an Irish hospital.

Children are also a cohort of patients in which there can be great variability in LOS. To address this, [197] use a gradient boosted tree to classify whether or not a child will be a long-stay patient in the paediatric ICU (with long-stay being defined as a stay of greater than 4 days). They also use the static model for a dynamic problem approach by extracting features from the time-series of the patient's vital signs and repeatedly feeding these to the classifier. Note that this prediction concerns the LOS within a ward and not the hospital stay as a whole.

As with conditioning on demographics, conditioning on co-morbidities is also done in discharge prediction. In fact, this tends to be the most popular form of setting the problem due to patients with different ailments and treatments generally requiring different recovery times.

One such prediction is carried out for patients with congestive heart failure in which the authors apply a static cubist model [9] dynamically as data is updated during the patient stay [136]. The model is used to regress the likely LOS in hospital of the patient.

Further discharge predictions have been carried out on patient cohorts who have suffered from stroke [114], patients who have suffered hip-fracture [106], patients suffering from schizophrenia [204], patients admitted for cardiac care [166], patients post-brain tumour surgery [177], patients who have undergone total hip-arthroplasty [181] and patients who have undergone surgery due to colorectal cancer [99]. In all of these studies, there is no consensus for defining a 'long-stay' patient.

2.7.3 Interim Summary

Discharge prediction is one of the more popular areas of patient flow for researchers to apply machine learning. Discharge prediction has been carried out by either predicting whether a patient is likely to be long-stay or by directly regressing the expected LOS of the patient. It has been applied to both emergency and inpatient settings. In the inpatient setting, studies have conditioned their datasets according to demographic.

There have also been studies that condition their dataset according to the comorbidity or treatment that the patients of interest have undergone.

A clear inconsistency between studies is the definition of a long-stay patient. Having a common dataset with pre-defined long-stay patients will improve the ability of researchers to compare models and build upon previous work. Figure 2.8 shows the structure of the literature published in this field. Table 2.4 shows data availability and popular methods used to tackle the discharge problems. It should be noted that the difficulty with a labelled dataset for discharge readiness is that generally it is not recorded when a patient is ready for discharge but when they actually are discharged. In order to provide a more complete picture of the works that have been conducted in the space of machine learning for discharge, Figure 2.9 provides a flow chart including the models and datasets that have been used to make the predictions.

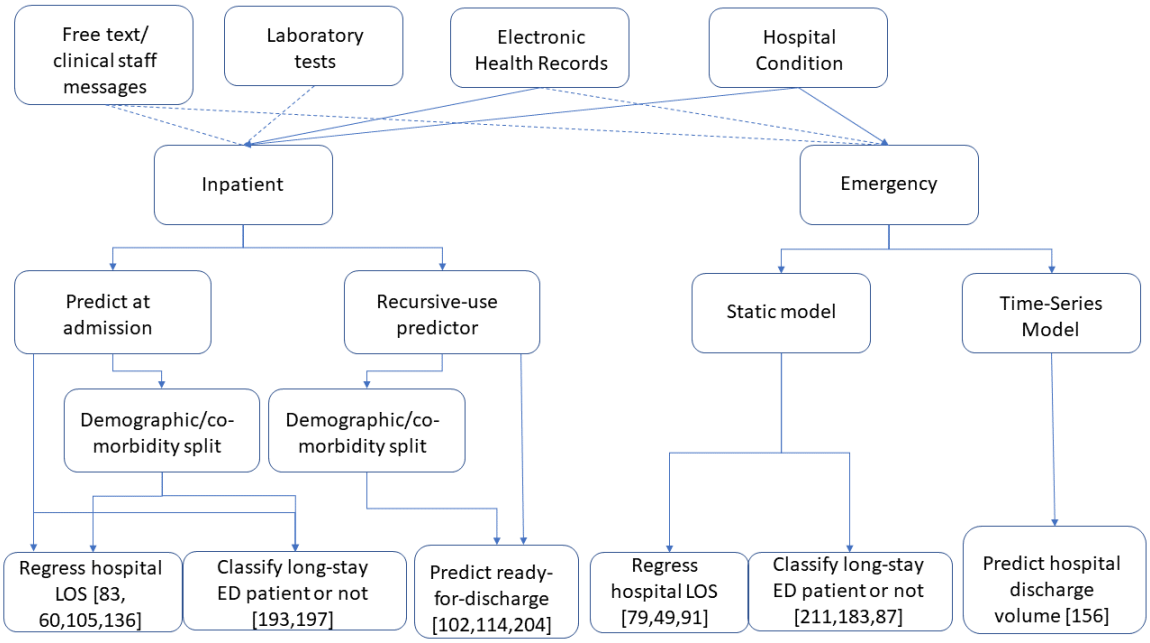


Figure 2.8: Visualisation of the studies carried out on discharge prediction. Dashed lines indicate some studies opt to use these features.

Table 2.4: Table showing the popularity of different methods and data availability for each of the discharge prediction problems.

	Discharge prediction problem		
	LOS	*Long-stay*	*Discharge?*
Labelled dataset readily available?	✓	✓	✗
Regression methods popular?	✓	✗	✗
Classification methods popular?	✓	✓	✓
Point processes popular?	✗	✗	✓
Bayesian methods popular?	✗	✗	✗

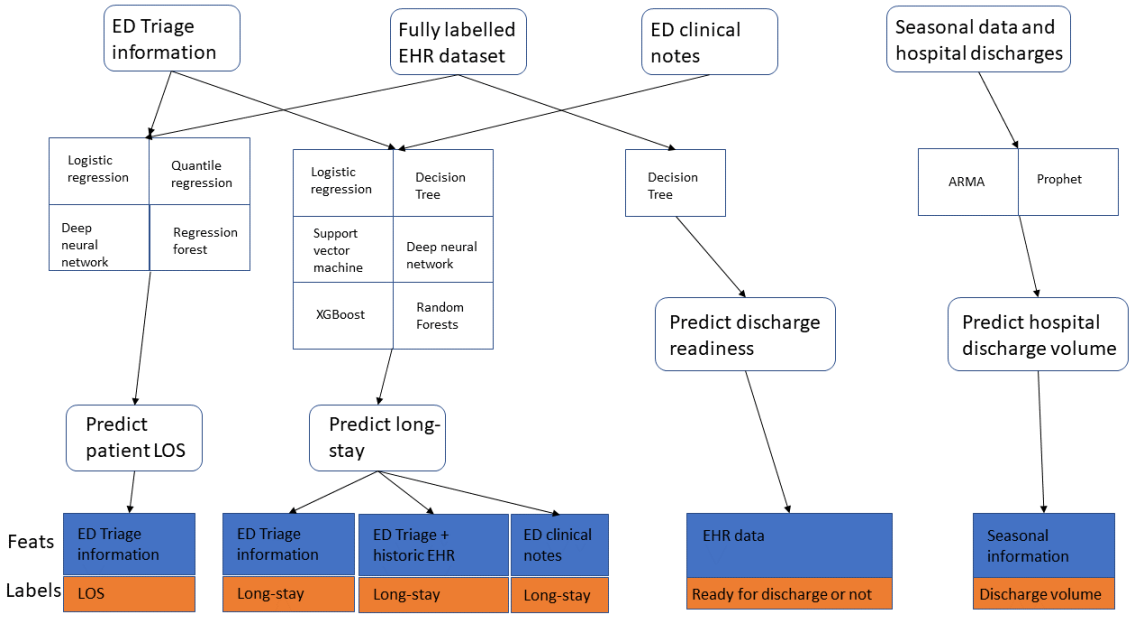


Figure 2.9: A flowchart showing the models that have been used for the separate prediction problems for predicting discharges. The top row shows the sources of data used, the row below shows the models that have been used in various different works for the prediction problem, below this is the problem being tackled and below each of these problems is one of the datasets used in a study to train the model.

2.8 The Future of Machine Learning in Patient Flow

The current research efforts in the discipline of machine learning in patient flow have demonstrated the feasibility and potential of machine learning to optimise patient flow in all of the four subcategories outlined in this thesis. However, due to the difficulty of expanding and scaling machine learning models across different healthcare contexts

and institutions, the current research efforts are still removed from delivering value in the routine and daily management of patient flow in healthcare institutions. In this section, we outline the future research opportunities to advance the applicability of machine learning in patient flow.

2.8.1 Priorities in Patient Flow

While all of the problems outlined in the above review are important for clinical practice, solving some of these problems requires more urgent solutions than others. An example of a high-priority problem to solve is predicting readiness for discharge. One of the greatest problems dealt with in patient flow is the 'bed-blocker' phenomenon whereby patients do not have appropriate destinations to be discharged to. Predictions of readiness for discharge will not solve the lack of space in care homes, however it will allow for more effective allocation of the time and attention of clinical staff.

Following on from this, solving the ED-inpatient interface problem would be the next most important. This prediction would prevent the filling up of the ED due to inability to transfer patients into the hospital. Having an accurate model here would create a more streamlined flow of patients into the hospital, but naturally would depend on there being enough flow out as well. Unlike the discharge prediction problem, predictions here are actionable as clinical staff control the bed space within the hospital.

An equally important task is the prediction of ED attendances. This represents the front-end of the hospital with the discharge readiness status representing the back-end. Being able to accurately predict patient admissions numbers in the ED would allow for accurate planning of staffing rotas, thereby reducing costs and time wasted. It would also greatly improve the care provided for each individual patient.

Finally, the problem that should be least prioritised is inpatient transfer prediction. Despite being important, inpatient transfers are generally quite rare due to patients being admitted to appropriate wards from the outset. However, there is value in

predicting resource flow and patient movement in order to plan that resource.

2.8.2 Current Challenges

Data Challenges

Throughout this review, the need has been emphasised for a common dataset that all researchers can use to benchmark their models and experiments on, as well as have agreed definitions of what age ranges 'elderly' patients fit, amongst other definitions. However, creating a publicly available dataset does not come without its own challenges. The first issue is that of patient privacy. While there are many data anonymisation methods that can be used to remove association of the data with individuals, prior information such as the source hospital can be used to reconstruct the identities of the patients. There then exists a trade-off between how much information is hidden and how useful the data is to machine learning practitioners. A potential solution for this is sourcing data from multiple medical centres and compiling them together in a dataset. This brings us onto the second challenge which is a lack of standardisation in the recording of health data. In order to take advantage of the data from the EHRs from multiple hospitals, we must first stipulate that these hospitals record data in an agreed fashion.

One example of a publicly available healthcare dataset used for benchmarking is MIMIC-III [107]. The success of this dataset can be seen through the volume of works that have used it for model comparison. However, for the purposes of patient flow, this dataset is difficult to use due to its focus on intensive care patients. It therefore does not include the data from the EHRs on the key resource utilisation and patient flows in the hospital (unless they are between intensive care units). A dataset built in a similar fashion to MIMIC-III but with the appropriate patient flow data would benefit the research community greatly.

Technical Challenges

Currently the majority of patient flow models use a specific dataset from a hospital that can be derived from a certain subset of patients. The model is then applied to aid that hospital in prediction, with very few researchers extending their models beyond the specific hospital that they're researching. This approach is limited due to the variable and dynamic nature of healthcare datasets. Distributions of patient admissions and demographics from the same source hospital are subject to issues such as covariate shift whereby the underlying distributions of the features change with time. Examples are the changes to the distributions that can be found in the EHRs of hospitals during flu season or during the COVID-19 pandemic.

Variability also exists across healthcare delivery institutions and organisations ranging from small primary care centres to large tertiary hospitals. These organisations are different in their resources, organisational structure, staff training, and culture. These differences create variability in healthcare delivery practices, organisational processes, and patient flow across these different institutions as well as variability in what data is recorded and in what format it is recorded.

Differences also exist in the distributions recorded by healthcare institutions due to the differences in populations across the world. Examples include the prevalence of different diseases across different communities and geographical contexts (e.g. the presence of type II diabetes mellitus can vary from 3.5% to over 20% across different populations) [111], [150].

Another issue that is faced is the lack of complete information delivered by the majority of prediction algorithms. While it is useful to know that a patient will be admitted to a certain location in the hospital, having some knowledge of their severity or the likely medications that will be needed for them will further help with the planning of their stay.

These issues faced during deployment create challenges in the applications of machine learning, particularly in the generalisation of models to other hospitals and

for their continual use over long periods of time.

In the face of these challenges, we believe that certain research directions will aid future researchers to prepare models that will better serve hospitals to improve patient flow. These research directions should address the issues discussed above, as well as ensure that they integrate seamlessly into the running of the hospital.

2.8.3 Feature Engineering

The majority of studies discussed in this review take advantage of the fact that there exist in many modern hospitals EHR systems which allow data extraction and dataset creation. However, there remain challenges in terms of data collection for the different tasks at hand.

The ED admission prediction relies on seasonal information which can be correlated with admissions but is generally a difficult prediction to make. Wearable sensors (worn in the home setting but monitored by an intermediary care provider) could benefit this prediction greatly, providing more granular information to the hospital. The sensors could also be provided to patients who need them most (and are most likely to be brought to the ED in an emergency, such as elderly patients in care homes).

We believe that further improvements to data collection could be made in the inpatient journey as well as in discharge prediction. Currently, while scans in the hospital are logged on the EHR, the movement of patients to scans are not, and nor is the resource associated in moving that patient. These data would be very helpful to provide a more complete picture of what resource each hospitalised patient utilises and thereby helping machine learning scientists create more accurate predictions of the likely resource needed.

In discharge prediction, one of the challenges is that it is generally not recorded when a patient is medically ready to leave the hospital but when they actually do. Augmenting a dataset with this information could help predict when a patient is ready to leave hospital and in doing so, allow the team looking after them to move their

resource to more vital care, with a more generalised team looking after the patient thereafter until discharge.

2.8.4 Key Modes of Learning

Multitask Learning

The first research direction to be considered is multitask learning, a machine learning method that allows multiple tasks to be learned at the same time. One of the aims is to exploit the learning signals generated by training on one task to create an inductive bias in the model that will allow the effective learning of another task by the same model [13].

Multitask learning can be applied to different problems across the four domains of patient flow to both related (e.g. predicting risks of various in-hospital complications) or unrelated tasks (e.g. predicting length of stay in the ED and predicting hospital admission destination). Once again this relates to the usefulness of having more granular information for clinicians to work with. An example may be when predicting the location of admission of a patient to hospital, also having some prediction of whether the patient is likely to deteriorate or not. This gives better indications of the likely resource requirement of the patient as well as their likely trajectory within the hospital. While this could be done using separate models for each prediction, a single model that can embed an accurate representation of the patient will be more informative and useful to clinical staff. As a result, a key component of this work will be in the development of representation learning algorithms [71], [137] that are capable of representing patient conditions upon presentation to the ED or admission to hospital.

Multitask learning has been applied in many healthcare applications to leverage the shared information across different tasks. Multitask learning is used in [149] to predict major adverse cardiac events, identifying each type of adverse event as a single

task as opposed to having a multiclass classification. This approach is also used to predict prescription patterns for various drugs that are given to similar patients [141]. Multi-task learning has also been used in medical imaging. [171] have used multitask learning in the detection of abnormal nodules on chest CT scans for lung cancer screening. They jointly train their model to segmenting potential abnormalities and identify the presence of a nodule in the region of interest. This is further evidence of how more granular information from the model can provide clinicians with better insights into the condition of the patient.

Transfer Learning

Transfer learning is based on the principle of knowledge transfer across different machine learning tasks and models. It is based on the notion that knowledge gained by the algorithm when trained to solve a particular problem can be stored and applied to solve another related problem, which means it is closely related to multitask learning. This approach includes transferring knowledge from the source domain, D_S, to the target domain, D_T, to help improve the learning of the target-domain task, T_T.

Transfer learning can provide significant advantages in the applications of machine learning in patient flow. It can enable (a) the transfer of knowledge across different tasks and (b) the transfer of knowledge across different populations. The former can help overcome the lack of clinical data for certain problems. For example, one of the barriers to developing effective machine learning tools for COVID-19 patients is the lack of data on COVID-19 patients. A transfer learning approach can provide a solution by using a model that is pretrained on a large non-COVID-19 dataset and adapting it to perform the task of interest in COVID-19 patients. Transfer learning has been used to overcome the lack of COVID-19 imaging data by [208]. They trained a convolutional neural network by using a pre-trained neural network (pre-trained on a dataset of bacterial and viral pneumonia chest x-ray scans) and fine-tuned this using scans from COVID-19 patients. This was done due to the scarce availability of

chest x-rays from these patients.

Transfer learning can also help us transfer knowledge across different populations. This is valuable clinically given the diversity and differences in the genetic predispositions, prevalence of diseases, lifestyles, and risk factors across different populations. Transfer learning is used in [155] to generalise their sepsis prediction algorithm to a new healthcare setting. They trained their prediction model using data from the MIMIC-III dataset (data from ICU patients) and transferred the model to a dataset from the University of California, San Francisco (UCSF) Medical Centre (a dataset of in-hospital patients from a variety of specialty wards). Their transfer learning approach was based on adding incremental amounts of data from the UCSF dataset to the MIMIC training dataset, resulting in better generalisation of the model to the dataset being introduced.

Transfer learning represents an interesting target for future research in patient flow machine learning applications. Transfer learning can be used to generalise models across different healthcare contexts and to overcome a lack of recorded data.

Continual Learning

Continual or lifelong learning refers to the ability to continually learn over time by accommodating new knowledge while retaining previously learned experiences. This approach has the potential to enable machine learning models in the healthcare space to adapt and adjust automatically to new context and settings like a new healthcare context, new patient population, or a new and emerging disease. This has the potential to enable the creation of dynamic clinical AI models that optimise clinical management decision in real time and learn from the continuous influx of information in real world healthcare context. A continually learning algorithm should be an adaptive algorithm capable of learning from a continuous stream of information, with such information becoming progressively available over time. The accommodation of new information should occur without forgetting or interference [179].

However, continual learning represents a long-standing challenge due to the susceptibility of machine learning models to catastrophic forgetting. This phenomenon refers to the decrease in model performance or the complete overwriting of the previously learned information when new knowledge is introduced.

In [205] they highlight the promise of continual learning in revolutionising the applications of clinical AI and leveraging the continuous influx of clinical information to improve patient care. In [184], the authors point out that machine learning algorithms that are capable of continuous learning are a critical future research and translational direction in healthcare AI. They also report that the FDA is considering widening its regulatory framework to include AI-based Software As Medical Device (SaMD) systems that are capable of continuously learning and optimising performance in real-time to improve patient care.

Continual learning promises considerable value in patient flow as it would enable machine learning models to adjust to different healthcare settings continuously and automatically. Therefore machine learning algorithms would be able to absorb the variation across different healthcare institutions and patient populations. Moreover, continual learning may enable machine learning algorithms to continuously learn after deployment to clinical settings gradually improving their performance through use.

2.9 Summary

We have seen in this review that machine learning in patient flow is a vast if disjointed field. There are many works published with the majority focused on the hospital associated with the authors and little by way of comparison to other hospitals or works. We therefore propose the introduction of a publicly available dataset based on the electronic health records of a given hospital. This should include enough information on all four subcategories of the patient flow process (as highlighted previously) and crucially, must have strict definitions for patient types. The dataset should include:

- Seasonal information such as the weather, national holidays and ideally EHR data from multiple hospitals.
- Strict definitions of what age ranges “elderly” or “young” patients fall into for reproducibility and model validation.
- Pre-defined tasks such as ”prediction of patient transfer in 3 hours from time of measurement”. By creating these pre-defined tasks we improve the ability of researchers to benchmark against each others work and develop upon each others models.
- A standardised definition of co-morbidities in patients

We believe that in creating this dataset, a culture of benchmarking on the dataset can be created thereby encouraging researchers to compare their models, build more sophisticated models based on previously published work and crucially provide some external validation to the trained models.

2.10 Conclusion and Motivation

In this review we have seen that there are four main components involved in the predictive patient flow process. These are: prediction of admissions to ED, prediction of movement across the EDii, prediction of movement within the hospital and prediction of discharge from the hospital. Having assessed the contributions to these fields, it can be seen that there remains a gap in the literature addressing predicting the location of admission of a patient upon presentation to the ED. This gap is an important one as addressing this prediction not only offers the opportunity to reduce crowding in the ED (due to waiting for space in the hospital) but also to predict resource requirements well ahead of time which is beneficial for the quality of care of the patient. This is also a much more actionable and valuable insight to clinicians than simply predicting if the patient will be admitted to hospital or not.

We therefore propose an investigation into a model that will predict the location of admission of a patient to the hospital from the ED. Unlike prior work completed in this field, this will predict the location of admission as opposed to simply being admitted or not. This should be a detailed enough prediction so as to inform hospital staff of what type of resource is needed and where it will be needed. This should also be flexible enough that any ward that is capable of treating the patient can receive them during busy periods. Due to the dynamic nature of hospital wards and their function, we choose to train a machine learning predictive model as opposed to a simulation. The model should provide an accurate prediction as early as possible in the patient journey through the ED. Therefore the model should use information from the triage of the patient so that the remaining four hours (on average) of the patient wait in the ED can be used to prepare space for them. In order to maximise this prediction accuracy, investigations into the method with which training the model is conducted will need to take place. The remainder of the thesis will focus on the creation, development and testing the efficacy of such a model.

Chapter 3

The IORD Dataset

3.1 Overview

All of the data used for this thesis were collected from the IORD database (Infections in Oxfordshire Research Database) for the Operational Research on Infectious Diseases (ORchID) project. The extracted dataset spans a period of 4 years (April 2014 - December 2018) and has 616,476 individual patient admission records of which 339,180 are unique individuals. The data is sourced from various sites in the Oxford University Hospitals Trust including the John Radcliffe Hospital, the Churchill Hospital, the Nuffield Orthopaedic Centre and the Horton Hospital. Data is also collected from various sites owned by the Trust such as care homes and nursing homes.

The ORchID extract from IORD consists of thirteen tables detailing different aspects of the EHR. These include:

- Diagnoses table - this includes each visit of an individual to the ED, their initial diagnosis code according to the ED physicians and the investigations and treatments that were conducted and administered respectively.
- ED Attendances table - This includes the time of arrival of an individual to the ED, which hospital in the trust they came to, when they left and how they did.
- Vital Signs table - This includes any vital signs that were recorded whether in

the ED or in the hospital. Vitals include blood oxygen concentration (SpO2), temperature, blood pressure, heart rate and respiration rate.

- Ward stay table - this table includes the admissions of an individual to all of the wards used in their stay. Using this table one can follow the movement of a patient through the hospital.

- Investigations table - this table includes all investigations, treatments and consequent diagnoses of a patient during their stay within the hospital (as opposed to their stay in the ED).

- Events table - this table includes all measurements taken during an inpatient stay including things such as oxygen levels, temperature, heart rate amongst other variables.

- Tests table - this table contains information on all the blood tests that have been requested during a patient stay and the returned measurements.

- Microbiology table - contains a list of any microbiology tests requested and the results from these tests for each individuals stay.

- Radiology table - included any radiological tests with a full written report included.

- Prescriptions table - includes the full prescription of drugs to medicate the patient with including the dosage and frequency of use. Also contains the beginning and end of the prescription period.

- Characteristics table - includes personal information on the patient including their age, sex, ethnicity and level of deprivation.

- Episodes table - during a patient stay, the patient is assigned to the care of a consultant. For each consultant the patient is assigned to, this is termed an

episode. The episodes table tracks each consultant the patient was assigned to and each investigation and treatment that occurred during that episode.

- Appointments table - this table contains information on what appointments the patient has in a daycase clinic or at the general practitioners office.

For the purposes of this thesis, we exploit all of the tables apart from the radiology and appointments tables in order to create our feature set. Exactly how this is done is detailed later in the chapter. An important distinction to be made within the database is the difference between the subject (or 'ClusterID' as is termed in the database) and the admission (denoted the 'AttendanceID' within the database). The subject or ClusterID is assigned to an individual and so for first time patients a new ClusterID will be created whereas for those visiting the hospital again, a ClusterID will already exist for that patient containing their entire patient record. The admission ID on the other hand is assigned according to an individual visit to the hospital. A subject can have multiple attendance IDs according to the number of times they have been admitted to hospital or the ED. As the ED is treated as a separate entity to the hospital, the attendanceID between the hospital and the ED are distinct. This allows us to connect multiple patient visits in a time-series and also allows us to consider admissions to the ED which did not result in hospitalisation. In connecting the patient visits we can use historical data obtained from various patients to guide the decision making of our models. Entries are recorded for both emergency and non-emergency patients but non-emergency patients will not have recorded attendances in the ED attendances table.

3.2 Patient Admissions to ED

While the database spans from April 2014 to December 2018, the admissions to the ED table stops at April 2018. This may be due to the nature of the ED being disjoint

from the rest of the hospital and therefore data collection being carried out slightly differently between the two.

Figure 3.1 shows the number of admissions to the ED between April 2014 and April 2018 for all patients in the dataset. The sharp decrease in the numbers admitting in April 2018 are due to the recording being stopped early in April and therefore not considering a full month's worth of patients. The overall average over the years is 3267 ED attendances per month (not including April 2018), which corresponds to approximately 110 ED attendances per day over this period. This provides insight into the consistent pressure that the hospital and its staff are faced with.

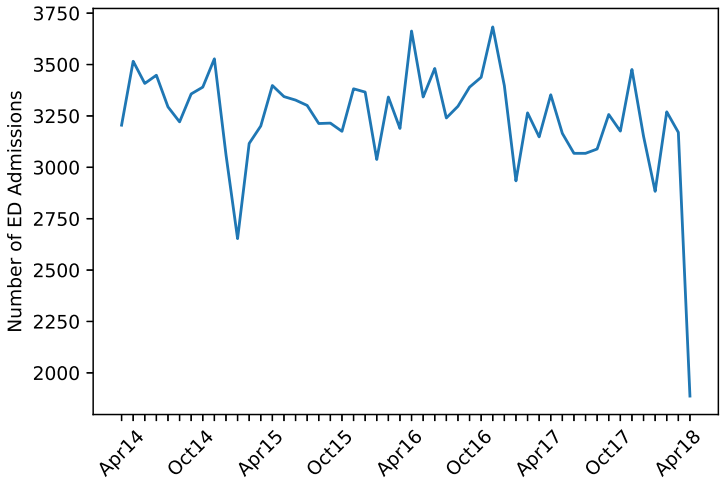


Figure 3.1: Admissions to the ED over the span of the dataset.

Admissions by Age and Sex

In the four years spanned by the dataset there were a total of 161,989 admissions to the ED. It can be seen that there is a 49.6% to 50.4% breakdown of male and female admissions respectively. This considers all admissions to the ED over the four year span of the dataset. While there are slightly more female patients presenting, the difference is marginal and indicates an approximately equal presentation rate between the sexes. This analysis indicates that when creating a model for the admission problem, it is unlikely that a single model would suffer from an inductive bias due to

the prevalence of one of the sexes.

We further investigate how admissions vary by age. We see in Figure 3.2 that, somewhat surprisingly, there is a greater incidence of younger people that are admitted in our dataset. However it is useful to remember that this plot shows admissions for emergency patients only, whereas upon comparison with all hospital admissions (not just emergency), we see that there is generally an older population in the hospital. This is shown in Figure 3.3 for comparison of age distributions.

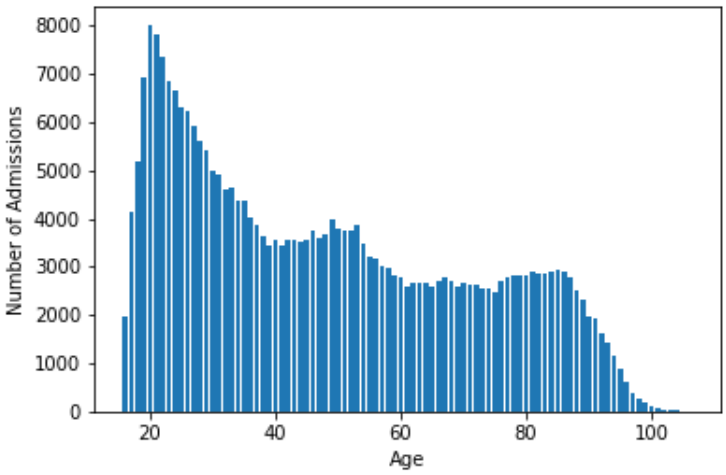


Figure 3.2: Total number of admissions to the ED by age for the duration of the dataset

As a result, we see that using the patients age may be helpful in determining the reason for their admission. It is likely that younger patients will be admitted with issues relating to physical trauma as opposed to more chronic conditions that older patients generally present with. Therefore the age of the patient may be useful in determining their admission location.

Admission Methods

It can also be informative to see where the majority of patients who come to the ED are admitted from. Figure 3.4 shows the distribution of patients who were admitted to the ED and where they were admitted from. It shows that the majority of patients

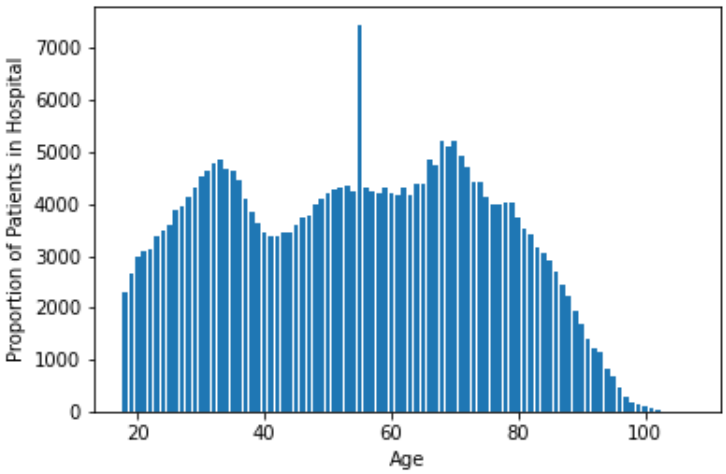


Figure 3.3: Scaled total number of admissions to the hospital by age for the duration of the dataset for comparison with the ED.

being admitted come from their usual homes or where they usually live, which includes homeless patients. In comparison, admissions from other NHS providers including hospitals are care homes are much smaller. This may be due to NHS establishments already providing a level of care, so patients should be monitored more carefully. There is also the fact that during an emergency, it can be difficult to ascertain from the patient where they have come from. As a result, many of the entries in the dataset can be recorded as "from usual abode" as a default response. There are very few admissions from any other sources of admission when compared to admissions from the usual abode, meaning it is unlikely that this will prove a very useful feature in a machine learning algorithm.

Registration

The patient admission to the ED begins with registration at the reception. This is done by the patient themself or by those who have bought the patient in if they are unable to do so themselves. Upon registering in the ED a wristband is printed to document the arrival of the patient and an entry is created in the admissions table. Any historic information about the patient is available at this point in the journey.

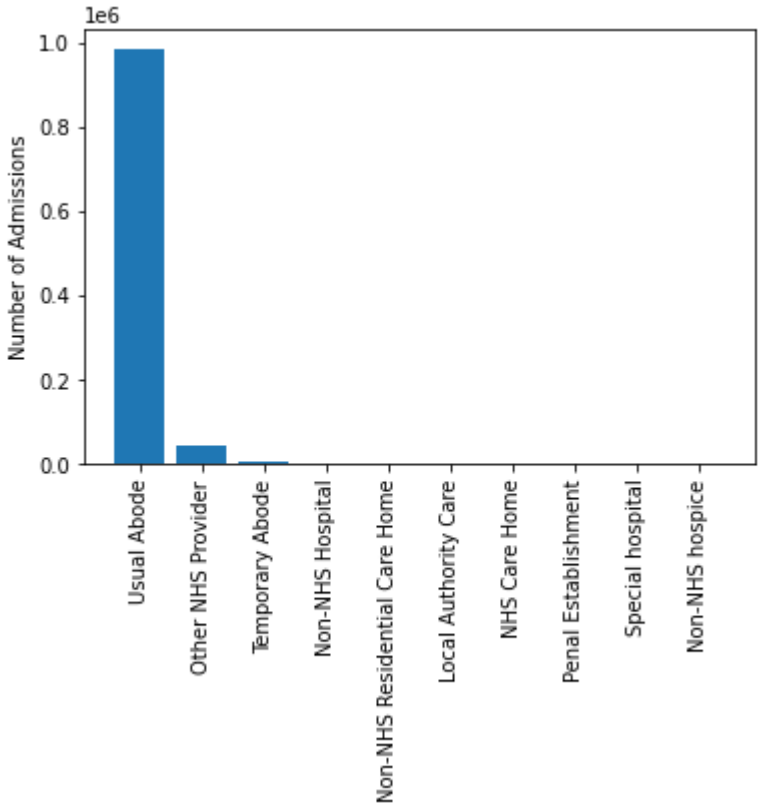


Figure 3.4: Distribution of where ED patients are admitted from. Note that admissions in this context refers also to those who are not subsequently admitted to the hospital.

The patient may then be asked to wait until the triage nurse is free to assess them. For patients in a severe state, the registration process is done for them and they are immediately sent for treatment without being triaged. If the patient's condition has been identified by the paramedics and no further investigation is needed, the patient can also be directly admitted to hospital without the need for presentation to the ED.

Triage

Triage is usually completed within twenty minutes of admission to the ED but can take longer during periods of high patient throughput. During triage a nurse assesses the patient. They initially take vitals measurements including heart rate, blood pressure, oxygen saturation, temperature and take notes on what the patient tells them is the problem.

As can be seen from Figure 3.5, the vital signs are very common measurements to

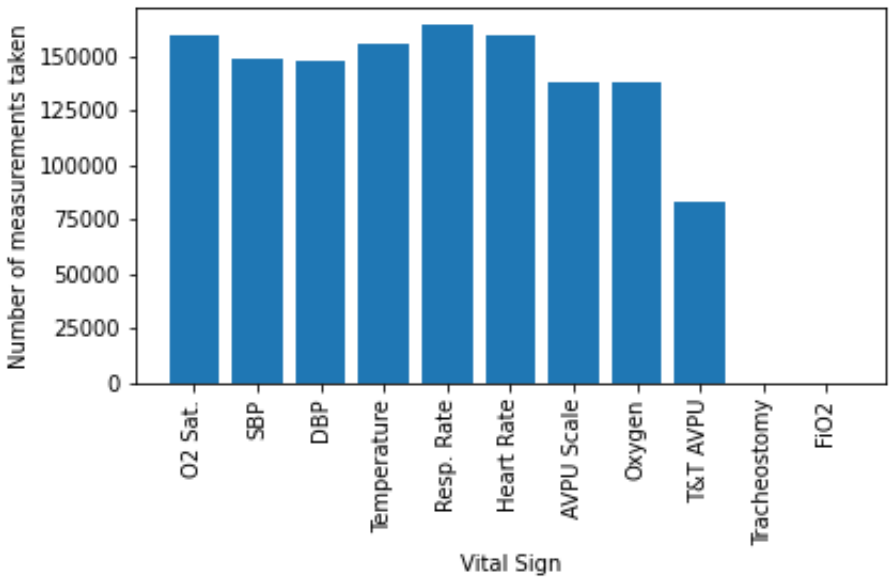


Figure 3.5: Each vital sign and how many patients in the ED have them taken over the span of the dataset. AVPU (alert, voice, pain, unresponsive) is a scale for measuring the conscious state of a patient.

take for patients being admitted to the ED. This makes them very useful as features considering the majority of patients will have these measurements taken, meaning a machine learning algorithm using these measurements would be easily implemented in practice. We see that the vitals that are recorded are: the oxygen saturation of the patient (recorded using pulse oximeters during triage), the systolic and diastolic blood pressure (recorded using a separate blood pressure cuff), the temperature of the patient (recording with a probe in the ear), respiratory and heart rates (also recorded using the pulse oximeter), the oxygen being delivered to the patient (note that the high incidence here is simply indicative of a measurement being recorded - the majority of which are 0 litres/min being delivered), "track & trigger" of the conscious state of the patient, monitoring of a tracheostomy mask and the concentration of any oxygen delivered (FiO2).

The triage nurse is also capable of requesting certain tests to be done such as a toxicology test, pregnancy test or a test for the detection of cardiac enzymes if a patient is complaining of chest or stomach pains. If the triage nurse is unsure then

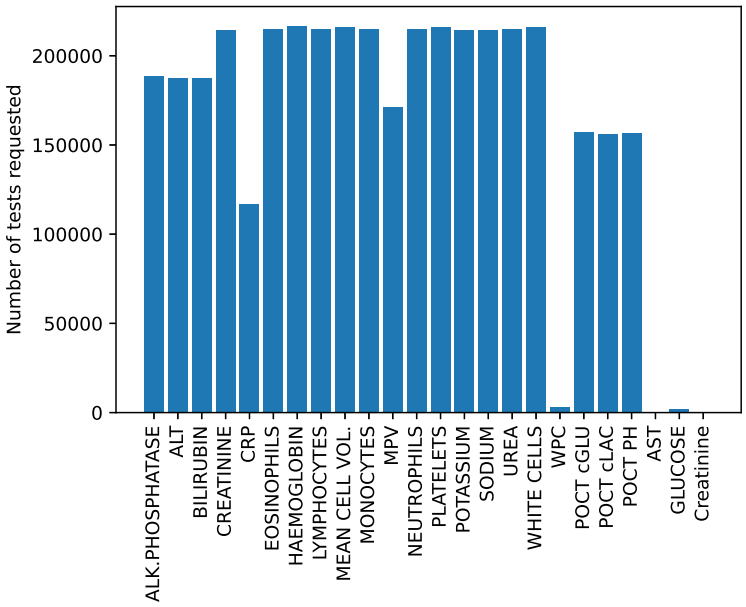


Figure 3.6: The number of each blood test that has been requested in the ED over the span of the dataset. POCT is an abbreviation for a 'point of care test'. It should also be noted that these tests are not generally requested individually but come as a suite of results from a specific type of lab test that is requested.

these tests can be requested by the ED physician later on when the patient is seen by them, but ordering them during triage reduces the waiting time and produces results earlier, allowing a clearer picture to be painted for the ED physician. Generally, a triage nurse will order the tests that they believe appropriate to investigate the patient's condition in order to provide the physician with as much information as possible to treat the patient. Figure 3.6 shows some of the blood tests that can be requested during an ED admission and how often they are requested in relation to one another.

Length of Stay in the ED

Another important piece of information that is provided by the dataset is how long patients have spent in the ED. This correlates strongly with how crowded the ED is and/or describes what capacity the hospital is at in order to receive patients. Figure 3.7 shows how the length of stay is distributed for the patients who visited the ED. The mode length of stay is at 3 hours and 50 minutes, which is just before the 4 hour

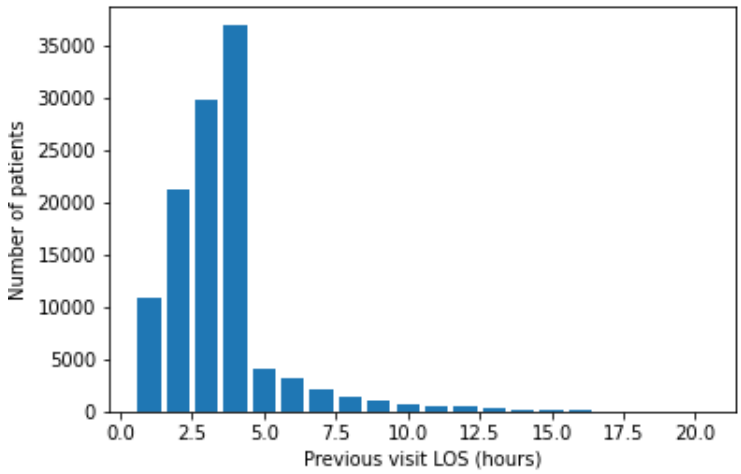


Figure 3.7: Length of stay (LOS) in hours of ED patients.

target that has been imposed on hospitals. This is indicative of patients generally needing longer to be processed than 4 hours, with a last-minute admission to hospital in order to protect from breaching the 4 hour target. While this satisfies the target, it suggests that the process of admission is reactive. It is possible that either the ED is over-capacity which would prevent timely processing of patients or that the hospital is at capacity and incapable of taking more patients in. Both scenarios would benefit from a tool that could plan the trajectories of patients at an early stage in the patient journey. This analysis also suggests that predicting the trajectory of a patient into the "emergency assessment unit" (EAU) is unlikely to be useful. The EAU is a ward that is positioned adjacent to the ED. As a result it can be used as an overspill area for the ED. Learning from the characteristics of patients who are admitted to this ward is therefore unlikely to be of use due to the highly general nature of cases that are admitted to the ward.

3.3 Patient Admissions to Hospital

The next important table found in the database is the 'Ward stays' table. This details the timestamp for every admission ID to a particular location or ward. There is also

a timestamp for exit from that latter ward. Where there are multiple entries for the same admission ID, these can be concatenated to display a journey through the hospital, with the earliest timestamps being the initial wards. Unfortunately, the exit ward timestamp is generated at the same time as the timestamp that is assigned to entry to the new ward so there is no means of determining if there are long waiting times for patients in between ward transitions (i.e, waiting in corridors as is often portrayed in media outlets). However for the purposes of predicting the next location and LOS within the wards, these data are essential.

It is important to reiterate here that the ED is seen as a separate entity to the rest of the hospital by clinicians. An admission for an emergency patient is only logged on the database once they are sent from the ED to another ward within the hospital for treatment or for recovery. It is also possible that an emergency patient can be admitted immediately to a hospital ward and bypass the ED. This can be the case for certain types of patients that are collected by ambulances. For example, patients with chest pain might be admitted directly to the Coronary Care Unit.

We begin by inspecting the general state of transfers within the hospital. Figure 3.8 shows a plot of every ward listed in the IORD database, numerically encoded for ease of display, and the number of patients transferred into these wards (and where transfers can occur either as an initial location or from arriving from another ward within the Trust).

As can be seen from Figure 3.8, the John Radcliffe (JR) Hospital is the biggest in size in the OUH NHS Trust with the greatest number of wards, as well as being the busiest with the greatest number of patients transferred into the wards. The Churchill, Horton, and Nuffield Hospitals are significantly smaller, and "Other OUH sites" comprise barely 4% of all transfers. The busiest ward is clearly "J-WD EAU", which is the Emergency Assessment Unit within the JR. This is the ward to which patients are generally admitted after being seen in the ED. As previously mentioned, it can also be used as an overspill area to lighten the burden on the ED. The only

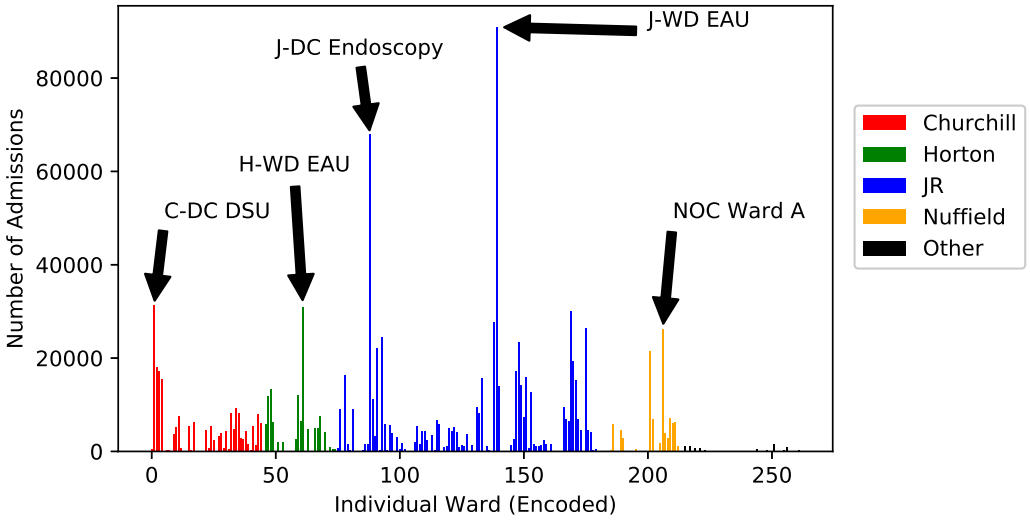


Figure 3.8: Histogram displaying the count of how many patients were transferred to each ward within the OUH Trust between January 2013 and March 2017. The labels are the encoded names of the individual wards.

hospitals with EDs are the Horton and the JR. It is clear from Figure 3.8 that the EAU receives the greatest number of patient transfers in any hospital with an ED. The next busiest ward is "J-DC Endoscopy", where the "DC" stands for Daycase and "Endoscopy" is the ward in which endoscopies are carried out on patients. As this is a day unit, all admissions to this ward will be pre-planned; such admissions therefore will not be used in order to create the prediction model. Generally speaking, admissions to Day Case Units are always pre-planned, and as previously discussed, in order to provide a useful model we will not consider the prediction of the movement of pre-planned patients.

In Figure 3.9 we repeat Figure 3.8 only considering patients who were admitted in an emergency and:

now see that considering only emergency patients acts as a filter for identifying those wards that are generally used for emergencies. This is important because we aim to produce a predictor that does not provide a clinician with redundant information (e.g., predicting that a pregnant woman will be admitted to the maternity centre.)

Figure 3.10 shows the distribution of admissions by ward within the JR. It can

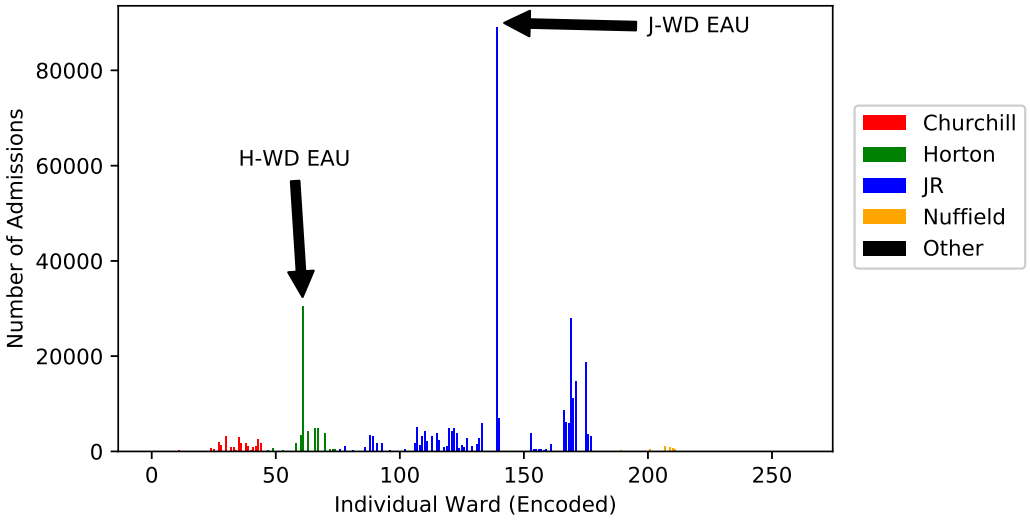


Figure 3.9: A histogram now displaying the total number of patients transferred into the wards (WD) when considering only patients who have been admitted in an emergency. It can be seen the emergency assessment unit (EAU) is still the dominant ward for transfers in the John Radcliffe (JR) and Horton (H) hospitals.

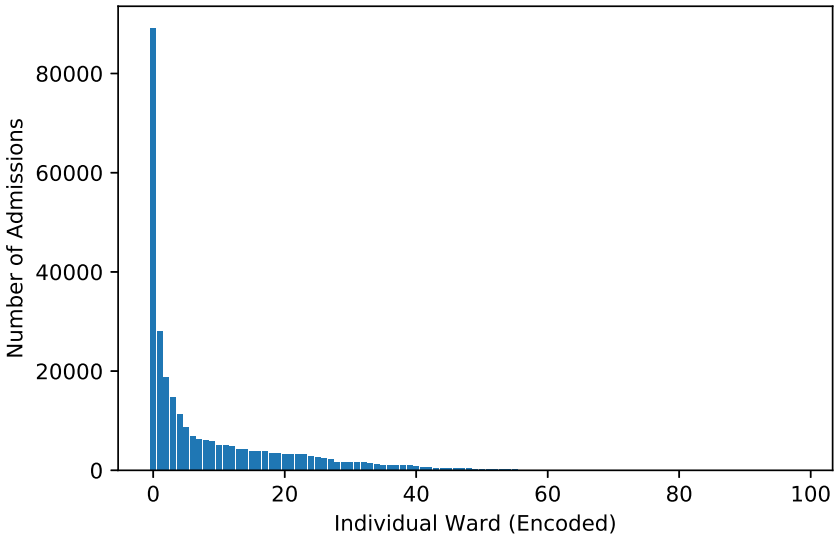


Figure 3.10: The distribution of admissions to the JR by ward. Wards have been organised by number of admissions in order to show the imbalance in the data that is present. The first ward is the EAU and the second is the Surgical Emergency Unit (SEU)

clearly be seen that there is a large imbalance in the admissions, with the EAU dominating the distribution followed by the Surgical Emergency Unit (SEU), which is the surgical equivalent of the EAU. There are almost 85,000 instances of patients

being admitted to the EAU and approximately 30,000 of patients being admitted to the SEU.

Due to the imbalance in the number of patients admitted to the EAU and the SEU, we must consider the admission to the EAU and SEU as separate to admissions to the rest of the wards in the hospital. Comparing the number of admissions to the EAU versus admissions to anywhere in the hospital balances into approximately equal numbers. If we remove the EAU admissions, we see that the SEU now dominates the ward admissions with SEU admissions comprising approximately 30% of this new dataset. As there is a great imbalance between the EAU, SEU and the rest of the wards, a practical system would need to either not consider the first two wards, or make separate classifiers for these two prior to predicting amongst the remaining wards.

Due to there being no clear dominant ward for admission after discarding the EAU and SEU, wards can be grouped into "ward-types" to reduce the number of possible output destinations. This is due to there being 108 wards in the John Radcliffe hospital alone, many of which have very similar functions but whose use is not as prevalent on the dataset as their 'sister' wards. This problem then becomes sparse due to there being few wards that are frequently admitted to. Aside from sparsity, there is also the fact that many of these wards are run by the same departments who are in charge of resource expenditure within these wards. As a result, by basing a model on the grouping of wards into "ward-types", we reduce the sparsity of the problem while at the same time providing department-level information on potential resource requirement. Combining these together has the advantage of capturing the important information of what departmental resource was required to admit and treat the patient in question.

There is also the problem that the utilisation of the wards throughout the span of the dataset is not consistent. The hospital is a highly dynamic environment, reacting to the demand placed on it. As a result, combining wards into ward types allowed the

appropriate attribution of the ward admitted to to the department that was in charge of resource and treatment in that ward during the time. This labelling was done with the help of our clinical collaborators from the John Radcliffe Hospital.

The task can thus become a multi-class classification. Given a prediction of a ward type, any of the wards grouped within this ward type can be used as an admission location for the patient. This is due to the staff who work in these wards being trained to treat patients with similar conditions. As a result, similar patients will be admitted to wards under the same department and the staff in these wards will be sufficiently skilled to treat them.

A model can therefore be constructed in two ways. The first is to pose our admission as binary classification problem asking whether the patient will be admitted to the EAU or not? Upon classification as a non-EAU case, the second stage of this first approach is then to pose another binary classification; will the patient be admitted to the SEU or not? A negative classification in this case then leads to the multi-class classification of which ward-type the patient will be admitted to. Figure 3.11 is a visual representation of this first approach.

The second approach is to disregard the EAU and SEU entirely. This is due to their function being vague and varying with ED crowding where the EAU can often become an overspill of the ED to prevent patient waiting times of over 4 hours being recorded in the EHR. This second approach would then only need to perform two stages of prediction. Firstly, will the patient be admitted or discharged from the ED? Secondly where in the hospital will the patient be admitted to (i.e., the last stage of Figure 3.11).

3.3.1 Ward Admissions According to Diagnosis

As discussed in Chapter 2, one of the largest bottlenecks in the hospitalisation process is during the transition between the ED and admission to the hospital. This was referred to as the EDii. Figures 3.8 and 3.9 show hospital ward utilisation for all

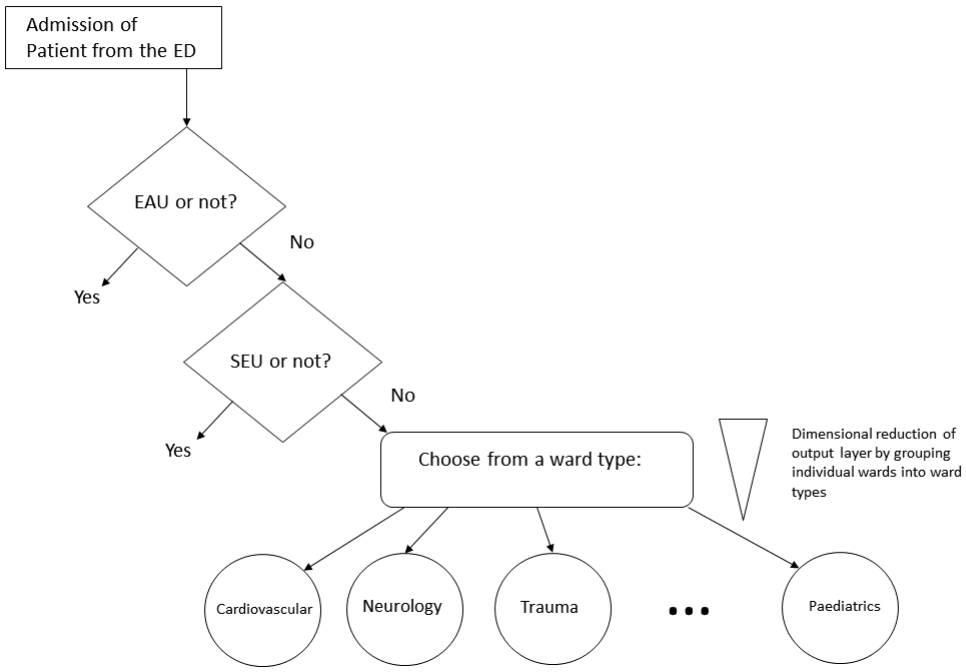


Figure 3.11: Flowchart showing the stages carried out in our prediction algorithm. Note that no children were included in the dataset and that paediatrics is written here as an example of a ward type.

patients and those crossing the EDii respectively. We now investigate how this changes when conditioning the dataset according to the primary diagnosis codes of the patients admitted. This analysis will provide insights as to how patients are admitted to hospital based on their primary diagnosis code and therefore, how resource is allocated according to patient diagnosis code. Figure 3.12 shows the ward utilisation for patients admitted with a primary diagnosis relating to cardiac problems. We see large spikes in wards such as J-WD CAS and J-WD Card (cardiology wards). It should be noted that ICD-10 (diagnosis) codes are generally allocated and registered after a patient has been discharged from the hospital. Therefore using a diagnosis code to predict ward admission location is not possible. However, Figure 3.12 is useful for two reasons. First, it shows that with enough information on an incoming patient that may hint towards a certain diagnosis or condition, the required ward for treatment may be predictable. Second, it shows that patients who clearly have a cardiac problem will be admitted to a ward that treats cardiac problems. Having a model that predicts such

obvious outcomes will not be clinically useful. A setup designed to predict location of admission should therefore use the primary ICD-10 code as a filter when curating the training dataset to make sure a model is not trained on cases that would not require any prediction. Such an approach would provide the greatest utility to clinicians in the ED.

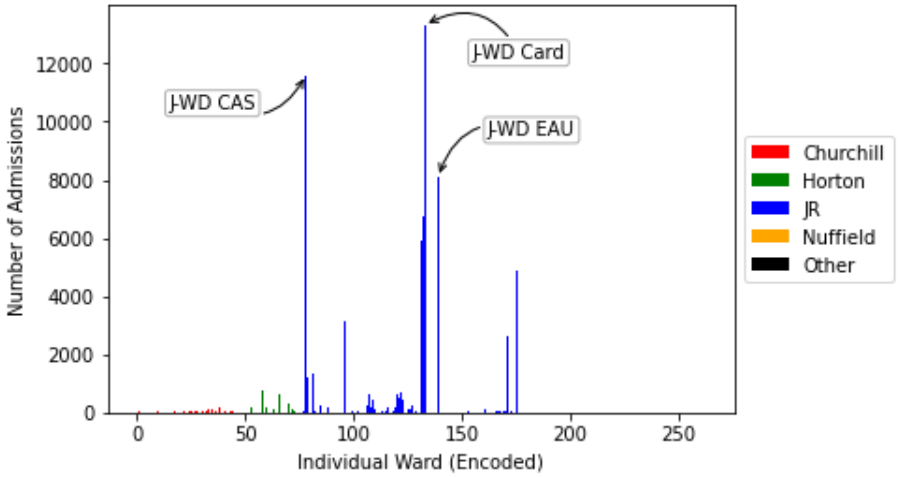


Figure 3.12: Number of ward admissions when conditioning the dataset on patients whose primary diagnosis related to a cardiac condition.

To reinforce the notion that the location of admission is predictable, we condition the dataset on patients with primary diagnoses related to neurological problems, trauma presentations and pregnancy related issues. These are displayed in Figures 3.13, 3.14 and 3.15 respectively. We see once again that specific wards are associated with these various conditions. We also see that there are clusters of wards that get used (as can be seen in Figure 3.15 for childbirth). This is due to these wards being similar in function and therefore being capable of treating similar patients. As a result, it is possible that a classifier being trained to predict admission location of an emergency patient would only need to provide a prediction of the ward function that is required. Space can then be found by staff analysing what space is available in the wards that fall under this functional grouping.

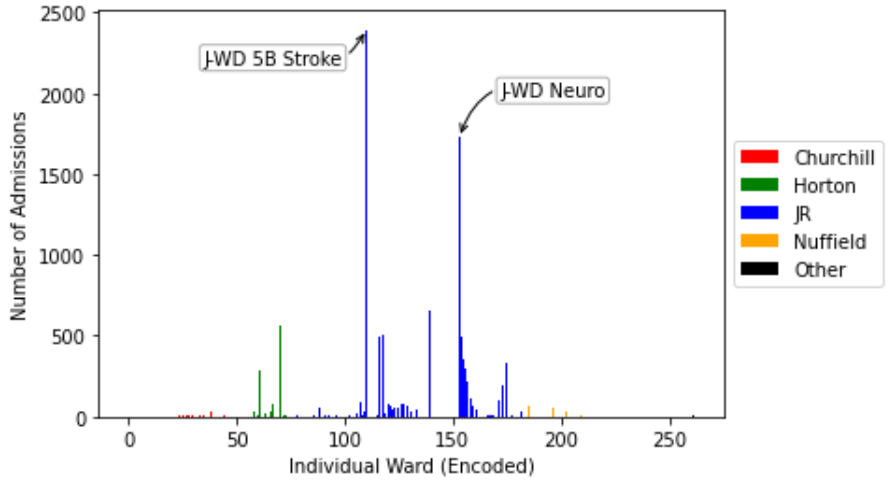


Figure 3.13: Number of ward admissions when conditioning the dataset on patients whose primary diagnosis related to a neurological condition.

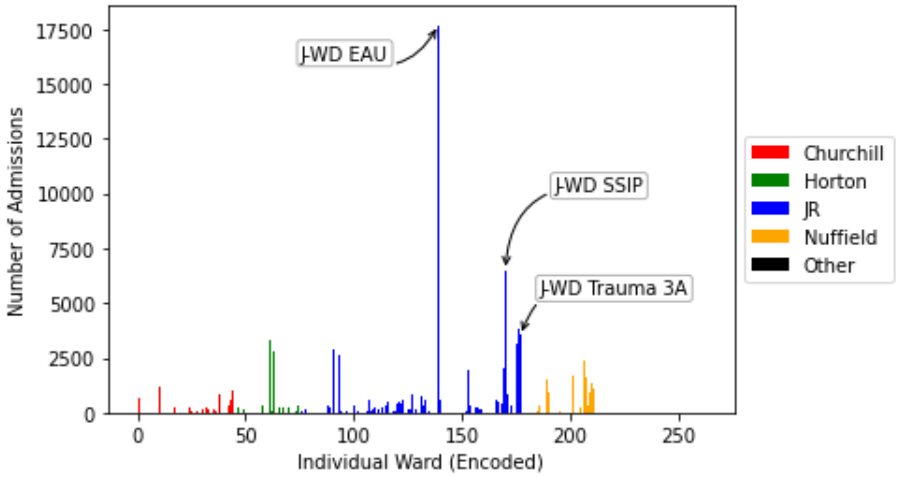


Figure 3.14: Number of ward admissions when conditioning the dataset on patients whose primary diagnosis related to trauma.

3.4 Transfers Within the Hospital

Having assessed how the numbers presenting to the ED change with time, as well as the features that are recorded during the process we then went on to investigate the wards that patients are admitted to across the EDii. In this section, we look at how patients are moved through the hospital once they have been admitted.

In order to carry out this investigation, we create a transition matrix that contains counts of the number of transfers from one ward to another throughout the span of

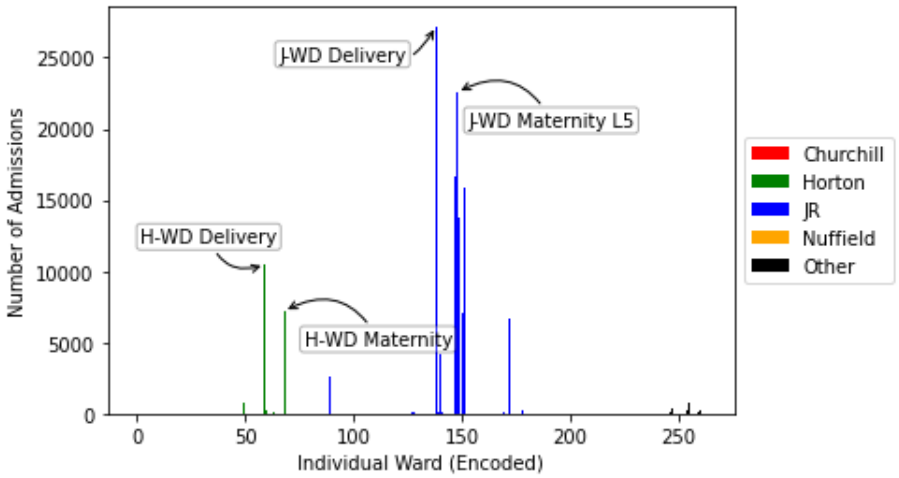


Figure 3.15: Number of ward admissions when conditioning the dataset on patients whose primary diagnosis related to a pregnancy-related condition.

the entire dataset. Figure 3.21 shows the transition matrix for the Horton Hospital. We choose to use this hospital from the OUH NHS Trust for display as it is one of the hospitals with an ED and a smaller number of wards, which leads to a greater clarity of the matrix. We see in Figure 3.21 that the matrix is sparse. The greatest number of transitions that occur is from the delivery to the maternity wards. The next largest number of transitions is patients being moved from the EAU to the general medical wards. We also see a very similar pattern in the John Radcliffe Hospital.

The transition matrix is useful in that we see that transitions in the hospital are generally rare. Furthermore, they are highly predictable. It is obvious to clinical staff that a woman in labour is highly likely to move from a delivery ward to a maternity ward. It is also obvious that moving from the EAU to a general medical care ward is the logical next step in a patient trajectory.

3.4.1 Conditioning the Transition Matrix

Given the sparsity of the global transition matrix, it is unlikely that conditioning the matrix on patient subgroups would provide any greater insight. We conditioned our patient dataset on gender and various co-morbidities of patients including patients with diabetes, hypertension, ischaemic heart disease and traumatic injury. All of the

transition matrices that were generated were of a similar form to 3.21 but sparser, with very few unexpected ward transitions. While it could be interesting to investigate the unexpected ward transitions to understand the operation of the hospital at the time, creating predictions of these unexpected transitions will be of little utility to clinical staff. As a result, we choose to not focus on patient transitions within the hospital.

3.5 Length of Stay

Length of stay is an important consideration for hospital management. It is common for patients to stay longer than required for their treatment which is an inefficient use of resources [1]. In this section we examine the current situation within the OUH Trust in terms of length of stay of patients.

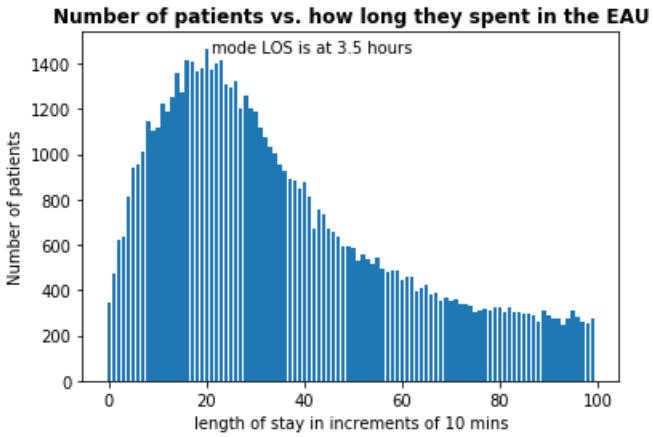


Figure 3.16: A plot of the number of patients who stay in the EAU for a certain time (rounded to the nearest 10 minutes). The mode centres around 3.5 hours duration which is how long the majority of patients spend in the ED.

Figure 3.16 shows the distribution of LOS for patients in the EAU. The mode is at a LOS of 3 hours 50 minutes, and 28% of patients are transferred out of the EAU within this time. It is interesting to note that the most common LOS within the ED is also 3 hours 50 minutes which can be seen in Figure 3.7. This further reinforces the notion of the EAU operationally being used as an overspill for the ED in order to

avoid the fines associated with long ED stays.

In order to assess flow, we must consider the "pipeline" or the route through which patients are sent. We therefore look at the most common destination to which patients are transferred from the EAU. It is found that there are 7 main wards which are transferred to from the EAU. Figure 3.17 shows plots of the LOS within the EAU along with the LOS in the most common subsequent destinations (we only plot for three subsequent wards for clarity).

A key point of interest to note from these plots is the difference in peak values between the EAU and subsequent wards. The mode LOS for the EAU is 3 hours 50 minutes, whereas the mode LOS for the 'short-stay ward' (SSW), ward 5A and ward 7C are all 18 hours. Given a fully occupied hospital, where every bed in a likely patient journey is occupied, there cannot be flow of patients in such a configuration. The only way this could be handled is by increasing "capacity", which in our case would be to use bed space in areas of the hospital not generally used for a certain patient type. This reinforces our approach of using ward-types to generalise the functionality of a ward wherein, if we cannot follow the optimum pathway, we can select the next best destination. This was deemed as being especially helpful by our clinical collaborators.

Another point of interest is the periodicity of the LOS in subsequent wards. They have period 24 hours for all plots. This is reflective of the fact that hospitals are unable to discharge patients overnight and so must wait until 8 a.m. before discharge is possible.

Key Insights

We here summarise the key insights to be taken away from the analysis of these features:

- There is no discrepency between the sexes in terms of number of admissions but there is a trend in age. Both of these can be strongly informative of admission location so will be included in the feature set.

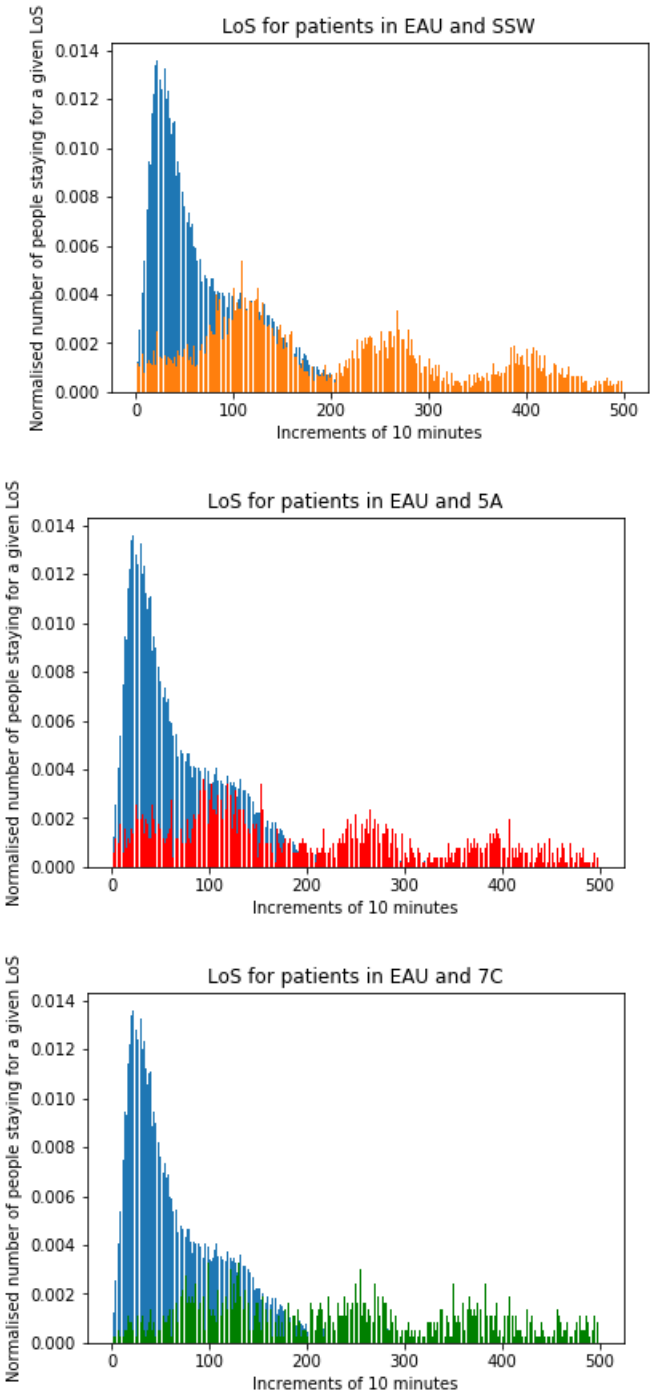


Figure 3.17: Plots showing LOS of patients in EAU (blue) compared to the 3 most common post-EAU wards: (i) SSW (orange), (ii) ward 5A (red), and (iii) ward 7C (green).

- The usual abode of the patient is highly degenerate meaning it is unlikely to help in discrimination and therefore will be left out of the dataset.

- Vital signs are often taken at triage meaning they may be useful for discriminating between patient admission locations and so these will be used.
- Likewise, suites of blood tests are often requested at triage and this may be highly informative for where in the hospital the patient will be admitted.
- The average length of stay in the ED is approximately 3 hours and 50 minutes indicating that the 4-hour target is highly influential over ED waiting times.
- The vast majority of admissions are to the John Radcliffe hospital. Conditioning on ICD-10 codes shows clear patterns in admission locations of patients. Some encoding of historic ICD-10 will therefore be useful to include in the dataset.
- Transfers in the hospital are sparse and therefore predicting transfers in hospital is unlikely to be of benefit to clinicians.
- Average dwell times in the most popular wards after the EAU are much longer than the average dwell time of the EAU. This indicates that without being able to take action on discharge, predicting dwell times is also unlikely to be useful.
- The most likely useful prediction is predicting the location of admission of a patient who is to be admitted to hospital as early as possible in the ED attendance.

3.6 Construction of the Training Dataset

A training dataset must be created in order to train a machine learning model to carry out predictions that will be clinically useful to staff in the hospital. As discussed at the end of Chapter 2, the prediction that is believed to provide the greatest value to clinicians with the data that is available is the prediction of the admission location of a patient presenting to the ED. The transition matrix in Figure 3.21 shows that predictions of intra-hospital movements are likely to be of little use to clinicians. While

predicting discharge will be of great benefit to the hospitals, without appropriate aftercare available for patients, there is little that can be done with such a prediction as seen in Figure 3.17. As a result, in agreement with clinical collaborators, it was believed that creating a dataset for the ward admission location prediction problem would have the greatest impact. Our filtering of obvious admissions to the hospital further encouraged the use of a predictive model capable of learning complex relationships.

To do this, data from early in the ED admission process must be used. This includes data from the triage process as well as any historic information of the patient stored on the EHR. The number of potential features that can be used should be maximised as this will provide a greater potential for pattern recognition over the training distribution.

Having analysed the contents of the IORD database, we now seek to create a dataset that can be used for the ward location of admission prediction problem. To provide a reproducible extract of the dataset that was used for this, we provide in the following sections a step-by-step processing of the original tables discussed previously.

Stage 1

The first stage consists of identifying patients who were admitted to the hospital via the ED. It is checked whether or not the patients have a history of admissions to the hospital (whether via the ED or as an outpatient) and only those with a history are used. The diagnosis table is then checked to find patients with a previous diagnosis code (this is administered at the end of a patient stay regardless if admitted via the ED or directly). The diagnosis table is then merged with the ED admission table according to patient visit identifier. From the ED attendances table we extract the investigations requested (i.e., what tests the triage nurse has requested), whether the patient has a previous admission to the ED and how long ago that admission was. From the diagnoses table we extract the number of historic diagnoses the patient has, the previous diagnosis that they were admitted with and whether or not they are a

frequent admission which we have defined to be having an average of 3 admissions within a year. As can be seen in Figure 3.18 the numbers drastically reduce due to the fact that the majority of ED visits that result in admission are people with no prior history in the hospital (i.e., they have not been admitted to hospital after passing through the ED).

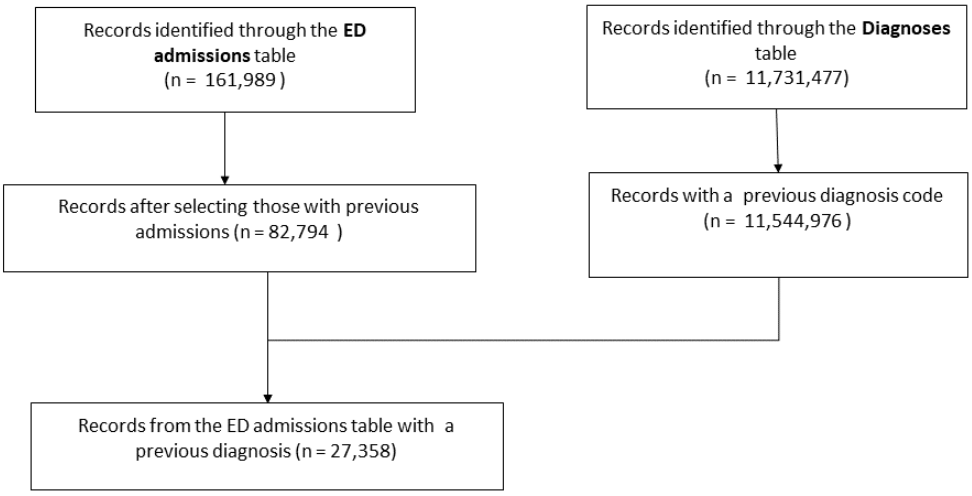


Figure 3.18: Stage 1 of building the dataset that was used for the ward location admission prediction. The ED attendances table is combined with the diagnoses table to extract patients who have been admitted to the hospital via the ED.

Stage 2

In the next stage we load in the table with the vital signs that are recorded during triage. From these we extract the heart rate at entry, the respiration rate at entry, whether it is believed the patient is experiencing atrial fibrillation or not, and whether continuous monitoring of vitals is requested. We then import the episodes table from which we extract the admission method of the patient, the admission source (or where

the patient has been admitted from) and the specialty of the consultant who looked after the patient during their last admission (termed previous management). Once again, we join these tables together so that we may expand the number of features we have to characterise the patient.

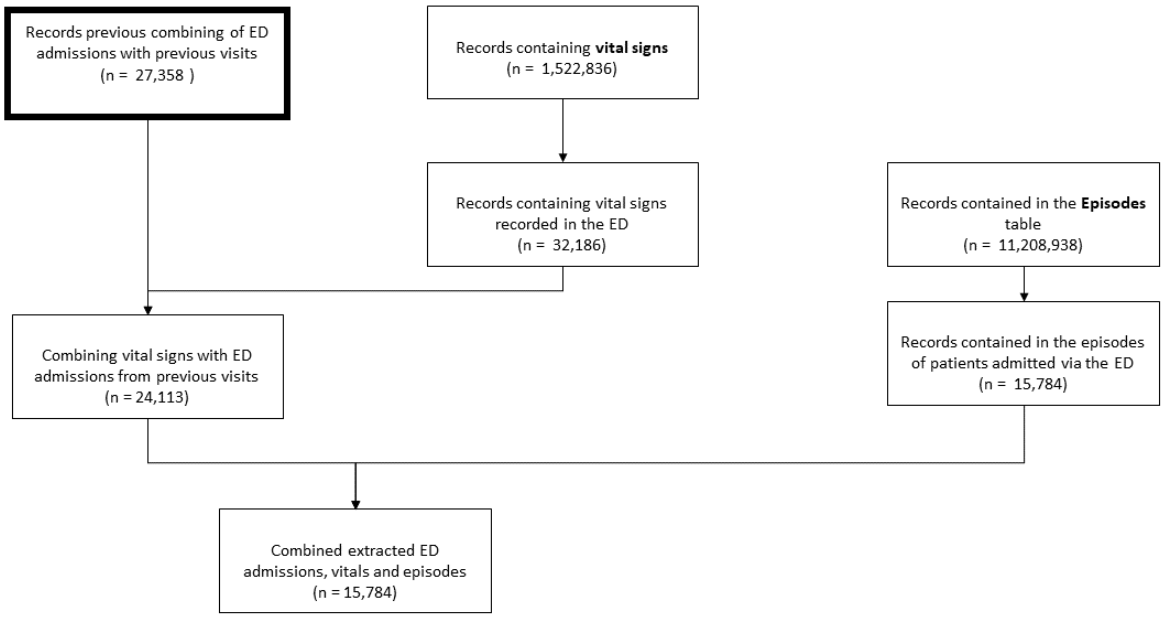


Figure 3.19: Continuation of the dataset build. The box shown in bold contains the table that was derived at the end of the flow of Figure 3.18.

Stage 3

In the third stage of development of the dataset, we load in the patient characteristics dataset where we have access to the patient's age, gender, ethnic category, distance of their home to the hospital and their deprivation score according to the index of multiple deprivation (IMD) [101].

The ward table is then uploaded in order to associate each ED admission with a ward that the patient was subsequently admitted to. Once these wards are grouped together into ward-types, these will act as the labels for our prediction. As discussed

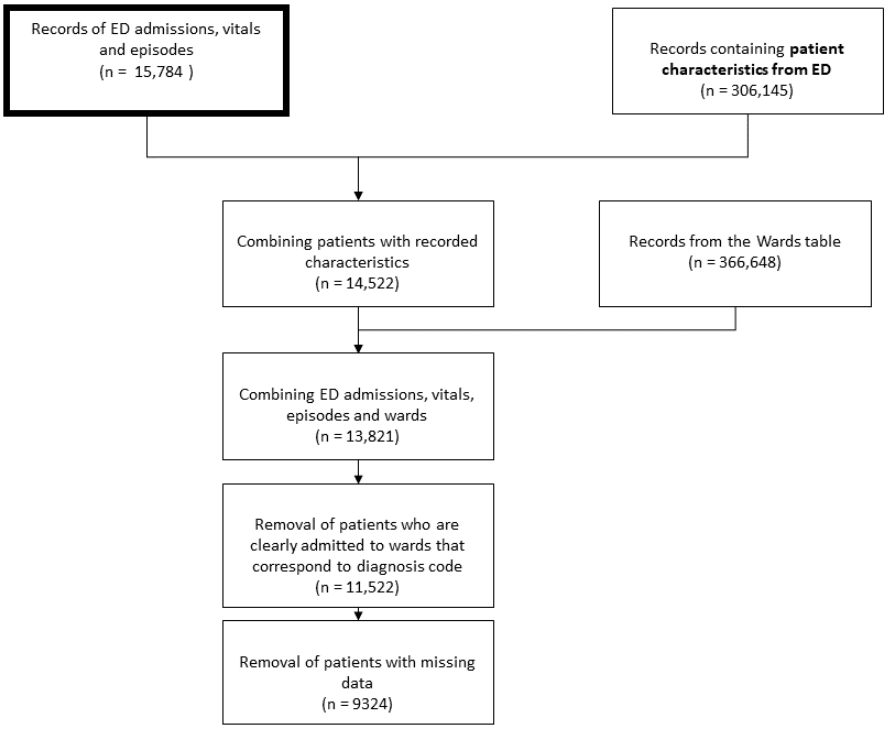


Figure 3.20: Continuation of the dataset build. The box shown in bold contains the table that was derived at the end of the flow of Figure 3.19. Note that the ward table that is loaded in discards patients admitted to the EAU and SEU across the EDii.

previously, we choose to group these into seven groups, namely:

1. Medical wards - wards that take care of patients requiring medical, non-invasive treatments

2. Cardiac wards - wards specialising in the care of patients with cardiovascular problems

3. Neuro wards - wards specialising in the care of patients with neurological problems (neurosurgery is included in this category)

4. Trauma wards - wards specialising in the care of patients who have undergone traumatic injury

5. Intensive care units - wards dedicated to the priority monitoring of patients in a critical condition

6. Surgical wards - wards that specialise in the treatment of patients through invasive/surgical methods

7. General wards and Obstetrics and gynaecology - more general wards that incorporate wards assigned for rest, or specific patient subsets.

At this point we have constructed a feature set that specifies the condition of the patient. However, knowing the condition of the hospital upon entry of the patient is also likely to be informative. Having an understanding of the level of resource utilisation in the hospital at that point may be helpful in predicting the location of admission, as at busier times patients may be admitted to wards other than those used for their condition.

We begin by using the ED attendances table to extract the hour of the day and month of the year that the patient was admitted in. We also extract which day of the week it is as this also has implications on how busy the hospital is. Furthermore, using this table we create a count of how many admissions were registered in the table 12, 8, 4 and one hours prior to the admission of the patient in question. This provides a more temporal representation of how crowdedness in the ED is changing. If the values get larger then the ED is getting busier and this has implications for resource utilisation. In addition to this we add a count of what are termed 'breaches'. As discussed previously, the NHS has a target that all patients in an ED should not wait for longer than four hours prior to being seen by a clinician. In order to assess the crowdedness of the ED we add as additional features the count of patients who have 'breached' in the past 12, 8, 4 and one hours. Once again this provides a representation of the temporal situation and helps us to understand if breaches are increasing or decreasing at time of admission of the patient.

Based on discussions with our clinical collaborators, we also included information on how sunny the day was, the maximum and minimum temperatures and whether or not it had rained that day. This is due to the phenomenon of traumatic injuries

presenting more often in an ED during warmer weather [58]. The full set of features can be found in Tables 3.1 and 3.2.

Table 3.1: Table containing the patient specific features available at initial medical assessment that were used in all of the models.

Bacteriology test requested?	Biochemical tests requested?
Blood cultures requested	Blood gas test requested?
CT scan requested?	Cardiac enzyme test requested?
Clotting study requested	Blood cross-matching requested?
Heart rate at entry	Dental investigation requested?
ECG requested?	Haematology test requested?
MRI scan requested?	Immunology test requested?
Continuous vitals requested?	No tests requested
Orthopaedic tests requested?	Other tests requested?
Pregnancy test requested?	Respiratory rate at entry
Serology test requested?	Previous specialty
Toxicology test requested?	Ultrasound test requested?
Urine test requested?	X-ray scan requested?
Admission method	Admission source
Age	Experiencing atrial fibrillation?
# Historic diagnoses	Previous management
Previous admission to ED?	Ethnic category
Frequently admitted?	Gender
Dist. of address to hospital	No. Investigations requested
Previous ED visit days ago	Previous visit LOS
Mortality indicator severity score	Historic diagnosis codes

Table 3.2: Table containing the environmental/hospital features that were used in all of the models.

Hours of sunlight	Hour admitted to ED
Month admitted to ED	# ED Attendees in last 12 hours
# ED Attendees in last 4 hours	# ED Attendees in last 8 hours
# ED Attendees in last hour	# Breaches last 12 hours
# Breaches last 4 hours	# Breaches last 8 hours
# Breaches last hour	Has it rained today?
Min. day temp (degrees)	Max. day temp (degrees)
Weekday (one-hot)	

3.7 Other Datasets

In order to validate the methods that we propose, it is important to apply the methods developed to multiple datasets. We choose to do this on a publicly available medical dataset, MIMIC-III [107] as well as two image recognition datasets, CIFAR-10 [44] and MNIST[51], due to their popularity in the field of machine learning.

MIMIC-III

The MIMIC-III dataset is a freely accessible critical care database developed by the MIT lab for computational physiology. It contains the demographics and vitals measurements of approximately 60,000 intensive care unit admissions. Predicting the ward allocation problem is not possible with this dataset as all of the patients are ICU admissions. As a result, we aim to validate our methods on the binary classification of survival prediction upon entry to hospital. To do this, the patients for this dataset are also filtered according to emergency admissions only, and only those with a full feature set (based on 11 features that are available early in the patient's admission) are considered. This provides a dataset of 8806 patients. As MIMIC-III is an ICU focused dataset, replicating the experiment we have carried out with the OUH dataset is not possible. All features used are shown in Table 3.3.

Table 3.3: Table containing the features used for mortality prediction on the MIMIC-III dataset. Features are defined in MIMIC-III documentation.

Admission type	Admission location
Insurance	Language
Religion	Marital Status
Ethnicity	Has previous chart events
Previous ICD9 code	First care unit
First ward ID	

CIFAR-10

To further validate our methodology, we also report results on the CIFAR-10 image recognition dataset [44] to assess the model's performance on a non-tabular dataset. CIFAR-10 is a dataset of 60,000 32x32 colour images representing ten different classes. They are evenly balanced, meaning that each class has 6000 images associated with it. The ten classes are the following: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck.

MNIST

MNIST is a dataset of images of handwritten digits. It is composed of 70,000 32x32 images. The labels correspond to the value of the number written in the image. There are ten labels corresponding to the values of 0 to 9. The images are approximately evenly balanced, with the number 1 being slightly more prevalent than the others at 11% of the dataset. The least prevalent is the number 5, at 9.04% of the dataset.

eICU

The eICU dataset is another database hosted on Physionet [19] along with MIMIC-III. It is an ICU focused dataset containing patient data from various ICU admissions spread across 208 unique hospitals in the US. This dataset has been introduced due to its use in validating the methodology described in Chapter 6. We follow the example of [214] in attempting mortality prediction using the features found in Table 3.4. There are 73,718 patient records stored in the database with 6,167 dying during their stay. Any patients with missing features were removed from the dataset. Summary statistics on this dataset can be found in [214].

Table 3.4: Table containing the features used for mortality prediction on the eICU dataset. Features are extracted from the first measurement of each patient upon admission to ICU.

Heart rate	Mean arterial pressure
Diastolic blood pressure	Systolic blood pressure
Blood oxygen saturation	Respiratory rate
Temperature	Glucose
Oxygen delivery rate	pH
Height	Weight
Age	Admission diagnosis
Ethnicity	Gender
Glasgow coma score total	

3.8 Summary

In this chapter we investigated the data available in the IORD dataset. We showed the distributions of commonly measured features in the ED and discussed how these could be used for prediction. We displayed attendances to the ED with time, where in the hospital patients are admitted, how patients transfer between wards and lengths-of-stay within the OUH NHS Trust. In chapter 2 we discussed how prediction of arrivals to the ED is very difficult due to there being very little data on which to base the predictions. Autoregressive methods based on the admission numbers have also shown limited utility. Figure 3.21 shows that the transitions within the hospital are sparse, meaning there is limited utility to clinical staff. Figure 3.17 shows that predicting LOS is useful provided that it is actionable, and in many cases this is not true. As a result, we chose to focus constructing a dataset to tackle the problem of predicting the location of admission of a patient in the hospital.

We showed how we used the IORD dataset to construct the dataset that would be required to predict the location of admission. The wards are grouped into ward-types (which was done with the help of our clinical collaborators) rendering this a seven-class classification.

Due to the imbalance of admissions to certain wards, the problem can be posed as a multi-stage prediction (predicting admission to EAU and SEU before ward-type) or

we can simply predict the ward-type to which patients will be admitted. This could be done while removing admissions to the EAU and SEU. This would acknowledge that the SEU and EAU have multiple functions for which prediction is not useful.

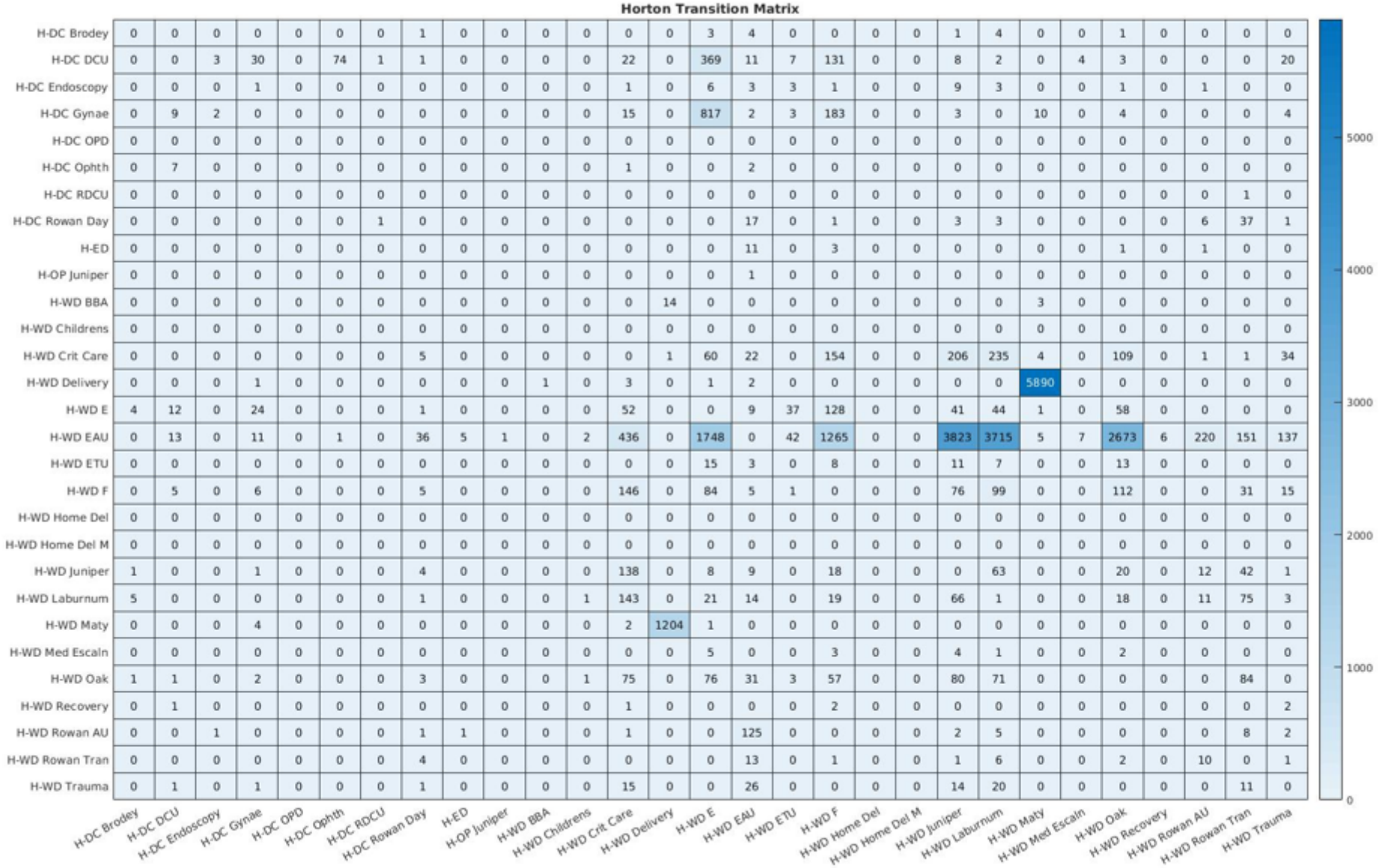


Figure 3.21: The ward transition matrix for the Horton hospital.

Chapter 4

Predicting Admission Location Using the Mahalanobis Curriculum

Chapter Abstract

This chapter presents a deep learning method of predicting where in a hospital emergency patients will be admitted after being triaged in the ED. Such a prediction will allow for the preparation of bed space in the hospital for timely care and admission of the patient as well as allocation of resource to the relevant departments, including during periods of increased demand arising from seasonal peaks in infections. The problem is posed as a multi-class classification into seven separate ward types. A novel deep learning training strategy is created that combines learning via curriculum and a multi-armed bandit to exploit this curriculum post-initial training. We successfully predict the initial hospital admission location with area-under-receiver-operating-curve (AUROC) ranging between 0.60 to 0.78 for the individual wards and an overall maximum accuracy of 52% where chance corresponds to 14% for this seven-class setting. Our proposed network was able to interpret which features drove the predictions using a 'network saliency' term added to the network loss function. We have proven that prediction of location of admission in hospital for emergency patients is possible using information from triage in ED. We have also shown that there are certain signifying tests which indicate what space of the hospital a patient will use. It is hoped that

this predictor will be of value to healthcare institutions by allowing for the planning of resource and bed space ahead of the need for it. This in turn should speed up the provision of care for the patient and allow flow of patients out of the ED, thereby improving patient flow and the quality of care for the remaining patients within the ED.

4.1 Introduction

Deep neural networks (DNNs) have revolutionised the field of machine learning by providing a way to utilise very large datasets as well as large feature spaces to make meaningful predictions. State of the art performance has been achieved by DNNs in a wide range of tasks proving their efficacy as learning algorithms. Their strength in function approximation has not been overlooked by the medical community, with numerous publications exploiting them to make useful predictions for various healthcare scenarios [63], [82], [127].

One of the challenges of utilising DNNs is that they are non-convex optimisation problems, meaning that the best performance that the algorithm is capable of may not be achieved [34]. As a result, much work has been carried out in developing methods of presenting data to the network for training in a structured fashion [43]. This has since been called a curriculum and is widely used when training DNNs today [43].

The aim of this work is to utilise the concept of curriculum training to train a model that will predict where in a hospital a patient will be admitted based on very early information obtained in the ED from the triage nurse. We aim to show that the movement of patients from ED to one of seven different ward types in hospital is predictable. This would allow allocation of a bed and resources for the patient well ahead of admission to ensure that they receive care and treatment in as timely a fashion as possible. We also aim to demonstrate that this prediction can be made given data collected from a patient at point of entry to the ED, which in turn will improve

the flow of patients out of the ED and into the hospital. Difficulties in admitting patients to the optimal hospital ward are often most marked during periods of high demand, such as during peaks in seasonal infections including influenza. We therefore test the performance of our model throughout the year.

4.2 Related Work

Making Models Accurate

In order to provide as accurate a prediction as early as possible in the admission process, we must investigate methods that maximise classification performance. The training set is a tabular dataset composed of a mixture of categorical and continuous variables (see Chapter 3). Current state-of-the-art methods on tabular data include random forests [34] and the XGBoost algorithm [104]. Neural networks have also been shown to be highly effective in classification [84].

One of the recent developments in the training of neural networks is the introduction of 'curriculum learning', a method by which the accuracy of the trained model can be improved. Curriculum learning stems from the observation that children in schools learn by beginning with simple ideas and progressing on to more complex topics. By doing so they are able to understand fundamental principles on which they can build to learn more complex topics (which in themselves are usually simply superpositions of the fundamental principles). Curriculum learning proposes that neural networks may also benefit from this structured approach to learning. By presenting the network initially data that are 'easier' to optimise over, the optimisation surface (of network prediction error versus network parameters) is more likely to be convex [43]. This has an analogy with numerical continuation methods, where a complex optimisation surface is decomposed into layers, beginning as a completely convex surface and gradually increasing in non-convexity [61]. In this thesis we will exploit this methodology in order to train neural networks on noisy medical data. We will then compare this to

normal batch methods of training networks and see the effect that the curriculum has on the prediction accuracy.

It has been shown in [140] and [98], amongst others that training neural networks in a structured fashion by presenting batches from easiest to hardest leads to improvement in performance of the network, as well as increasing the speed of training. However, we also see in the works of [168] and [43] that training in an 'anti-curriculum' (hardest to easiest data) can also lead to improved performance. A curriculum may also be defined for specific problem sets, with results from [85] and [88] showing that a curriculum tailored for a specific task provided better results for that task than a general curriculum. This is indicative that training according to some type of structure is important for learning, and that structure may be task specific.

The use of non-stationary bandits in learning has also been explored in [128] where a curriculum is arranged and a bandit selects which batches to train a neural network on. The bandit is trained by measuring how a particular batch of data improves the performance of the network which in turn affects the probability of selecting that curriculum batch to train on. The better the performance, the more likely the bandit is to choose this batch of data again. The authors of [128] propose four different algorithms to select the next curriculum batch to train on. These are: the use of a non-stationary bandit to select the next batch to train on; using linear regression and a windowed linear regression on the performance of the network to predict the batch most likely to provide the best performance after training; using Thompson sampling to select the next batch for training. The authors found that the non-stationary bandit was the most effective method of choosing the next batch of data providing the best performance and faster training. While these approaches have an effective performance on the training problems presented in the work, the authors do not utilise the curriculum to guide their network weight space into the domain of a global minimum. Another work which uses a similar approach is that of [117], where a curriculum is also generated and a non-stationary bandit is used with the EXP3.S

algorithm [23] to select the next curriculum batch to train on. However, once again, without using the curriculum initially, this algorithm will not always provide a better or faster training of the network.

Interpreting Models

Aside from simply improving the accuracy of a model it is important, particularly when using deep models in the healthcare domain, to provide a level of interpretability to the decision-making process. In [133], the authors emphasise the importance of understanding what in the input space has driven a decision in order to learn from the model, or to validate the classification. We again see this in a review of deep learning in healthcare by [131], where one of the fundamental challenges noted is interpretability of deep learning models and relating the decision made back to the input space.

Various methods have been proposed for interpretability. Traditionally, applying the LASSO loss to linear models has been used a form of interpretability as to feature importance for prediction [34]. However, for non-linear models LASSO is no longer applicable as an "interpretable loss". Classification trees have strong interpretability properties due to their formulation [34]. However, without information-gain-based learning, this form of interpreting predictions is difficult to implement. There are also Bayesian methods for identifying feature contribution to predictive outputs such as through the use of automatic relevance detection (ARD) where a distribution is defined for each feature as a prior [65]. The current state-of-the-art for interpreting neural network predictions is through the use of Shapley values [4]. We therefore propose that a form of interpretability be used in our model so that clinicians can make use of this information and prioritise data collection in the ED for implementation of the algorithm.

4.3 Novelty

The novelties of this chapter are as follows: we have developed a novel strategy for the training of neural networks combining a curriculum training phase (presenting data for training according to some sequence) with a multi-armed bandit phase (a non-stationary reward maximising algorithm as explained in the methodology) to maximise prediction performance on noisy biomedical data. This also incorporates a saliency layer before the inputs which allows interpretation of the importance of the input features. To the best of our knowledge no other work has proposed the framework of predicting where in the hospital a patient from the ED will be admitted. This is also believed to be the first work to employ deep learning architectures in order to carry out hospital admission prediction.

4.4 Methodology

4.4.1 Curriculum Learning

Due to the non-convex nature of optimising artificial neural networks (ANNs), a structured method of presenting data to the network via curriculum learning was introduced with the aim of reducing the likelihood of the weights being optimised into a local minimum [43]. There are similarities between curriculum learning and numerical continuation methods as pointed out in [43], where optimisation of a complex surface is achieved through first optimising over smoother more convex versions of the surface. Consider a family of cost functions $\mathcal{C}_\lambda(\theta)$ such that $\mathcal{C}_0$ is easy to optimise over (and which is likely to be more convex than other functions), $\lambda \in [0, 1]$ is the ranking of "difficulty to optimise" and where $\mathcal{C}_1$ is the actual cost function that is to be minimised. By optimising over the network parameters, θ, for $\mathcal{C}_0$, as $\mathcal{C}_0$ is simply a smoother version of $\mathcal{C}_1$ we bring our parameters into the domain of a minimum of $\mathcal{C}_0$ as well as $\mathcal{C}_1$. We then gradually increase λ while keeping θ at the local minimum. This helps to

avoid local minima which may be present in the more complex optimisation space. The aim, therefore, is to create batches of data, $B \in \mathcal{B}$ (the set of all batches), ranked according to λ (i.e., B_λ with $\lambda = 0$ being the "easiest" batch of data to optimise progressing to the "hardest" as λ increases.) These batches are then presented to the network for training in order of increasing λ. Note that the batch $B_{\lambda+\kappa}$ will contain all of the data in B_λ for $\kappa > 0$, as an increment in λ represents the addition of more "complex" data to the previous batch.

With application to real data, we need to define "easiness" of fitting to the data. We define a sequence of batches of data $B_\lambda(z)$ comprised of individual data entries, z, such that $\int B_\lambda(z)dz = 1$ (i.e., our whole dataset). Note that z is a tuple composed of input features as well as corresponding label as the most generalised form of curriculum may incorporate class label as a measure of complexity. We also define $B_\lambda(z) \propto W_\lambda(z)U(z) \;\; \forall z$, where $W_\lambda(z)$ is the weight assigned to example z at the point λ in the curriculum sequence and $U(z)$ is the sampled training data ($W_\lambda(z)$ is 0 for data at low values of λ i.e, excluded in these batches). The "easiness" of the fit to data is described by:

$$H[B_\lambda(z)] < H[B_{\lambda+\epsilon}(z)] \qquad \forall\ \epsilon > 0 \tag{4.1}$$

where H is the entropy of data batch B. The weights of the examples also increase with λ as:

$$W_{\lambda+\epsilon}(z) \geq W_\lambda(z) \qquad \forall\ z,\ \forall\ \epsilon > 0 \tag{4.2}$$

to balance training as the "less complex" data will have been presented to the network for training a greater number of times. This is because the first curriculum batch ('easiest') will also be a part of all the other curriculum batches, i.e, for N curriculum batches denoted by B, $B_0 \subset B_1 \subset \ldots B_N$. Therefore the data in B_0 is presented to the network a greater number of times and so the rest of the data must be weighted

to account for this, so that all data is presented an equal number of times.

In this work we define "complexity" of the data using the Mahalanobis distance in order to encode the notion of entropy. The Mahalanobis distance is a multi-dimensional generalisation of measuring the number of standard deviations that exist between a point and the mean, μ, of a probability density function (p.d.f), D [18]. The larger the Mahalanobis distance the more unlikely the data entry is to belong to the (assumed unimodal) distribution (and which is therefore of higher entropy). We therefore assume that our data belong to a single p.d.f, with mean μ and covariance $\mathbf{S}$. Due to the input features being of mixed data types, we encode our input features through a trained denoising autoencoder to gain a representation of the data in an embedded space before calculating the Mahalanobis distance (see next section). In using the Mahalanobis distance, our curriculum organises our training data such that we train according to the most similar samples first (the smaller number of samples of different classes in this batch increases the likeliness of finding a more global minimum, therefore making it "easier" to optimise over) before progressing on to the easier to differentiate between samples. This mirrors the approach that is used in the SVM in defining the separation boundary where data of differing classes are closest together.

Algorithm 1 Curriculum learning with multi-armed bandit learning

procedure INITIALISATION
    Num. iterations = N_a
    Num. of curriculum batches = $N_{choices}$
    Calculate Mahalanobis of each data entry = D_m
    Sort D_m from lowest to highest = $\hat{D_m}$
    Divide instances into $N_{choices}$ batches = $\mathcal{B}$ s.t. $\mathcal{B}_0 \subset \mathcal{B}_1 \subset \ldots \mathcal{B}_{N_{choices}}$
    loop:
    for i in N_a **do**:
        for j in $N_{choices}$ **do**:
            Train network on batch $\mathcal{B}_j$
    Launch bandit as defined in Algorithm 2

4.4.2 Encoding Data of Mixed Types

The first task for our model is to organise data into batches according to their complexity. As previously mentioned, this can be achieved according to the entropy of the data (among other methods not described here for brevity). For tabular data (categorical and numerical data such as the information recorded at triage), we choose to represent the entropy through the use of the Mahalanobis distance. The Mahalanobis distance is defined as:

$$d\left(\mathbf{x}_n\right) = \left(\left(\mathbf{x_n} - \boldsymbol{\mu}\right)^T \mathbf{S^{-1}} \left(\mathbf{x_n} - \boldsymbol{\mu}\right)\right)^{\frac{1}{2}} \tag{4.3}$$

where $\mathbf{x_n} \in z_n$ are the input features of datapoint n, $\boldsymbol{\mu}$ is the vector of the mean value of each feature, and $\mathbf{S}$ is the covariance matrix. As certain inputs to a neural network may be of different data types, we encode the input data prior to organising into a curriculum using a denoising autoencoder for the Mahalanobis distance to be defined in a scalar space. This is so that all of our features will be represented in a continuous space. This overcomes the issue of using a continuous metric such as Mahalanobis distance with features that are binomial and multinomial distributed. Using the original feature space and standardising the features was compared against but the curriculum (and baseline methods) all performed better with the encoded representation consistently. As a result, this approach of encoding was used throughout the thesis.

Let $\mathcal{Z}$ be a training dataset consisting of inputs and labels $\mathbf{x_n}$ and y_n respectively. We first train a denoising autoencoder such that $\phi\left(\mathbf{x_n}\right) \approx \mathbf{x_n}$. To construct our curriculum from this latent representation, we calculate the Mahalanobis distance, $d\left(\phi\left(\mathbf{x_n}\right)\right)$ for all n. We then sort the data from the lowest Mahalanobis distance to the highest and create N batches by dividing the whole set into N separate batches. Two methods of creating the batches are investigated:

1. All batches are separate and contain separate training data; i.e, if B_0 is the batch

with the lowest $d\left(\phi\left(\mathbf{x_n}\right)\right)$ and B_N that with the highest, then $B_0 \cap B_1 \cap \ldots B_N = \emptyset$

2. Batches are cumulative supersets; i.e, $B_0 \subset B_1 \subset \ldots B_N$.

4.4.3 Regularisation Using a Mahalanobis Curriculum

This section is an investigation as to how the way in which data is introduced to the network may naturally induce regularisation during training. We postulate how the Mahalanobis curriculum may naturally regularise itself. As before, $\mathcal{Z}$ is a training data set consisting of datapoints z_n where $z_n \in \mathcal{Z}$ and z_n consists of input features and a label such that $z_n = \{x_n, y_n\}$.

Using equation 4.3 we can now create a vector, D_m, of distance of each datapoint from the mean of the assumed probability density function of the dataset, where $\mathcal{X} \rightarrow D_m, \ \ \forall \ \ x_n \in \mathcal{X}, \ \ \mathcal{X} \subset \mathcal{Z}, \ \ \mathcal{Z} \subset \mathbb{R}$.

We now seek to create N batches of training data of increasing entropy of size k datapoints where $k = \frac{\mathbf{card}(D_m)}{N}$.

We then extract the indices of the lowest entropy features using the following formulation:

$$j_N = index\left(\bigcup_{i=1}^{Mk} \min\left(\left(\ldots\left(d_m \setminus d_{m_1}\right) \setminus d_{m_2}\right) \ldots \setminus d_{m_i}\right)\right) \tag{4.4}$$

for $M = \{1, 2, ..., N\}$, and d_{m_b} is the b_{th} smallest element of the set D_m. We are then able to construct the N curriculum batches $B_N = \{\mathcal{X}\{j_N\}, \mathcal{Y}\{j_N\}\}$. The training proceeds by presenting the batches in B for the smallest N first and then gradually increasing N.

Consider a typical cost function used for backpropagation, the mean-squared error: $\frac{1}{k}\sum_{n=1}^{k}\left(\hat{y_n} - y_n\right)^2$. For a linear system this can be re-written as $\frac{1}{k}\sum_{n=1}^{k}\left(\boldsymbol{w}\boldsymbol{x_n} - y_n\right)^2$ where $\boldsymbol{w}$ is a vector of weights of the linear model. Note that here we are considering the case of a linear model. Due to the difficulties associated with the non-linearities induced by the activations of the nodes of a neural network, we demonstrate empirically

in the results section that this analogy holds.

Using the definition of the Mahalanobis distance as shown in Equation 4.3, if we consider $\boldsymbol{x_n}$ to be a random variable, we see for normally distributed data $\boldsymbol{x_n} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{S})$ and $\boldsymbol{x_n} \rightarrow \boldsymbol{\mu} + \sqrt{\boldsymbol{S}} d_m$. For ease of notation, we assume that all input features are orthonormal, i.e, $\boldsymbol{S}$ is a diagonal matrix. Therefore we see that $\boldsymbol{x_n} = (d_m^2.(\boldsymbol{SI}))^{\frac{1}{2}} + \boldsymbol{\mu}$, where $\boldsymbol{I}$ is the identity matrix and which we substitute back into our expression for MSE, which expands to the following expression:

$$MSE = \frac{1}{k}\sum_{n=1}^{k}(\boldsymbol{w}^T\boldsymbol{w}\left(d_{m_n}^2(\mathbf{SI})\right)+2\boldsymbol{w}^T\boldsymbol{w}\left(d_{m_n}^2(\mathbf{SI})\right)^{\frac{1}{2}}\boldsymbol{\mu}-2\boldsymbol{w}\left(d_{m_n}^2(\mathbf{SI})\right)^{\frac{1}{2}}y_n+(\boldsymbol{w\mu})^2-2\boldsymbol{w\mu}y_n+y_n^2) \tag{4.5}$$

When we are training with a curriculum, we train initially with low entropy data so that $d_{m_n} \rightarrow 0$: $MSE \rightarrow \frac{1}{k}\sum_{n=1}^{k}\left(\boldsymbol{w}^T\boldsymbol{w\mu}^T\boldsymbol{\mu} - 2\boldsymbol{w\mu}y_n + y_n^2\right) = \frac{1}{k}\sum_{n=1}^{k}\left(\boldsymbol{w\mu} - y_n\right)^2$. For very low entropy values we are simply calculating the mean squared error with respect to the mean of our assumed p.d.f. and so we bias our model towards a model fit to the mean.

Now we investigate as d_m becomes large: We assume that the first 3 terms in the expanded MSE equation will dominate the response due to the large value of d_m:

$$MSE \rightarrow \frac{2}{k}\sum_{n=1}^{k}\left[\boldsymbol{w}\left(d_{m_n}^2.(\mathbf{SI})\right)^{\frac{1}{2}}\left(\frac{\boldsymbol{w}}{2}\left(d_{m_n}^2.(\mathbf{SI})\right)^{\frac{1}{2}}+[\boldsymbol{w\mu}-y_n]\right)\right]$$

Here there are two important things to notice: first, the cost function now contains an additive loss term proportional to $||\boldsymbol{w}||$. This means that in the case of overfitting where the magnitude of the weights increases dramatically, the error function will be penalised for this. This is artificially introduced using L1/L2 regularisation whereas here it naturally arises with data that is perceived to be of higher entropy. Second, the difference between prediction and label is no longer squared meaning we have much more gradual learning with higher entropy data (which is positive as we don't want to learn the noise that is associated with these data).

By using a curriculum we initialise our function approximation using the mean

of the data. This is advantageous as it greatly reduces the likelihood of our function approximation being skewed by outliers and possibly even erroneous data. The curriculum can be seen as a form of pre-training which allows the model to find a good local minimum for the problem at hand from which to further fine-tune its performance with further training.

4.4.4 Multi-Modal Curricula

The Mahalanobis curriculum makes an implicit assumption that the dataset is unimodal, whereas in the vast majority of cases in real-world data this is not the case [65]. In this section we investigate the advantages and disadvantages of using a multi-modal curriculum as opposed to the unimodal. We experiment with various clustering algorithms to find our modes and induce Mahalanobis curricula over these in order to compare with the unimodal setup, and examine the efficacy of each.

In order to find our modes we propose the use of a Gaussian Mixture Model (GMM), as well as the k-means clustering algorithm. These allow the formation of clusters in the feature space which we can treat as modes of the dataset. We will also experiment with more heuristic clustering schemes such as clustering data points according to label. Such a scheme is reliant on high-fidelity in the labelling, something that can often be lacking in medical datasets.

Gaussian Mixture Model

A Gaussian Mixture Model (GMM) is a model that assumes all points are derived from a mixture of Gaussian distributions with unknown parameters [34]. This is particularly useful for multi-modal datasets where a single Gaussian will not represent the multiple modes adequately and so using a combination is necessary. The number of Gaussians is a user-specified parameter and these are then fitted using the expectation maximisation (EM) algorithm.

The EM algorithm is an iterative training procedure where one first computes the

"expectation" of the data points according to the current cluster assignments and one then "maximises" the likelihood of the data given the cluster assignments by adapting the cluster parameters (in the case of the Gaussian these correspond to the cluster centres and covariance matrices). Typical practice when using GMMs is to initialise the centroids of the Gaussians using the k-means algorithm.

More formally, consider a model with parameters θ that are to be fit to describe observed data X. Our likelihood function would be $L(\theta; X) = P(X \mid \theta)$. Now consider that there remain some unseen data or latents, Z. Our likelihood function would become $L(\theta; X, Z) = P(X, Z \mid \theta)$. Typically, we would seek to marginalise out the latent variable by integrating with respect to Z, however Z is often intractable. As a result we implement the following iterative procedure:

- E-step: Define $q\left(\theta \mid \theta^{(t)}\right)$ to be the expected value of the log-likelihood of θ with respect to the distribution over Z given the current parameter estimates and observed data so that $q\left(\theta \mid \theta^{(t)}\right) = \mathbb{E}_{Z \mid X, \theta^{(t)}}\left[\log L(\theta; X, Z)\right]$
- M-step: Find the parameters that maximise the likelihood q: $\theta^{t+1} = \arg\max_{\theta} q\left(\theta \mid \theta^{(t)}\right)$
- Repeat until convergence

K-means Clustering

The k-means algorithm is another clustering algorithm that can be used to identify clusters of modes in datasets. One can think of k-means as a more specific form of a GMM where only the cluster centroids are considered and not the covariance matrices. Similarly to the GMM, training a k-means model also follows an iterative procedure. These are termed the assignment step and the update step. To begin with, the number of clusters must again be specified by the user and these can be randomly initialised or selected in another way.

In the assignment step we assign a data point to the cluster with the nearest mean. As this involves a concept of distance, various metrics are used however most

implementations use Euclidean distance. This can be described as:

$$S_i^{(t)} = \{x_n : \left\|x_n - m_i^{(t)}\right\|^2 \leq \{x_n : \left\|x_n - m_j^{(t)}\right\|^2 \; \forall j, 1 \leq j \leq k\} \tag{4.6}$$

where S is a cluster at iteration t, x_n is the data point to be clustered, m is the cluster centroid and k is the total number of clusters.

In the update step we recalculate the centroids assigned to each cluster as:

$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j \tag{4.7}$$

meaning that we update each cluster centroid as the mean of the data points assigned to that cluster due to the assignment step. The procedure then repeats until convergence as in the case of GMMs.

4.4.5 Comparison of Unimodal and Multimodal Curricula

As most real-world datasets are multimodal, we choose to implement both the unimodal curriculum as well as the multi-modal and investigate the performance differences (if any) and any insights we can gain from training in these different ways. The aim of carrying out this exercise is to help identify if there are certain parts of medical datasets that are more useful for training than others. This would allow for targeted collection of data in hospitals without compromising on classification model performance. To compare we propose:

- Sampling the dataset according to Equation 4.4 on the whole training dataset (unimodal assumption) to generate N batches and training in forward and reverse sequences
- Using a GMM to cluster our training dataset into s clusters. For each cluster we then use Equation 4.4 to create N batches so that we have a total of sN

batches. We then concatenate batch $b \in N$ from each cluster so that we again have N batches of lowest entropy from each cluster in the first batch going to the highest from each cluster in the last batch.

- Repeating the above, but training with the separate clusters independently in different orders as opposed to concatenating the batches.
- Repeating the analysis, but creating clusters using k-means
- Repeating the analysis, but creating clusters according to class assignment

We further experiment with batches being cumulative supersets of each other, different entropy metrics than the Mahalanobis distance and mixing the order of presentation of the batches as shown in Section 4.6.

4.4.6 Multi-Armed bandits

The curriculum is trained in a cyclical fashion which, as described previously, is hypothesised to be beneficial for finding a local minimum near the global minimum. However, after initial training there is no reason why this cyclical training should provide the best possible performance of the model. Given that we now have discrete batches of data created by the curriculum, we introduce a multi-armed bandit in order to choose the best batch to train the network on.

A multi-armed bandit is a method in which choices need to be made based on allocation of a finite resource, where the aim is to maximise the expected reward of allocation of the resource [54]. The probabilities of reward based on choice are only partially known at the time of allocation and the optimal choice to maximise reward becomes clearer as resource is spent. The multi-armed bandit is an example of an exploration versus exploitation problem as is often framed within reinforcement learning problems. A hyperparameter that is manually chosen, ϵ, defines the rate with which exploration of the choices occurs (by choosing a batch at random) as opposed

to exploiting the batch with the highest reward. Due to the non-convex nature of training deep neural networks, we can view the training of a deep neural network as a multi-armed bandit problem. For multi-class classification, certain classes are learned more rapidly depending on the data that has been presented to the network to train it. By using the concept of batches of data split according to their "easiness" as introduced by curriculum learning, we can treat this as a problem of choosing the right data to train the network on in order to maximise our reward, which in this case is the general accuracy of the model in a multi-class classification.

Algorithm 2 The multi-armed bandit for training of the network after initially trained with a curriculum

procedure INITIALISATION
    rate of exploration = ϵ
    resource available = N_a
    Prob. curriculum batch gives max reward = P
    Num. training data batches = $N_{choices}$
    Count of number of times batch is chosen = K
    Batches by Mahalanobis distance = $\mathcal{B}$
    Accuracy of class i on training set = δ_i
    loop:
    for i in N_a **do**:
        if $\epsilon > u \sim U(0,1)$ **then**
            batch = $\mathcal{B}[int\,(u \sim U(0, N_{choices}))]$
        else
            batch = $\mathcal{B}[\arg\max(P)]$
        Train on batch and find accuracy on training set
        if t_0 **then**
            $t_0 \rightarrow A_T^0 = \sum_1^C (\delta_c)$
        else
            $t_{1:N_a} \rightarrow A_T^t = \sum_1^C \left((\delta_c^t - \delta_c^{t-1})\,/\delta_c^{t-1}\right)$
        Test on validation set
        A_v^t = overall accuracy on validation set
        reward = $A_T^t \times A_v^t$
        $K[\text{batch}] = K[\text{batch}] + 1$
        $\alpha = 1/K[\text{batch}]$
        $P[\text{batch}] = P[\text{batch}] + \alpha \times (\text{reward} - P[\text{batch}])$

Algorithm 2 shows how the multi-armed bandit problem was applied for training.

We begin by defining the exploration rate, ϵ, how many batches of data we have, $N_{choices}$, and how many attempts we have at training the network with the batches, N_a. We also initialise vectors of zeros of the same length as the number of training data batches, K and P.

For a value of $\epsilon = 0.1$, the bandit would explore (choose a different training data batch at random) 10% of the time. Otherwise, the bandit will choose the training data batch that has the greatest probability of returning maximal reward.

Once the training data batch has been chosen, we train using these data. The reward is then calculated. For multi-class classification, we require a reward function that will improve the accuracy of prediction over all classes and not just the classes that are more prevalent in the data. We therefore define our reward function with respect to the learning rate of all the classes as well as the performance on the validation set to ensure that the model does not overfit.

$$R = \sum_{i=1}^{C} \frac{\delta_c^t - \delta_c^{t-1}}{\delta_c^{t-1}} \times A_v^n \tag{4.8}$$

where A_v^n is the validation set accuracy of the current training episode, δ is the accuracy of class c over the training set and t is the current training episode. By incorporating A_v^t, as soon as the model begins to overfit on the training data, reward due to the first term in Equation 4.8 will increase; however, any detriment to the general performance will be reflected by A_v^t which will prevent the reward increasing (i.e, a decrease in the accuracy over the validation set would lead to the sum of the learning gradients being multiplied by a small number, thereby reducing the reward).

4.4.7 Baseline Methods

In order to verify that our methods are indeed useful for the training of neural networks, we initially conduct our analysis using several established baseline methods. These are multiple logistic regression (LR), a support vector machine classifier (SVM), a

random forest (RF) and the XGBoost algorithm (XGB).

Multiple Logistic Regression

Multiple logistic regression is based on the assumptions that the system to be represented is linear, and that this linear system can be described parametrically using coefficients that are scalar values. The aim therefore of the regression is to optimise the coefficients so that they most accurately transform the recorded data values to the desired output value [68]. For binary classification problems, the regression is then used in the logistic function which outputs the prediction and the weights are obtained by minimising the negative log-likelihood between the label and the predicted value.

Polynomial regression is extended by allowing the features to be multiplied together up to the limit of the degree of the polynomial that is set. This allows a greater degree of freedom in terms of fitting data, however, it is also prone to overfitting. As a result, we will need to use regularisation which can involve adding a penalty term to the cost function to avoid overfitting the model to the data [68].

where the additional term penalises model complexity, thus constraining the fitting process:

$$r = \sum_{1}^{n} \left(y_n - \boldsymbol{w} \cdot \boldsymbol{x}_n \right)^2 + \beta \sum_{j=1}^{f_d} |w_j| \tag{4.9}$$

where y_n is the target to be predicted for data item n, f_d the total number of features, $\boldsymbol{w} \cdot \boldsymbol{x}_n$ is the model's unregularised prediction of the output and where β is a penalty factor which determines the extent to which we penalise coefficients with large absolute values (which cause overfitting) [65]. This is then used in the logistic function as:

$$\hat{y} = \frac{1}{1 + e^{-r}} \tag{4.10}$$

These are linear models in the weights of the inputs and in order to understand if this is the most appropriate approach, we must also assess non-linear approaches too.

Support Vector Machines

To address non-linear models, a support vector machine (SVM) classifier was used for the first two stages of the hierarchical model. The SVM is a non-probabilistic classifier that defines a hyperplane that is as wide as possible between the separate classes (i.e., it separates the classes with maximum margin). The SVM can also perform non-linear classification by mapping the input data into a higher-dimensional space in which the data become linearly separable via the "kernel trick" [14].

Given a dataset of the form $(\boldsymbol{x_1}, y_1)$ ... $(\boldsymbol{x_n}, y_n)$, where $\boldsymbol{x_m}$ are vectors of input data and y_m is the corresponding class $\{-1, 1\}$, we specify a hyperplane which separates the classes as $\boldsymbol{w} \cdot \boldsymbol{x} - b = 0$ where $\boldsymbol{w}$ is normal to the hyperplane. The data points that are closest to this hyperplane from each class are called *support vectors*. We define two parallel hyperplanes that run through the support vectors. These are described by the equations $\boldsymbol{w} \cdot \boldsymbol{x} - b = 1$ and $\boldsymbol{w} \cdot \boldsymbol{x} - b = -1$. The distance between the two class hyperplanes is given by $\frac{2}{|\boldsymbol{w}|}$ and the best classification is achieved by minimising $|\boldsymbol{w}|$ so that we maximise the gap between the two classes.

Random Forest

A random forest is an ensemble method that uses a collection of decision trees to classify. Decision trees are usually constructed in a top-down manner by creating 'splits' at certain variables where the data can be best separated [34]. To carry out this separation some measurement of entropy of the variable must be calculated. Various methods are used including calculating the entropy explicitly or calculating the Gini impurity [65]. The resulting tree is an interpretable decision-making sequence that allows us to separate the data according to the features that make the data most separable at each step.

Random forests use bootstrap (sampling from a dataset with replacement) aggregating (taking the average output of the ensemble models or the majority vote for classification), which is also known as "bagging", over an ensemble of decision trees.

As decision trees are prone to overfitting with increasing depth [34], using bagging helps to increase the bias and reduce the variance of the outcome. By also increasing the number of models used in the ensemble, we again can reduce the variance in the output.

XGBoost

XGBoost or 'eXtreme Gradient Boosting' is a form of gradient-boosted tree that has enjoyed great success when applied to tabular datasets. It is built on a foundation of gradient-boosted trees, a boosting method whereby a series of sequential trees ('weak learners') are iteratively trained in order to create a final strong learner. Consider a model, F, that is to predict some output y. For every stage of boosting, we would get $F_{m+1}(\boldsymbol{x}) = F_m(\boldsymbol{x}) + h(\boldsymbol{x}) = y$ where $h(\boldsymbol{x})$ can be written as $h(\boldsymbol{x}) = y - F_m(\boldsymbol{x})$. Therefore, $h(\boldsymbol{x})$ is a model that fits the residual between the label and the current model estimate, $\hat{y}$. We then iterate using this and $F_{m+1}(\boldsymbol{x}) = F_m(\boldsymbol{x}) + h(\boldsymbol{x})$. Note that using a mean-squared error loss function over the residual would be of the form $\frac{1}{2}\left(y - F_m(\boldsymbol{x})\right)^2$ of which the negative gradient with respect to F would be $y - F_m(\boldsymbol{x})$, simply the residual. As a result, various losses and their associated gradients can be used for this step.

Feedforward Neural Networks

For the final stage of the hierarchical model a fully-connected feedforward neural network was used. The feedforward network is a class of artificial neural network that is characterised by nodes and layers. The nodes represent operations on data from the previous layer, and the hidden layers are defined as those between the input and output layers. All nodes are fully feedforward connected, meaning that the output of a node in layer l will be fed into every node in layer $l + 1$.

Data are presented to the network at the input layer and are typically normalised in the range $[-1, 1]$. Each connection leaving the input layer corresponds to a synaptic

weight. As this is a fully-connected network, between layer l and $l+1$, there will be $N_l N_{l+1}$ weights where N_l and N_{l+1} are the numbers of nodes within layers l and $l+1$ respectively.

Each node corresponds to an activation function, a method non-linearising the input signal. The activation function used for this work is the rectified linear unit or ReLU for short. It is described by $f(x) = \max(0, x)$, which is a function whose output can be considered to be "firing" or "not firing". Combinations of these allow for complex, non-linear patterns to be learned [93].

An important property for the activation function is for it be differentiable in order to be able to train the network where the ReLU function's advantage is that it is trivial to differentiate, and which is therefore appealing for large networks in which (very) many derivatives must be found.

Given an input vector with f_d dimensions, the operation within the j^{th} node of the first hidden layer would be described by:

$$a_j = f\left(\sum_{i=1}^{N_l} w_{ij} x_i + b_1\right) \tag{4.11}$$

where b_1 is the bias term and w_{ij} is the weight from the i^{th} node in the previous layer to the j^{th} node in the subsequent layer and f is the activation function.

In the next layer, the operation carried out in Equation 4.11 is repeated, now using as inputs the outputs of the previous layer. A network being used for multi-class classification (as in our case) is activated in the final layer by the softmax output as described by Equation 4.12.

$$P(y = j \mid \boldsymbol{x}) = \frac{\exp\left(\frac{\boldsymbol{x^T w_j}}{T}\right)}{\sum_{c=1}^{C} \exp\left(\frac{\boldsymbol{x^T w_c}}{T}\right)} \tag{4.12}$$

where $\boldsymbol{x^T w_j}$ is the output of the final layer of the network for node or class j, C is the total number of classes within the classification task and T is the "temperature"

of the function.

Backpropagation is used to train the network weights. ANNs provide no guarantee of a convex minimisation over error with respect to synaptic weights and overcoming local minima in the optimisation space is a common problem [8], [93].

Along with the architecture of the network, the performance of an MLP can also be heavily influenced by the optimisation scheme used to train the weights. For our work, we found the best performance was achieved through the use of stochastic gradient descent with momentum. This turns the parameter update into a linear combination of the learning gradient in addition to some contribution from the previous update described by: $w_{t+1} = w_t - \eta\nabla\mathcal{C}_t(w_t) + \alpha\Delta w_t$, where w_t is the weight being optimised over, η is the learning rate, $\mathcal{C}_i$ is the cost function being optimised and α is our momentum coefficient. As various batches of data could provide better learning gradients for different classes, the momentum term stops erratic changes to the parameters of the network due to the stochastic sampling of the data. The analogy is that the learning gradients will have a momentum in a certain direction, and any batches which provide erratic results will slow the momentum in this direction but not change it entirely. To tune the hyperparameters of the network we use 5-fold cross-validation and report all of our final results on the held-out test set.

Prediction Interpretation

After training the model it is useful to understand from the clinical perspective why the model has made its predictions and why errors arise. We investigate this by modifying the architecture of our model slightly. We add a layer of weights to the input space that are multiplied element-wise by the inputs changing the function approximator from $f(y \mid x; \theta)$ to $f(y \mid w_{in} \odot x; \theta)$. Having multiplied the inputs, x, by the weights w_{in} we then pass the weights through the softmax function to find the relative importance of each feature to the prediction and then add the entropy of this output to the cost function. We therefore change our cost-function so that it now

becomes:

$$\mathcal{L}(\theta) = -\left(g_s(w_{in}) \log\left(g_s(w_{in})\right) + \sum_j \left(y_j \log\left(\hat{y_j}\right)\right)\right) \tag{4.13}$$

where g_s implies the softmax function, w_{in} are the pre-multiplying weights of the inputs, y_j is the real one-hot label of the prediction, $\hat{y_j}$ is the models predicted distribution over the classes and j is the data point. Using this loss we then use backpropagation as usual and update both θ, the network weights, and w_{in}. The effect of this function is to encourage sparsity in the inputs while maintaining the objective of classifying the patients. This will allow us to see the most important features for this prediction problem. We train until we achieve the same accuracy as was achieved previously, with the knowledge that we have achieved the maximum performance possible with as sparse a feature space as possible.

4.5 Datasets

In this study we considered the patient data collected in the EHR of the OUH NHS Trust as discussed in Chapter 3. The features used for prediction can be found in Tables 3.1 and 3.2. A training set of 60% of the dataset was used and was balanced (on the basis of admitted ward group) leaving 5327 patients for training on. The validation set was 20% of the dataset and testing was also 20% and the classes were kept in the same distribution as the original dataset.

To validate the efficacy of the methodology we implemented the algorithm on the MIMIC-III dataset. This was split into the same train-validation-test proportions as before with only the training set being balanced as before.

4.6 Results and Discussion

4.6.1 Choosing Between Unimodal and Multimodal Curricula

The first design choice to make when building the curriculum is what kind of metric to use to sort the data into batches. We must also consider whether we should design a unimodal or multimodal curriculum. As described previously, we experiment with a unimodal setup (where the full training set is treated as a single mode) and a multimodal setup (where we create n clusters using GMMs, k-means and by class before creating separate curricula for each cluster). Table 4.1 shows the average accuracy scored by the multimodal and unimodal curricula. The table shows the average accuracy scored on the test set when run over 10 different random initialisations. We see that when using the Mahalanobis distance or cosine similarity as entropy metrics for the curriculum, we get best performance in a unimodal setup where we only have one cluster. In the case of using the Wasserstein distance as our entropy metric, the best performance occurs with a bimodal curriculum, however this performance is still worse than that of the unimodal curriculum. Furthermore we experimented with assigning clusters according to label which necessitates seven clusters for this problem. When training a curriculum according to label, none of our experiments exceeded an average of 37% accuracy on the test set and so this setup was discarded. As a result, we choose to use the unimodal Mahalanobis curriculum to run our experiments on. Varying numbers of clusters were also experimented with, however we were unable to beat the performance achieved by the unimodal setup.

4.6.2 The Prediction

The aim of the algorithm is to classify the patient as being admitted to one of previously described seven ward types. Initially, a multinomial logistic regression and an SVM

Table 4.1: A table of the average accuracy achieved by various curriculum learning setups. The entropy metric is chosen between Mahalanobis distance, cosine similarity and Wasserstein distance. We then cluster according to a GMM or k-means.

	Mahalanobis		Cosine		Wasserstein	
no. clusters	GMM	k-m	GMM	k-m	GMM	k-m
1	**0.46**	**0.46**	0.40	0.40	0.39	0.39
2	0.43	0.41	0.44	0.40	0.41	0.40
3	0.40	0.39	0.39	0.38	0.40	0.39
4	0.38	0.35	0.36	0.34	0.38	0.36

were used for the task (trained using stochastic gradient descent). These however provided poor performance, with the prediction accuracy being 14% for both methods, close to that of chance given a seven-class classification. We then implemented our curriculum training methodology on both simple classification models as is undertaken in [98], to determine whether or not the proposed curriculum learning could improve their performance. We found that a simple multinomial logistic regression model had its classification accuracy unchanged with or without curriculum learning, whereas the SVM improved from 14% accuracy to an average of 17% accuracy when using the curriculum only, and to an average of 21% when the curriculum was combined with the proposed multi-armed bandit. Random Forests performed better again achieving a 25% average accuracy on the held-out test set and XGBoost was the best performing of the baselines with a 35% average accuracy.

In Figure 4.1 we implement a feedforward neural network for the hospital admission location problem. Use of the feedforward network provided good performance for the multiclass classification for some classes but not for all as indicated in Table 4.2. The maximum accuracy achieved on the validation and held-out test sets was 39% over all classes. However, it can also be seen from Figure 4.1 that the loss and accuracy plots are very noisy. The five different seeds all provide very different performances at the end of training, with a difference of approximately 10% performance on the validation set as seen in the accuracy plot in Figure 4.1. The range of losses shown in the loss plot indicates to us that after training the five seeds have found different

local minima within the weight space. This indicates that this is not a very stable place from which to launch a non-stationary bandit search of the weight space, as for different seeds we will be starting our optimisation from different locations and our final performance will be dependent on the initial seed.

In Figure 4.2 we repeat the experiment however this time incorporating a curriculum into the training regime. Using a Mahalanobis based curriculum not only achieves a higher maximum accuracy overall (46% over all classes) than stochastic mini-batch training, but also smoothes out the accuracy and loss of the five seeds. As can be seen in Figure 4.2, the variance in the best performing and worst performing seeds is much smaller. We also see in the loss plot that all seeds eventually converge to the same loss, indicating that due to the curriculum all of the seeds have likely converged to a very similar local minimum, due to the bias that was introduced in training with low entropy samples first. This not only improves the performance for the whole classification but also improves the performance of the individual classes that did not perform well initially, which can again be seen in Table 4.2. The losses and the accuracies being much smoother provides us with a stable basis to begin an exploration versus exploitation approach to training the network.

The multi-armed bandit is then incorporated into Figure 4.3, showing how the bandit explores until it finds the best batches to train the network on given what has previously proven successful. We are able to exploit the batches of data in the curriculum to provide us with a better or equal performance to a network trained only using a curriculum. We see in Figure 4.3 that the average accuracy initially decreases due to the exploration that is required and eventually jumps to a value of 52% accuracy overall, the strongest average from any of our training regimes. The performance eventually falls from 52% over all classes due to the algorithm being constrained to continue selecting batches to train on, moving the weights out of the region of the weight space that achieved 52% accuracy.

For all experiments, the performance is recorded and the best performing model

saved out. Each method is trained until the onset of overfitting is exhibited. Figures 4.1 and 4.2 show performance on the training and validation sets, whereas Figure 4.3 shows the performance on the validation set. The validation accuracies reported were also found on the held-out test set. The optimal network architecture found after cross-validation was a network with 3 hidden layers and 100 nodes on the hidden layers, all activated by the ReLU function. The optimal batch size was 90 for stochastic mini-batch training, the temperature of the output softmax was 2 and momentum for stochastic gradient descent was 0.9.

We also experimented using the curricula from [43], [98], [115] and [117] to organise input data. We found that these achieved average accuracies of 40%, 40%, 43% and 42% respectively for curriculum learning, whereas our method achieved 46%. When incorporating the multi-armed bandit, these curricula achieved accuracies of 45%, 46%, 49% and 46%, whereas ours achieved 52%.

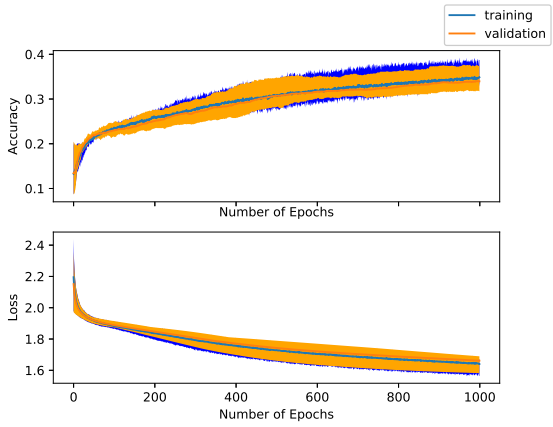


Figure 4.1: Batchwise training for five separate seeds. Shaded regions indicate maximum and minimum performance.

To further examine the efficacy of this method, we carry out an experiment using the publicly available MIMIC-III dataset [19], [107]. We see from Figure 4.4 that the stochastic mini-batch training once again provides highly variable performance with a maximum performance of 61%. Figure 4.5 shows the curriculum regime, once again converging the losses and achieving a better maximum accuracy for all seeds

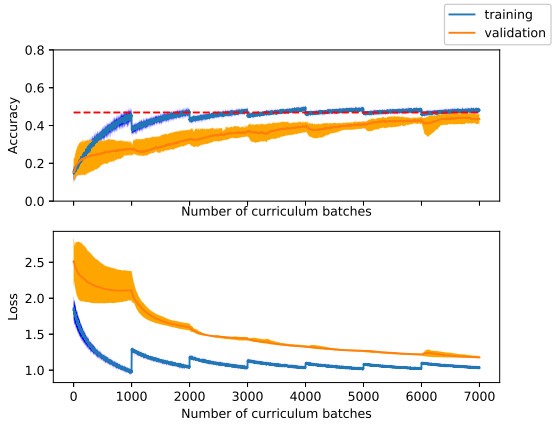


Figure 4.2: Mahalanobis curriculum for five separate seeds. The red line shows the maximum accuracy achieved on the held-out test set.

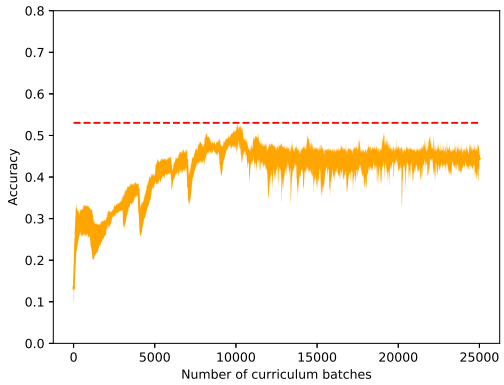


Figure 4.3: Curriculum followed by the MAB for five seeds. The validation set is plotted alone for clarity. The red line shows the maximum accuracy achieved on the validation set.

achieving 66%. Finally, Figure 4.6 shows that our algorithm once again produces the best maximum performance of 69.5% by combining the curriculum regime with the MAB after a brief period of exploration. The curriculum once again smoothes out the losses into a similar minimum in order to provide a stable point from which to launch an exploration of the weight space. The multi-armed bandit then exploits the positioning in the weight space to find a better local minimum. As before, the best performing model is saved out before continuing experimentation with the batches. Figures 4.4 and 4.5 once again report results on training and validations sets and Figure 4.6 is displayed only on the validation set. We again find that the reported

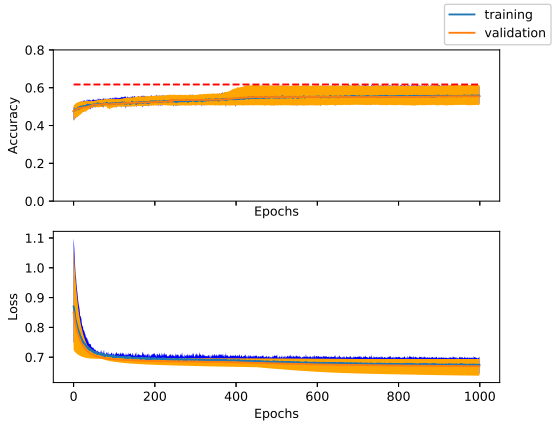


Figure 4.4: Batchwise training for five separate seeds on the MIMIC-III dataset. Shaded regions indicate maximum and minimum performance. The red line shows the maximum accuracy achieved on the validation set and held-out test set.

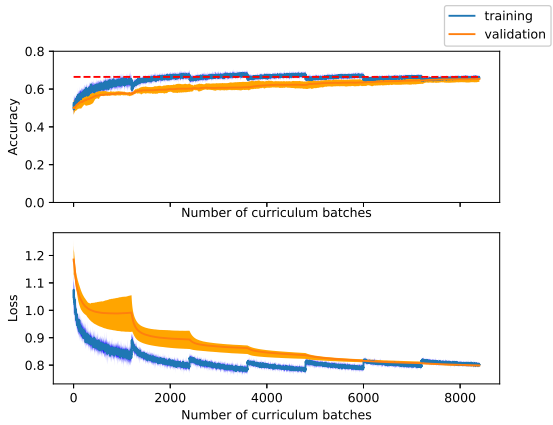


Figure 4.5: Mahalanobis curriculum on the MIMIC-III dataset for five separate seeds. Shaded regions indicate maximum and minimum performance. The red line shows the maximum accuracy achieved on the validation set and held-out test set.

validation accuracies were also found on the held-out test set. We have therefore shown that this training scheme produces a better performance for two separate classification problems from two separate datasets.

4.7 Discussion

To analyse the performance of our approach across the seven ward-types in the OUH dataset we look at the AUCs of the unique classes after training with the different regimes.

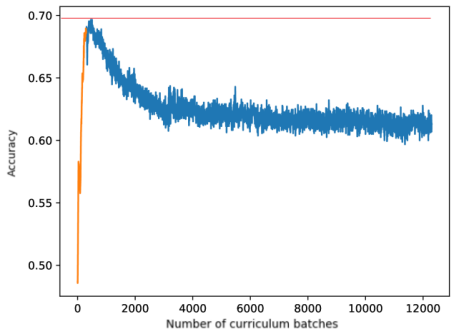


Figure 4.6: Curriculum (*orange*) followed by a multi-armed bandit batch selector (*blue*) on the MIMIC-III dataset. The mean performance of the differently seeded models on the validation set is plotted alone for clarity. The red line shows the maximum accuracy achieved on the validation set and held-out test set.

Table 4.2: Maximum performance of various models on ward type prediction for the individual ward types. We test an SVM, feedforward deep neural network trained by stochastic mini-batch training (ff-NN), curriculum learning with a deep neural network (CL) and our proposed method curriculum learning and multi-armed bandit training (CL-MAB). Chance corresponds to an accuracy of 14% and an AUC of 0.5.

Test Data	Model			
	SVM	ff-NN	CL	CL-MAB
Avg. Acc.	0.14	0.39	0.46	**0.52**
Medical AUC	0.50	**0.67**	**0.67**	**0.67**
Cardiac AUC	0.55	**0.78**	**0.78**	**0.78**
Neuro AUC	0.56	0.51	0.56	**0.60**
Trauma AUC	0.66	**0.75**	**0.75**	**0.75**
ICU AUC	0.65	**0.71**	**0.71**	**0.71**
Surgical AUC	0.50	0.59	**0.63**	**0.63**
General/Obs&Gynae AUC	0.54	0.64	0.66	**0.68**

We see that the best performance is achieved by the combination of the curriculum learner and multi-armed bandit, the incorporation of the latter improving the prediction performance on groups 2 and 6 without detriment to the other classes. We further investigate by extracting the latent representation of our test data from the embedded space of the final layer in the network after training. We then apply the t-SNE algorithm [42] to view the clusters that are formed within that space.

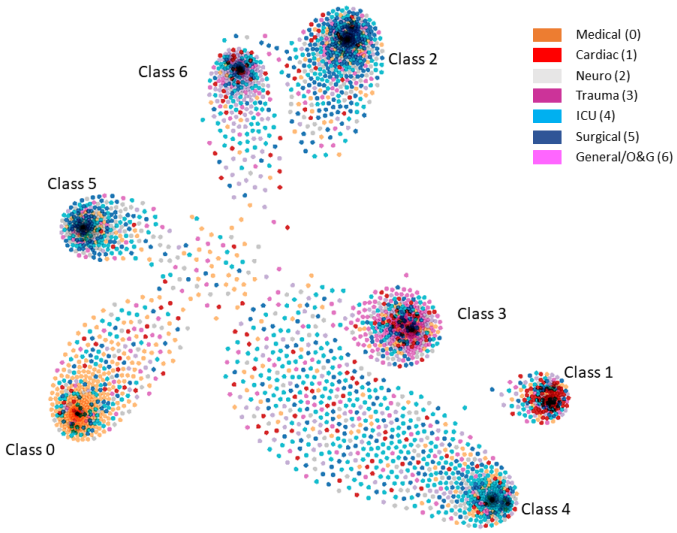


Figure 4.7: Visualisation of clustering of the latent representations of the final layer using the t-SNE algorithm.

The result in Figure 4.7 shows that there are some well-defined clusters, coloured by orange, pink, turquoise, red and lilac. However, there are two clusters (which correspond to classes 2 and 5) which are not clearly defined by colour and this can be explained as they have low AUC values (see Table 4.2).

Feature Importance

To gain a clearer understanding of why the AUCs for the separate classes are different we use the modified architecture (with the pre-multiplying saliency vector) that was described in Section 4.4.7 to interpret feature importance. We train the modified architecture with the unencoded input features. We then extract the trained weights of the inputs, w_{in} which are the ones that are then used in the curriculum learning setup as previously described. The only features that have weights in the sparse vector (and are therefore considered important for the prediction) are listed in Table 4.3 and Figure 4.8. Table 4.3 shows the binary features which were found to be important for prediction. Figure 4.8 shows how frequently previous diagnoses appear for patients admitted to a certain ward type. These were compared with the previous diagnoses of the patients admitted to each ward type for the whole dataset and where there was

overlap in the diagnoses, these were boxed and labelled as seen in Figure 4.8(b).

Table 4.3: Proportion of patients from each class who had the following tests carried out. All patients in this table were correctly predicted by the model. Values $\geq 15\%$ are highlighted in bold. Classes are: 0 - medical rest, 1 - cardiac, 2 - neurology, 3 - trauma, 4 - ICU, 5 - surgical rest, 6 - general rest.

Feature	Predicted and Actual Class						
	0	1	2	3	4	5	6
Pregnancy test	0.00	0.00	0.01	0.00	0.03	0.00	**0.19**
Blood culture	**0.16**	0.03	0.10	0.02	**0.31**	0.09	0.09
Cardiac enzymes	0.06	**0.58**	0.12	0.09	**0.27**	**0.17**	0.07
Blood cross-match	0.06	0.09	**0.39**	**0.45**	**0.42**	0.00	**0.17**
Frequent Flier	0.04	0.04	0.04	0.02	0.03	**0.22**	0.06

We see from Table 4.3 that the model has learned a distribution based on these 'important' features. These tables explain why the model does not predict accurately for all patients.

- The blood culture test is predominantly carried out for patients who go on to be admitted to classes 0 and 4 which correspond to the '*medical*' wards and the ICUs. This test is used to check for bloodstream infection which can have serious complications and as a result, the model has learned to associate a request for this test with admission under medicine, representing most patients admitted with an infection, and with the need for intensive care.

- Cardiac enzyme tests are those that are used to indicate a heart attack has occurred or is occurring, or if there is blockage in the heart's arteries [7]. It is therefore unsurprising that the model associates a request for this test with class 1, which corresponds to the *'cardiac'* ward types.

- Blood cross-matching (the procedure of searching for appropriate blood to use if a transfusion is required) is a common test asked for from patients who are usually admitted to classes 2, 3 and 4 corresponding to *'neuro'*, *'trauma'* and

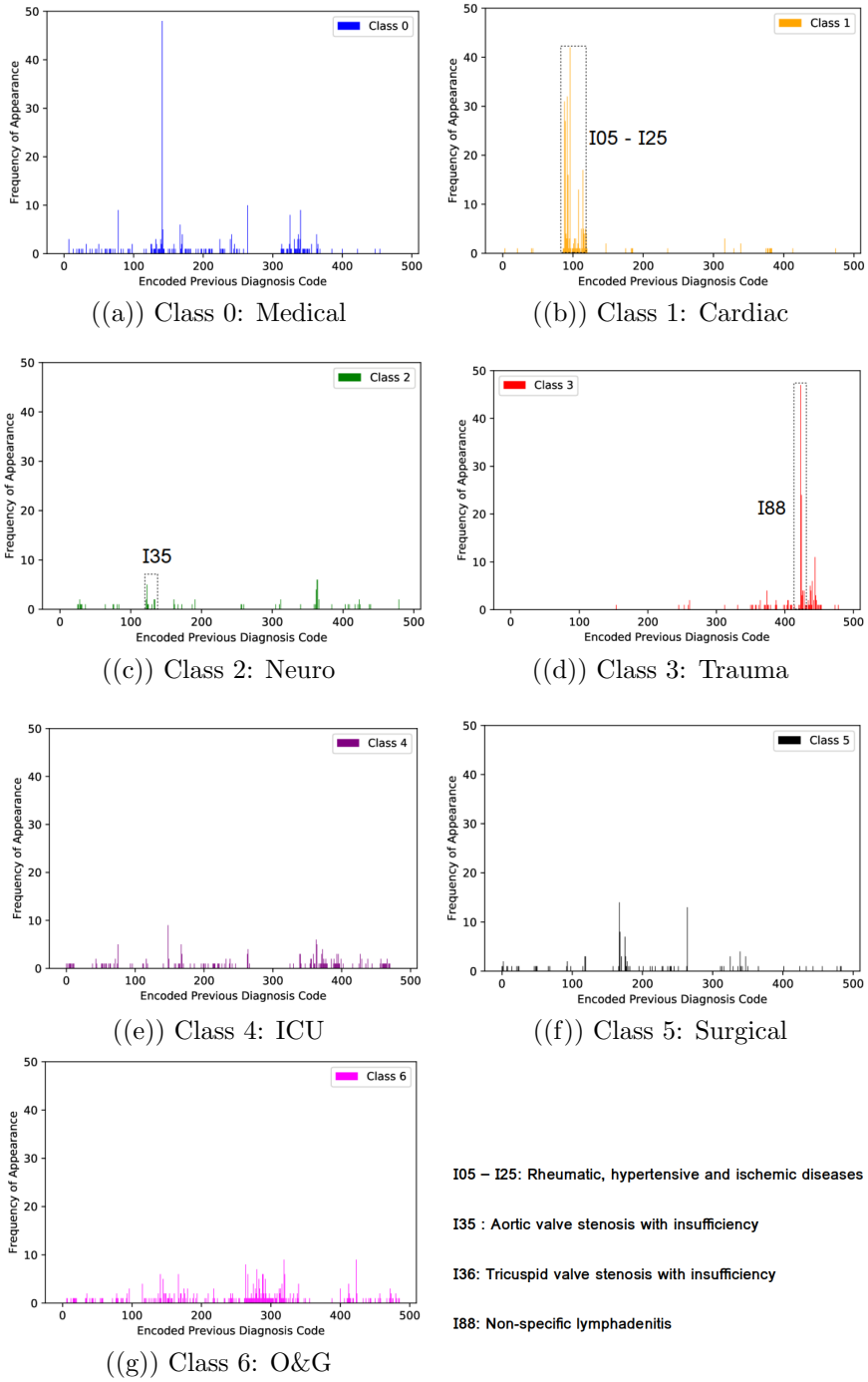


Figure 4.8: Historical diagnosis code by admission location for correctly predicted patients. Plots are of frequency of appearance by encoded diagnosis code.

'ICU' ward types respectively. This represents the subset of patients likely to require surgery during their admission.

- The frequent flier flag is mostly associated with patients admitted to surgical

wards (class 5). It is not immediately clear why this is. However, it is hypothesised that this ward function may act as a spare space where beds are available for emptying the ED.

- Pregnancy tests are correlated with the general rest wards (class 6). This likely reflects that admissions under obstetrics and gynaecology fall into this group of patients.

Using the tables and figures we can now see how the predictions are determined.

1. Class 0 (*'Medical'* ward type) are mainly predicted by a blood culture test request and no other tests.

2. Class 1 (*'Cardiac'* ward type) are dominated by having only a cardiac enzyme test requested and no others. Presence of a previous diagnosis of a rheumatic, hypertensive or ischaemic disease further increases the likelihood of admission.

3. Class 2 (*'Neuro'* ward type) are predicted by a blood cross-matching request and previous diagnoses, the most prevalent of which correspond to 'aortic valve stenosis with insufficiency'. These are documented in the literature to highly correlate with stroke [20], possibly explaining the reason for these patients' predicted admission to *Neuro*. Upon investigation of the dataset, 86% of the patients who had been previously diagnosed with aortic stenosis would go on to have a subsequent diagnosis associated with cerebral infarction or stroke.

4. Class 3 (*'Trauma'* ward type) is characterised again by a blood cross-match but with different previous diagnoses. In this instance the diagnosis (indicated by the red spikes in Figure 4.8(d)) corresponds to nonspecific lymphadenitis or swelling of the lymph nodes. This is not descriptive enough to gain a physical insight as to why this classification is made. These patients are generally older than the average age of the population of the dataset (65 years old versus 60 years old generally) and are at a greater risk of previous accidental harm. It

is therefore expected that our CL-MAB algorithm has associated a common previous diagnosis code with the greater age of this population and therefore a greater risk of injury. Further investigation would be required to verify that this indeed is the association learned by the algorithm for this patient subset.

5. Class 4 (*'ICU'* ward type) is characterised by a request for blood culture, cardiac enzymes and blood cross-matching. This wide spectrum of tests requested is indicative of the critical condition the patient is likely to be in upon presentation.

6. Class 5 (*'Surgical'* ward type) is also characterised by a Cardiac Enzyme test requested. It is also not clear if having a 'Frequent Flier' flag causes the prediction.

7. The cause of a prediction of class 6 (*'General / O&G'* ward type) is mainly due to a pregnancy test and this is most likely due to the inclusion of O&G admissions in this ward type.

The overlap in important features for the *'neuro'* and *'trauma'* classes may also explain the difference in AUCs reported in Table 4.2. It is very possible that many *'neuro'* admissions are predicted to be *'trauma'* due to the similarity in their input importance. This may also be the case for *'surgical'* and *'cardiac'* admissions. To improve our model, it will be important to determine if there are further specific features that can be obtained at ED triage time for all classes that may help distinguish these classes.

For comparison we check the proportion of these features present for the whole population using the real labels of what ward type each patient was admitted to. These proportions are shown in Table 4.4.

From Table 4.4 we see that the model has learned the underlying distribution quite accurately. The exceptions are in classes 5 (*'Surgical'*) and 6 (*'General/O&G'*). For Class 6, we see that the pregnancy test is not very important for prediction but the blood cross-match is. This motivates the introduction of a gender-specific

Table 4.4: The total population of the dataset is tabulated here using their real ward types as the label. Values $\geq 15\%$ are highlighted in bold.

Feature	Actual Class						
	0	1	2	3	4	5	6
Pregnancy test	0.01	0.00	0.01	0.01	0.02	0.00	0.07
Blood culture	**0.18**	0.05	0.10	0.03	**0.26**	**0.29**	0.12
Cardiac enzymes	0.13	**0.51**	0.09	0.11	**0.25**	**0.24**	0.14
Blood cross-match	0.08	0.11	**0.33**	**0.42**	**0.35**	0.07	**0.16**
Frequent Flier	0.05	0.05	0.05	0.03	0.05	0.05	0.05

model. For Class 5, the model has not learned that a blood culture test request as well as a cardiac enzymes test request are most indicative for this class and not the frequent flier flag. This may explain the reason for the poor performance in AUC for Class 5. Class 2 (*'Neuro'*) also has a relatively poor performance and based on the proportions shown in Tables 4.3 and 4.4, it could be due to blood cross-matching tests being important features for classes 3 (*'Trauma'*) and 4 (*'ICU'*) as well. To further improve the performance of the model we will investigate further features that are more specific to the individual ward types, as well as developing separate models for male and female patients. Another limitation of our work is that some patient admissions require specific equipment which can only be found in certain wards [100]. A future model should incorporate this requirement to maximise usefulness of the model to clinicians.

To further examine the usefulness of the model to clinical staff we investigate its performance plotted over time. Figure 4.9 shows how the model performance varies with time. The red shaded regions indicate the winter flu seasons where the ED is busiest with admissions. We see that the model does not suffer significant degradation in performance due to winter pressures. In addition, in three out of four of the flu seasons the model performs better than the yearly average. We believe this could be due to the grouping of wards into ward functions as opposed to individual wards, which bypasses the problem of patients being admitted to a ward atypical for their

condition but still capable of treating the patient. However, this may also be due to our preprocessing step of removing patients obviously admitted to an inappropriate ward for their diagnosis. While this filters the obvious cases, it does not remove all such cases from the dataset. We therefore believe that this model could still be useful in helping clinicians during busy periods to request bed space well in advance of the need for it, to allow timely admission of patients from the ED and into the hospital ward.

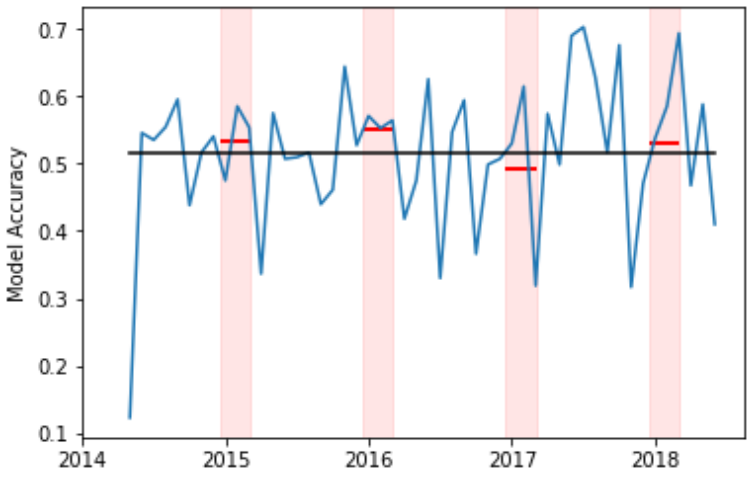


Figure 4.9: Performance of the model by month for the 4 years of data included in the dataset. The black solid line is the overall accuracy over the four years. The red shaded area shows the winter flu seasons and the solid red lines show the average performance of the model during those flu seasons.

4.8 Summary

We have presented a novel method of training and regularising deep learning model with the aim of predicting where a patient presented to the ED will be admitted in an OUH NHS Trust hospital. This prediction will aid in the provision of timely care and treatment for the patient and those still in the ED. Our model achieves AUC values between 0.60 and 0.78 for the individual ward types. Furthermore, our model also provides some insight as to the cause of the predictions, allowing the user to incorporate more important features for individual ward types in the future. We

believe this may be useful for ensuring timely admission to hospital and reducing the time to delivery of care. This will in turn improve the quality of care for patients still in the ED due to less crowding. This work may also be useful for resource prediction and optimisation in hospitals more generally.

The model presented in this work is first trained using a curriculum and then using the curriculum batches a multi-armed bandit is employed to improve the performance. While the algorithm described in Algorithm 2 is non-stationary, it is weakly non-stationary relying on the number of pulls of a certain batch to reduce the probability of choosing said batch. As a result, we will improve this by turning this problem into a full reinforcement learning problem. Treating the weights of the network as the state space, we will train a policy to select the best action to take (batch to train on) given the state space. We believe this will be a much more effective method of training due to the information provided to the trainer about the state of the weights of the network.

We would also like to further investigate features that can be obtained from the ED which correlate highly with the individual ward types. In doing so we will be able to reduce the input feature space and advise clinicians in the ED what needs to be measured for this prediction problem. It is hoped that by doing this, we will be able to mitigate the problem of missing features which can commonly happen in models with large input spaces. We will continue to investigate methods of identifying when patients were admitted to wards that were not ideal for their treatment. We believe that finding these cases will help to improve the performance of our models due to their reliance on historical data. We will also seek to integrate data on the equipment used during a patient stay to better inform the model of which wards are appropriate for admission.

Chapter 5

Student-Teacher Curriculum Learning

Chapter Abstract

We have seen that accurate and reliable prediction of hospital admission location is important due to resource-constraints and space availability in a clinical setting. Allocating resource in a timely and decisive manner is especially important given that changes to resource plans can be slow and inefficient in a busy clinical setting. In the previous chapter we showed that this prediction is indeed possible. However in order to maximise the usefulness of such a prediction, we must make sure that the prediction is as accurate as possible when it is first made. In this chapter we propose a student-teacher network trained via reinforcement learning to maximise the performance of a model predicting the hospital admission location of patients in the emergency department. A representation of the weights of the student network is treated as the state and is fed as an input to the teacher network. The teacher network's action is to select the most appropriate batch of data to train the student network on, from a training set sorted according to some measurement of entropy. By validating on three datasets, not only do we show that our approach outperforms state-of-the-art methods on tabular data and performs competitively on image recognition, but also that novel curricula are learned by the teacher network. We demonstrate experimentally that the teacher network can actively learn about the student network and guide it to achieve better performance than if trained alone. We test the robustness of the policies learned

by the teacher network and show how the teaching strategies learned by the teacher can be transferred to other classification problems for strong performance. The work included in this chapter was published as a conference paper at the International Conference on Machine Learning (ICML) 2020 [196].

5.1 Introduction

As has been seen in the previous chapter, a major problem that many hospital wards face is the timely preparation of beds for patients from the emergency department (ED) who are due to be admitted as inpatients [135]. The average time between initial examination (triage) in the ED and a request for a bed in the hospital is 4 hours [74] and the only information available to staff about the patient is that which is collected at triage. This information can be helpful to deduce the patient's condition, which in turn can be used to predict which of the relevant inpatient wards needs to prepare space. In the previous chapter, we focused on creating a prediction of the likely care pathway that a patient would take upon admission to hospital. In this chapter we investigate how we can make this prediction as accurate as possible as quickly as possible for the sake of allocation resource and planning.

When it is decided that a patient will require hospitalisation, the ward is often requested in a reactive manner after 4 hours in the ED have passed. Predictively allocating patients to wards however, should allow for the timely care and admission of the patient as well as allocation of resource to the relevant departments. Furthermore, this may contribute to reducing crowding of the ED as well as patient waiting times. With this in mind, we present a method of predicting where in a hospital emergency patients will be admitted after being triaged using a combination of reinforcement and curriculum learning.

Curriculum learning was introduced to train a neural network in a similar manner to that in which humans are educated. The approach stems from the observation

that children in schools learn by beginning with simple ideas and progressing on to more complex topics; it is believed that neural networks may also benefit from this structured approach to learning. By initially presenting the network with data that are 'easier' to fit, the optimisation surface (of network prediction error versus network parameters) is more likely to be convex [43].

In this chapter we propose using neural networks for classification by exploiting this notion of a curriculum and extending the teacher-student curriculum learning method of [128]. We use two networks, where one (the teacher) is trained with information about the state of the other (the student) in order to guide training and maximise performance. A teacher's observations of many previous students provide them with an idea of how a student can best learn [97] using a pre-defined curriculum. This work aims to mimic this action of learning from an agent (which is itself learning), with the aim of creating a policy (the function that maps a state to an action) to guide a future agent to achieve successful performance. Section 5.2 discusses related work and Section 5.4 outlines the setup of the student-teacher network. Results are shown in Section 5.7 and we discuss the curricula generated by the teachers in Section 5.8 before outlining the limitations of this approach in Section 5.10.

5.2 Related Work

In this work, we create a teacher agent that learns to train a feedforward student network by selecting the data to present to the student. Previously it was seen that we could organise our data into batches according to a curriculum. This defined the 'easiness' of a batch of data according to the entropy within the batch; i.e., a batch B_i is said to be easier than batch B_j if $H[B_i] < H[B_j]$, where H is the entropy of the batch. We also saw that while training according to a curriculum gave improvements in performance, the ordering of the curriculum does not necessarily need to be strictly from low entropy to high. It has been shown in [140] and [98] amongst others that

training neural networks in a structured fashion by presenting batches from easiest to hardest leads to improvement in performance of the network, as well as increasing the speed of training. However, we also see in the works of [168] and [43] that training in an 'anti-curriculum' (hardest to easiest data) can also lead to improved performance. A curriculum may also be defined for specific problem sets, with results from [85] and [88] showing that a curriculum tailored for a specific task provided better results for that task than a general curriculum. This is indicative that training according to some type of structure is important for learning, and that structure may be task specific.

Student-teacher curriculum learning has been explored in [128] where a similar initial configuration to that proposed here is used. Progress of the student is measured through its improvement in task performance, which in turn affects the probability of selecting a curriculum batch on which to train. The authors proposed four different algorithms to monitor the progress of the student. These are (i) the use of a non-stationary bandit to select the next training batch, (ii)-(iii) using linear regression and a windowed linear regression on the task accuracy to predict the batch most likely to provide the greatest improvement in performance, and (iv) using Thompson sampling to select the next batch. While these approaches have an effective performance on those specific problems presented in the existing studies, the selection of data is not based on information pertaining to the state of the student (but instead is based on prior information of the student in its previous state, and from previous performance of the algorithms). In this work, we propose using knowledge of the current state of the student to develop a policy for the teacher to train future students.

Reinforcement learning has proven to be highly effective in many applications, canonically within the domain of games [96], [124]. The training of a neural network can also be seen as a game where the task is to choose the right combination of training data to maximise a score, which in this case is the prediction accuracy. This makes reinforcement learning an attractive choice for training the teacher network. Our approach has parallels with the field of hierarchical learning, in which a higher-level

agent guides the training of a lower-level agent [109]. However, to the best of the author's knowledge, none of the methods used in hierarchical learning operate directly on the weight space of the lower-level agent.

5.3 Novelty

The contributions of this work include the development of a student-teacher learner, where the teacher is trained using reinforcement learning which, to the best of the our knowledge, is the first work to propose such a student-teacher configuration. It is also believed to be the first work to use a representation of the weights extracted from the student as inputs to the teacher. This method has demonstrably better performance for our application than all others to which it was compared. We believe that the resulting classifier will prove a useful tool for hospitals in order to predict the resources required ahead of time and improve the movement of patients out of the ED and into the wards.

5.4 Methodology

5.4.1 Data Pre-Processing

The first task for our model is to organise data into batches according to their complexity. As previously mentioned, this can be achieved according to the entropy of the data (among other methods not described here for brevity). For tabular data (categorical and numerical data such as the information recorded at triage), we choose to represent the entropy through the use of the Mahalanobis distance. The Mahalanobis distance is defined in Equation 4.3. As before, we encode the input data prior to organising into a curriculum using a denoising autoencoder for the Mahalanobis distance to be defined in a scalar space.

Once again, two methods of creating the batches from the teacher output were

investigated:

1. All batches are separate and contain separate training data; i.e., if B_0 is the batch with the lowest $d(\phi(x_n))$ and B_N that with the highest, then $B_0 \cap B_1 \cap \ldots B_N = \varnothing$
2. Batches are cumulative supersets; i.e., $B_0 \subset B_1 \subset \ldots B_N$.

5.4.2 Reinforcement Learning

Reinforcement learning refers to a group of methods that aim to allow agents to learn from their experience rather than through feedback from a ground truth (which corresponds to supervised learning) [16]. This is heavily inspired by the field of behavioural psychology where studies have shown that one of the effective routes to self-learning is through a reward-based mechanism [77]. In a reward-based learning setup, an agent is rewarded (and this is open to interpretation) according to whether or not they successfully complete a task. Through an iterative approach of trial and error and reward for task completion, the agent eventually learns the appropriate behaviours that allow for their task to be complete and therefore their reward to be earned.

Reinforcement learning problems are generally posed as Markov Decision Processes (MDPs). These can be characterised by the (S, A, P, R, S') tuple where S represents the state the agent is in, A represents the action that the agent is to take, P is the probability distribution over next states that the agent can find itself in, i.e., $P(s_{t+1} = s' \mid s_t = s, a_t = a)$. R is the reward as a result of the action taken, A, and S' is the next state the agent is in after taking the action. Reinforcement learning problems can further be decomposed into model-based and model-free variants. In the model-free variant, we have no understanding or model of the transition dynamics of the environment and therefore P is left out of the tuple.

Inherent to reinforcement learning is the concept of the value function. This function is a method of quantifying whether being in a certain state, S is beneficial to

achieve the goal of the agent or not. A higher value corresponds to there being greater potential reward in the future from that state. The value function is described by:

$$V^{\pi}(s) = R(s,a) + \gamma \sum_{s'} P(s' \mid s,a) V^{\pi}(s') \tag{5.1}$$

where R is our reward for that state and action pair, π is the current policy being followed and $\gamma \in [0,1]$ is the discount factor. The discount factor is a way of incorporating the potential for future rewards. A discount factor of 0 would prioritise immediate rewards by eliminating the second term and therefore defining the value as the reward gained by the next action taken from the current state. A discount factor of 1 fully considers the future potential reward which is encoded in the value function (of course the value is dependent on the policy being followed). More informally, what Equation 5.1 tells us is that the value of being in state s corresponds to the reward you gain plus any potential future reward you can access from that state. If the agent has an optimal policy then following the Bellman optimality equation as defined in Equation 5.2 will lead to the greatest reward.

$$V^{\pi^*}(s) = \max_{a} \{ R(s,a) + \gamma \sum_{s'} P(s' \mid s,a) V^{\pi^*}(s') \} \tag{5.2}$$

This equation uses π^*, the optimal policy and takes the actions that maximise the total value.

Knowing which state maximises the value is useful but a more granular and actionable measurement is knowing which state-action pair maximises value. In this setup, every state will have m possible actions that can be taken from it meaning it would have m state-action values (i.e., the value of taking a particular action from that state). This has equivalence with the Bellman optimality equation and can be

expressed as:

$$Q^{\pi^*}(s,a) = \max_a \{R(s,a) + \gamma \sum_{s'} P(s' \mid s,a)\, Q^{\pi^*}(s',a')\} \tag{5.3}$$

where V^π, the value, has been replaced with Q^π, the state-action value. The values estimated using Equation 5.3 are called Q-values.

When the state space is small, the Q-values can be computed using various reinforcement learning algorithms such as policy or value iteration [16]. With these methods, we allow the agent to traverse the environment until a reward is found and then use Equations 5.1 or 5.2 to update the values of the states iteratively. In policy iteration we update the policy by evaluating the policy and then improving it directly. In value iteration we allow the agent to discover the optimal values of the states and then assign the policy once the optimum values are known simply by following the Q-values that maximise discounted return.

However once the state space starts to become very large in size this becomes intractable due to the large numbers of updates that would need to occur. This is where the domain of deep reinforcement learning was introduced. The Q-function can be approximated using a function approximator such as a neural network. This led to the introduction of the Deep Q-Network (DQN) [95] and this is what is explored in this chapter.

Temporal Difference Learning

Temporal difference learning is a method of training reinforcement learning algorithms by bootstrapping from the current value estimates of the value function. The temporal difference update is given by:

$$V(s) \leftarrow V(s) + \eta\,(r + \gamma V(s') - V(s)) \tag{5.4}$$

where $r + \gamma V(s')$ is the target in this loss function. Using this method allows

learning to occur before the final outcome is known, unlike Monte Carlo methods, which require the agent to complete the task before assigning reward. As a result, Monte Carlo methods inevitably have much greater variance in their training and so take significantly longer to train. By using the current value function as a target (as done in Equation 5.4, we induce a bias in the learning. This allows learning to occur earlier, but at the expense of having learning occur under an intrinsic bias according to the seeding of our initial value function for parameterised methods. A family of methods that attempt to bridge the gap between these two extremes is called TD-λ. Here λ corresponds to the degree with which we reduce bias in the value estimate. For $\lambda = 1$ a full Monte Carlo update is used and for $\lambda = 0$ a biased estimate is used. An intermediate value of λ would consider n steps of experience as opposed to one. For our experiments we use TD-0 updates.

Discrete and Continuous Action Spaces

We discussed that as the state-space increases in size, function approximation methods become more appropriate for approximating the Q-function as is seen in DQN. However, there are also domains where the action space is also very large in size. In these setups, we again turn to function approximation to overcome the large size. We parameterise the action selection using a neural network and term this the 'actor' as this is the agent that takes the actions. The output of the actor is fed into the 'critic' which uses the approximated action to calculate a Q-value. The only difference in this case is that the actions are approximated using a neural network. These approaches are aptly named actor-critic methods.

5.4.3 Teacher Network

Figure 5.1 shows the overall architecture for the student-teacher network being proposed in this chapter. The teacher network, as most reinforcement learning agents, is characterised by the state, action, reward, next state tuple (*s,a,r,s'*) [16]. Using these

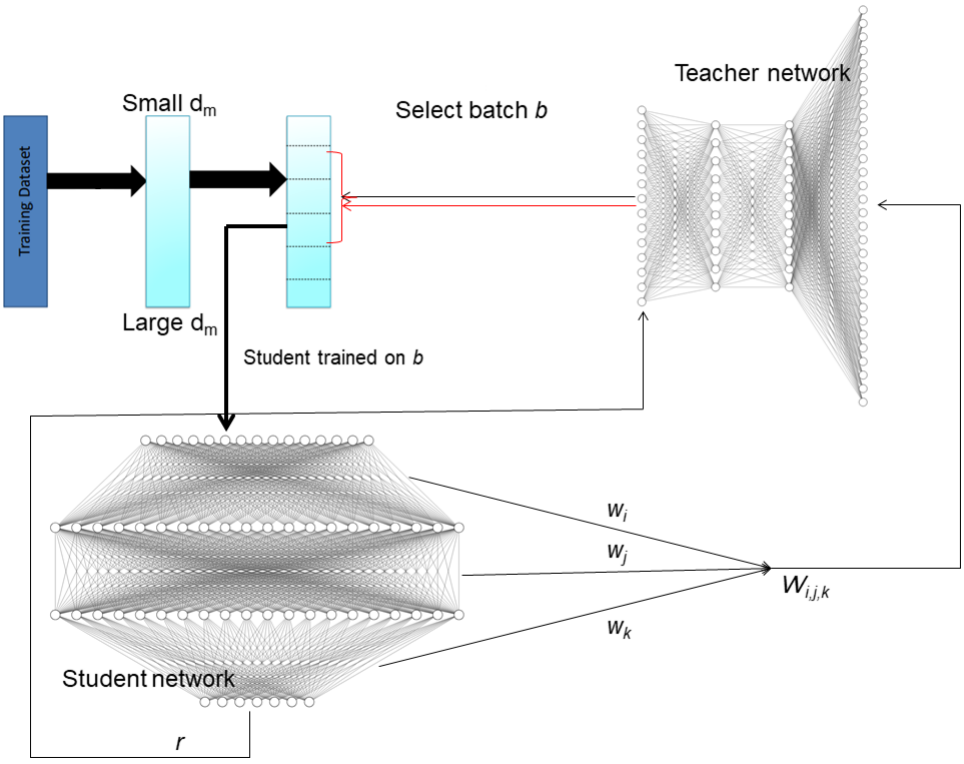


Figure 5.1: The architecture of the model we are proposing. The data is separated according to d_m, the Mahalanobis distance. The reward from the student, r is fed back to the teacher, as well as a representation of the weights from each layer of the student network. The first output of the teacher (black) is where along the sorted data to sample from, the second (red) is how much data around this to include in the batch.

we are able to discern the value of the student being in a given state and taking a certain action (being trained on certain data) for a given task.

We seek to discover a curriculum that will improve the performance of the student over alternative training methods. As previously mentioned, we aim to condition on the current configuration of the student and so we define the *state* therefore to be the weights of the student network, i.e., a representation of the weights between all of the nodes for all layers as described in Section 5.4.5.

The *actions* of the teacher are selecting the ideal batch of training data that are organised by a curriculum, to train the student network. This is done through the policy by selecting the appropriate data to train with, based on the output as

described in Section 5.4.4. Note that in this problem, the number of actions can change depending on how many batches we define our curriculum to have, or the method with which we select data to create a batch.

The *reward* is given by the improvement in accuracy of the student compared to the previous accuracy achieved before the training data batch was trained, i.e., $r_t = \delta_t - \delta_{t-1}$ at step t where δ is the accuracy of the student on the training set. As before, this would lead to overfitting if only trying to maximise the training accuracy. As a result, we measure the student's accuracy on the training set and the validation set jointly by multiplying them as $(\delta_t - \delta_{t-1})\,\delta_v$ where δ_v is the gradient in the validation accuracy to formulate the new reward. This way if overfitting occurs, the validation accuracy reduces thereby reducing the overall reward. If both the gradients are negative, we multiply this by -1 in order to ensure that we do not reward non-advantageous state-action pairs.

Our *next state*, $\boldsymbol{s'}$ is the updated weights of the student network after training on the batch chosen by the teacher.

To train the teacher we must make use of the Bellman equation. This describes the value of an agent (any model or actor which has a state and the capability to take actions) being in a certain state for a given task. It can also describe the value of an agent being in a certain state and taking a certain action for that same task (Q-value) [69]. The equation is described by:

$$Q^*(s,a) = \mathbb{E}_{s'}[r + \gamma \max_{a'} Q^*(s',a') \mid s,a] \tag{5.5}$$

where Q^* is the optimal action-value function, r is the reward of the action that was taken, γ is the discount factor, s, a and s', a' are the current state and action and the next state and action respectively. The discount factor is a way of weighting whether future or immediate rewards are important, with high values of γ favouring a long-term reward and low values favouring immediate reward. The discount factor is

generally treated as a hyper-parameter.

The intuition behind Equation 5.5 is that, if we have a function Q^* which tells us the value of being in a certain state and taking a certain action, then naturally, optimal behaviour is given by choosing the next action which maximises this.

As the state space of student weights is very large, a function for the Q-values must be approximated [17]. This can be done using function approximation methods such as deep neural networks. Now that $Q(s, a)$ is dependent upon network parameters, θ, it becomes $Q(s, a; \theta)$ (note that Q here is the action-value function for the current policy and not Q^*, the optimal policy) and the loss function to train our teacher network is given by the temporal difference loss:

$$\mathcal{L}(\theta_i) = \left(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i)\right)^2 \tag{5.6}$$

where θ^- are the parameters of the target network (the version of the teacher that is held constant for K steps to stabilise training as described in [95]).

Experience replay is used to decorrelate the data being trained on with the current policy. This is required to stabilise the learning of the Q-function. A buffer, R is kept of the previous (s, a, r, s') tuples that the teacher has encountered. A random sample of R of size m is then taken and is presented to the teacher as a batch for training. We train the teacher on the students most recent experience for m steps and then train the teacher on the random sample of historic experience of size m.

The architecture used for the teacher network is a fully-connected feedforward neural network with 3 hidden layers and 50 nodes in each layer. The hidden layer nodes are activated by the ReLU function and a dropout rate of 20% is used to prevent overfitting. Stochastic gradient descent with momentum [78] is used to train the network weights with a momentum value of 0.9. A discount factor of 0.95 is used and the target network is updated after every 20 times the non-target network is updated. An experience replay batch size of 10 is used after every 10 updates on the non-target

network.

5.4.4 The Teacher Action Space

Maximum flexibility in data selection would be to allow for a continuous action space, such as making every entry of data in the training set a batch of data in itself. We experiment using DQN (for discrete actions) and deep deterministic policy gradients (DDPG) (for continuous actions) as described in [94] which allows us to parameterise the action selection. With the latter approach, the teacher, g, consists of two networks, an actor and a critic. The actor is a network which selects the actions to take and the critic evaluates the 'value' of being in a particular state. The actor, parameterised by θ^{μ} (of which there is also a target actor), takes the encoded state of the student, f, as input and has two outputs. The first is the index along the curriculum (after ranking our training data according to the curriculum) at which we will centre our batch and the second is the width around this datapoint that we will expand our batch to. For example if our first output gives us the value 30 and the second output gives 50, then we will centre our curriculum batch on the datapoint in the curriculum indexed at 30 and select everything 25 below and 25 above this datapoint as a batch of data to train on.

The critic, parameterised by θ^{Q} (of which there is also a target critic) has two outputs (for our two independent actions, index selection and batch width selection) which are the Q-values to which the reward is added for training. The input to the critic is the same as the input to the actor (the state of the student) except it is altered by concatenating onto this the actions selected by the actor which also have some exploration noise applied. The critic is then trained with loss as in Equation 5.6. The actor is updated using the following loss:

$$\nabla\theta^{\mu} J \approx \frac{1}{k} \sum_{n} \nabla_{a} Q\left(s, a \mid \theta^{Q}\right) \mid_{s=s_i, a=\mu(s_i)} \nabla_{\theta^{\mu}} \mu\left(s \mid \theta^{\mu}\right) \tag{5.7}$$

where Q is the Q-function and μ is the policy. For a derivation of this please see [94]. A pseudocode is included in Algorithm 3 for a detailed description of how the teacher network is trained. We also experiment with teachers with discrete action spaces and

Algorithm 3 The student-teacher training routine for discrete batches using the DDPG algorithm

procedure INITIALISATION
    Training dataset organised into N batches of Mahalnobis curriculum
    initialise teacher critic network, Q, actor network, μ
    initialise target teacher and actor, Q_T, μ_T
    initialise random process $\mathcal{N}$ for action exploration
    initialise replay buffer, R, select batchsize of replay, m
    select update frequency value, u
    select stable update value, τ
    for x in X students **do**:
        initialise student network, f_x
        for i in I iterations **do**:
            Extract state of f_x, s_i
            Select action $a_i = \mu\left(s_i \mid \theta^{\mu}\right) + \mathcal{N}_i$ according to current policy and exploration noise
            Execute action a_i and observe reward r_i based on improvement in performance on validation set and observe new state s_{i+1}
            Store transition (s_i, a_i, r_i, s_{i+1}) in replay buffer, R
            Sample random mini-batch of n transition tuples from R
            Set $y_i = r_i + \gamma Q_T\left(s_{i+1}, \mu_T\left(s_{i+1} \mid \theta^{\mu_T}\right) \mid \theta^{Q_T}\right)$
            Update critic by minimising: $L = \frac{1}{m}\sum_i \left(y_i - Q\left(s_i, a_i \mid \theta^{Q}\right)\right)^2$
            Update actor using: $\nabla\theta^{\mu} J \simeq \frac{1}{m}\sum_i \nabla_a Q\left(s, a \mid \theta^{Q}\right) \nabla_{\theta^{\mu}} \mu\left(s \mid \theta^{\mu}\right)$
            if i mod U $= 0$ **then**
                Update target networks:
                $\theta^{Q_T} \leftarrow \tau\theta^{Q} + (1-\tau)\,\theta^{Q_T}$
                $\theta^{\mu_T} \leftarrow \tau\theta^{\mu} + (1-\tau)\,\theta^{\mu_T}$
            else
                continue

compare the performance. In the discrete action space, the teacher selects one of the curriculum batches to train with at any given iteration. A second head can be added to the output of the DQN to select the number of batches around this that will be concatenated to form a batch. However, as the batchsize is generally quite large for the discrete domain (as the dataset is split into a smaller number of batches meaning more data are present in each batch), it was found experimentally that training on

one batch at a time was more effective. The discrete action space teachers are trained using more appropriate algorithms such as deep Q-networks (DQN). A pseudocode for this type of teacher can be found in Algorithm 4.

Algorithm 4 The student-teacher training routine for discrete batches using the DQN algorithm

procedure INITIALISATION
    Training dataset organised into N batches of Mahalanobis curriculum
    Initialise teacher network, g
    Initialise target teacher by copying predictor teacher, g^T
    Select value of frequency of target network update and batchsize of replay data, M
    for x in X students **do**:
        Initialise student network, f_x
        for i in I iterations **do**:
            Extract state of f_x, s
            if $i = 0$ **then**
                Train student on random batch (action), a
            else
                Select a with highest Q-value from $g(s)$ according to a linearly decaying ϵ-greedy policy with respect to I
            Train student (f_x) on action selected
            Record performance improvement of student on training set and validation set and multiply for overall reward, r
            Add r to the output of $g^T(s)$ corresponding to the action taken to achieve this reward
            Use the error between outputs of g and g^T to backpropagate over the weights of g
            Save s, a, r and next state, s' into replay buffer
            if i mod $M = 0$ **then**
                Sample M samples from replay buffer to train g with
                update g^T with new state of g
            else
                continue

5.4.5 Student Network

The state of the student network is defined to be the matrix of weights of each layer in the network i.e, W^{ij}, where W^{ij} indicates the matrix of weights between layers i and j. W^{ij} is of size $M_i \times M_j$ where M_i and M_j are the number of nodes in layers i

and j respectively. As the size of these matrices is dependent upon the number of nodes in each layer, we seek a more compact representation of these matrices. We do this by comparing each row in a weight matrix to a fixed reference vector. For example, $W^{ij}_{1:} \cdot a_{ref}$ would compare the first row of matrix W^{ij} with our fixed reference, a_{ref}. The purpose of the reference vector is to allow us to compare the vectors of weights and allow us to identify them relative to the same reference. We represent each row using two scalars, $|\langle W^{ij}_{n:}, a_{ref}\rangle|$ and $\angle\left(W^{ij}_{n:}, a_{ref}\right)$ for $n = 1, 2, \ldots, M_i$. These are concatenated to give a vector representing a weight matrix between two layers, denoted by $\mathbf{v}_i \in \mathbb{R}^{2M_i}$. This is repeated for the next layer, giving $\mathbf{v}_j \in \mathbb{R}^{2M_j}$ and the vectors for each layer are then also concatenated together as a single vector, $\mathbf{v} \in \mathbb{R}^{2(M_i+M_j)}$. This vector $\mathbf{v}$ is then used as the representation of the state of the student. As described in Section 5.4.3, the performance of this network on a separate validation set as well as the training set is used as the reward which is used to update the teacher.

We define the student to be a fully-connected feedforward neural network with 2 hidden layers, with nodes $M_i = M_j = 50$. Each node in the hidden layer is activated by the ReLU function apart from the final layer where a softmax function is used to classify. The student is trained using stochastic gradient descent with a fixed learning rate of 0.001. We initialise a student and allow it to be trained using $N = 100$ separate batches sorted by the Mahalanobis distance. We then initialise another student and repeat the procedure. We repeat this for 10,000 students, therefore providing the teacher with 1,000,000 tuples in the experience replay buffer, R, (the collection of the agents' experiences) to sample and train from. At test time we initialise another student and allow the teacher to implement the learned policy.

5.5 Automated Curriculum Baselines

In this chapter we propose utilising reinforcement learning in order to train a teaching agent how to select a curriculum to maximise the performance of a classifier. As this is

a form of automated curriculum, we investigate other methods of automatic curricula in order to make a comparison. While the methods we have described in Chapter 4 outline manual methods of designing a curriculum (such as specifying the modality of the dataset and what metrics to use in order to quantify complexity of the data), here we look at various metrics that are a result of the training procedure and therefore not user-specified.

Loss-Based Curriculum (LBC)

The first of the metrics that we will consider is a loss-based curriculum. In this setup, we forward pass our training set through our model at iteration t. We then calculate the loss of each of our examples with respect to the ground truth and organise our batches according to this metric. For classification problems we use the crossentropy loss and for regression problems we use the mean squared error to calculate our loss. To summarise, in iteration t we forward pass our training set through our neural network f_t:

$$\mathbf{h} = f_t^{l-1}(\mathbf{x}) \tag{5.8}$$

where $l-1$ indicates we output the latents prior to the activation in the final layer of the network. We rank $\mathbf{h}$ according to lowest or highest value of loss (we term a reverse curriculum to go from the highest values of loss to the lowest). Once we have our ranking, we split our batches in the same manner we did in Chapter 4.

Gradient-Based Curriculum (GBC)

For this curriculum we rank the batches according to the magnitude of the gradients that are induced by each example in the training set. We calculate $\nabla_{\theta_{l-1}} \ell(\mathbf{y}, f_t(\mathbf{x}; \theta))$ where θ_{l-1} is the gradient with respect to the final weight layer in the network. We also inspect the gradients with respect to all layers and sum them. This method allows us to inspect which examples are causing large gradients in the final layers

during training and group these together. Once again, the loss for classification is crossentropy and for regression is mean-squared error.

Entropy-Based Curriculum (EBC)

In the entropy based curriculum scheme, we once again forward pass the entire training dataset and observe the outputs. We then calculate the entropy of the probabilistic output using $H(X) = \sum_i^n P(x_i) \log P(x_i)$. Those data points with a high entropy probabilistic output correspond to the data that the model is uncertain of classifying, and therefore distributes mass more equally amongst all classes.

Uncertainty-Based Curriculum (UBC)

For this form of curriculum we once again forward pass our training set through our network. However, we exploit the dropout that is included in the network architecture by varying the dropout with each pass. We repeat the pass n times with different dropout rates to gain n embeddings, $h_{1,\ldots,n}$, representing the same data point. We then calculate the variance across these embeddings. Our ranking is therefore calculated according to:

$$var(h) = \frac{1}{n} \sum_i^n \left(h_i - \frac{1}{n} \sum_i^n (h_i) \right) \tag{5.9}$$

The reasoning here is that where a robust representation of the data entry exists, the variance should be low with multiple passes with different stochastic dropout masks applied through the network. High variance indicates that the model is poorly conditioned for that data point and therefore that datum is badly represented by the network.

5.6 Datasets

We considered the same curated patient data collected in the IORD dataset. A training set of 60% of the dataset was again used and was balanced as before and the validation

Table 5.1: Average classification accuracies and standard deviations for various baseline and state-of-the-art methods on the Ward Admission (tabular), MIMIC-III (tabular) and CIFAR-10 (image) datasets. All models are averaged over the same five seeds apart from those highlighted with * which indicates that the accuracy reported from the cited text is quoted.

Method	Ward Admission Acc (SD)	MIMIC-III Acc (SD)	CIFAR-10 Sample 1 Acc (SD)	CIFAR-10 Sample 2 Acc (SD)	CIFAR-10 Sample 3 Acc (SD)
Batchwise	0.45 (0.01)	0.63 (0.01)	0.65 (0.02)	0.65 (0.02)	0.65 (0.02)
Curriculum	0.48 (0.02)	0.63 (0.02)	0.68 (0.01)	0.68 (0.01)	0.68 (0.01)
ST + curric	0.53 (0.03)	0.63 (0.02)	0.68 (0.02)	0.68 (0.02)	0.68 (0.02)
LBC	0.49 (0.01)	0.62 (0.02)	0.66 (0.01)	0.68 (0.04)	0.71 (0.05)
GBC	0.46 (0.02)	0.60 (0.02)	0.58 (0.03)	0.0.59 (0.04)	0.60 (0.02)
EBC	0.54 (0.02)	0.63 (0.01)	0.72 (0.04)	0.75 (0.02)	0.0.73 (0.01)
UBC	0.51 (0.01)	0.60 (0.02)	0.69 (0.03)	0.68 (0.01)	0.69 (0.02)
DeepFM	0.59 (0.01)	0.66 (0.01)	N/A	N/A	N/A
Deep+CrossNet	0.58 (0.02)	0.68 (0.02)	N/A	N/A	N/A
AutoInt	0.57 (0.02)	0.67 (0.01)	N/A	N/A	N/A
DenseNet*	N/A	N/A	0.96 (0.01)	0.96 (0.01)	0.96 (0.01)
GPipe*	N/A	N/A	**0.99** (0.01)	**0.99** (0.01)	**0.99** (0.01)
DQN ST	0.58 (0.01)	0.66 (0.02)	0.86 (0.04)	0.88 (0.02)	0.86 (0.03)
DDPG ST	**0.62** (0.02)	**0.70** (0.01)	0.90 (0.01)	0.90 (0.01)	0.89 (0.01)

set was 20% of the dataset, testing was also 20%, and the classes were kept in the same distribution as the original dataset. We carry out the same tests on the MIMIC-III dataset with the same train-validation-test splits. To further validate our methodology we also report results on the CIFAR-10 image recognition dataset [44]. Due to our model having the capability of selecting large batches and memory constraints, we take a random, stratified sample from the training set of 12,000 images of which 1,200 are made into a validation set. We then take a random, stratified sample of 3000 images from the test set as our held-out test set. We repeat this three times essentially creating three separate datasets which we identify as sample 1, sample 2 and sample 3 respectively in Table 5.1.

5.7 Results

To make a comparison between our student-teacher network and other state-of-the-art methods for classification, we show normal stochastic mini-batch training (batchwise),

vanilla curriculum training (curriculum) and the method proposed by [128] (ST + curric). We report the (ST + curric) result for the student-teacher method that worked best for each dataset out of the four approaches discussed in Section 5.2. The performances for both methods of batch formation (as discussed in Section 5.4.1) were compared. It was found that for direct curriculum training (curriculum), method 2 (using batches that are cumulative supersets of each other) was best, whereas for the student-teacher methods (ST + curric, DQN student-teacher, DDPG student-teacher), method 1 (the intersection of the batches results in the null space) worked best. We also see that for the automated curriculum baselines, the best performing of these is the entropy-based curriculum. However, these baselines still perform much worse than the DDPG student-teacher method.

For the Ward Admission and MIMIC-III datasets we also compare to state-of-the-art methods for classifying tabular data, namely DeepFM [119], Deep+CrossNet [138] and AutoInt [185]. For the CIFAR-10 dataset we compare with models that achieve state-of-the-art performance on image datasets, namely DenseNet [123] and GPipe [169]. For our non-medical dataset, CIFAR-10, we sort data using cosine similarity as entropy. We must use a different student for this dataset and so we use a student that has 4 convolutional layers with 32 filters of size 3x3 in the first two layers and 64 of these in the second two. These are all activated by ReLU and maxpooled and are followed by 3 feedforward layers of size 50 nodes each. We use the encoding of the weights of the feedforward component of the network as the state of the student. From Table 5.1, we see that our method (DDPG trained student-teacher network) outperforms at least 3 state-of-the-art methods for classifying tabular data. It is also capable of providing a competitive performance on image classification problems, approaching state-of-the-art performance. This demonstrates the robustness of this method as a training methodology for various datasets. In Section 5.8, we go on to investigate the policies that have been generated for training and discuss them in the context of the curriculum.

5.8 Investigation of Policies Learned

Learned Curricula

In this section we investigate the policies learned by the teachers through their experience of training many students. We implement the student in such a way that the configuration of the student at which the highest accuracy is achieved is saved.

Ward Admission

From Figure 5.2 we see that the student accuracy curve does not follow the usual smooth training arc seen during stochastic mini-batch training. We can see areas where the teacher aids the student in escaping local minima. The teacher swinging from training the student with low entropy data to high entropy data just before iteration 4000 in Figure 5.3 corresponds to a degradation in performance of the student in Figure 5.2. This may indicate that the teacher has used the high entropy data to push the student out of a local minimum (using the gradients from the selected batch), hence the performance degrades. This turns out to be an unsuccessful attempt, as with further training the model does not perform any better than it had done previously. However, just prior to iteration 7000, the teacher once again performs a 'high entropy spike' (shown by the orange crosses rising). This again deteriorates the performance by escaping from a local minimum, but with further training the student achieves a performance of 62% (indicated by the red dotted line in Figure 5.2), the highest it has achieved so far. Here we present further examples of the curricula that were learned by the teacher for the four datasets we have used. We show that the policies learned are consistent according to the dataset and reflect a strategy that has been learned by the teacher.

We show another example of training by spiking in entropy to escape local minima in Figures 5.4 and 5.5. Once again there is a spike in entropy of data selected for

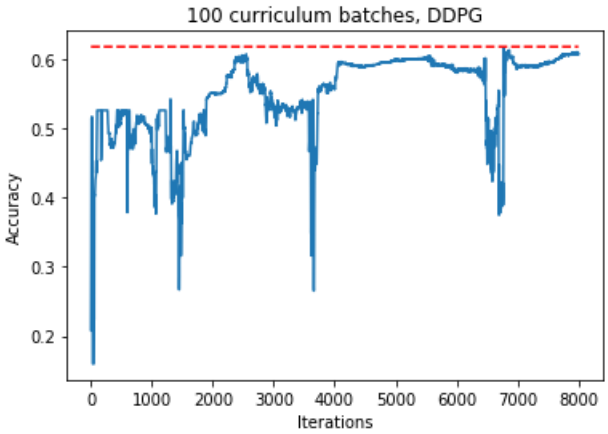


Figure 5.2: The performance of the student on the held-out test set of the Ward Admission prediction while it is trained by the teacher. The red dashed line is the best performance achieved by this student. Notice that it does not follow the typical smooth learning arc usually seen.

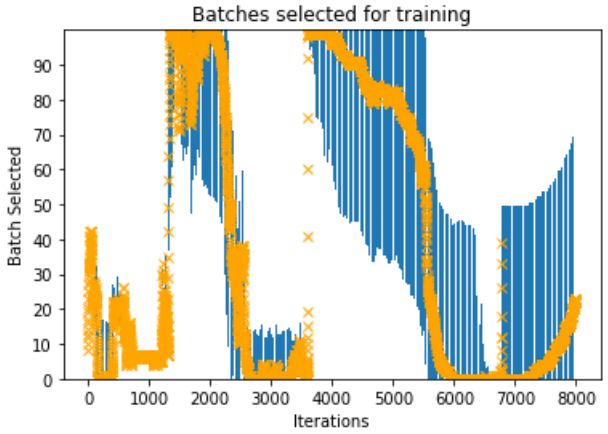


Figure 5.3: Actions generated by the policy of the teacher that has led to the performance of the student shown in Figure 5.2. Orange crosses are the first output (where to select data from) and blue bars are the second output (how much data around the central selection point to include in the batch for training). If the batch selected is near zero, this is low entropy data and if it is near the top of the batch selection, this is high entropy data.

training prior to 6000 iterations, which allows us to escape a local minimum and degrade the performance but upon further training achieve a better accuracy on the held-out test set. Further experiments show that this entropy spiking strategy is the preferred strategy for the ward admission dataset.

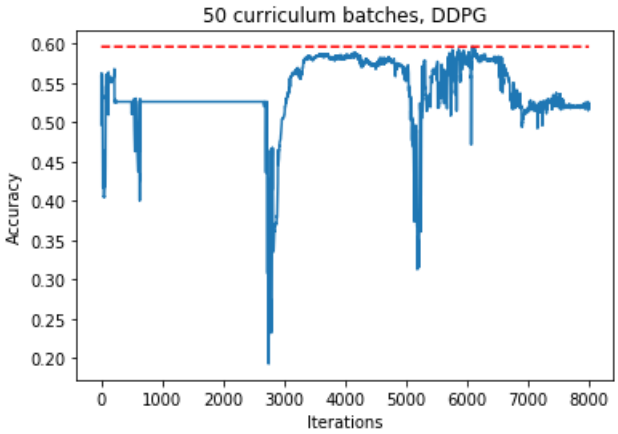


Figure 5.4: The performance of the student on the held-out test of the ward admission dataset while it is trained by the teacher. The red dashed line is the best performance achieved by this student.

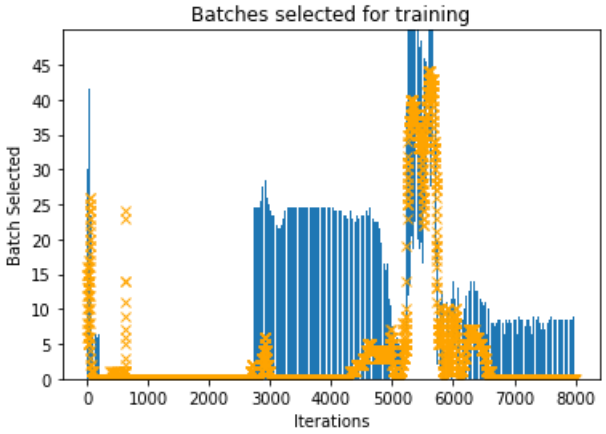


Figure 5.5: The actions generated by the policy of the teacher that has led to the performance of the student shown in Figure 5.4. Orange crosses are the first output (where to select data from) and blue bars are the second output (how much data around the central selection point to include in the batch for training). If the batch selected is near zero, this is low entropy data and if it is near the top of the batch selection, this is high entropy data.

MIMIC-III

Plotted below are various examples of the curricula that were developed to train students on the MIMIC-III prediction problem. All of these provided state-of-the-art performance on the prediction problem as shown in Table 5.1.

In Figures 5.6 and 5.7 we see that the teacher utilises very small data batches to train. This generally gives rise to very noisy training gradients which it seems the

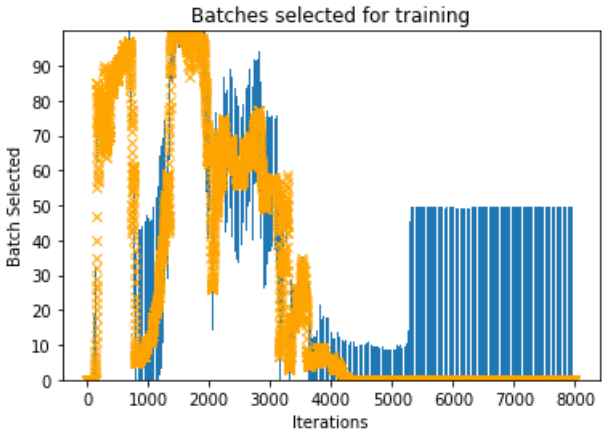


Figure 5.6: Curriculum generated for a randomly initialised student trained on the MIMIC-III dataset.

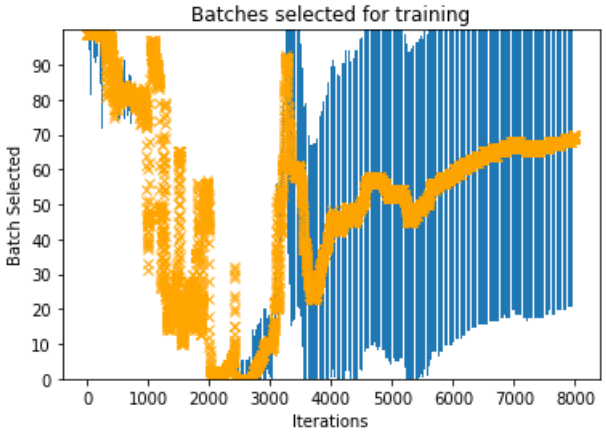


Figure 5.7: Curriculum generated for a randomly initialised student trained on the MIMIC-III dataset.

Figure 5.8: Curriculum generated for a randomly initialised student trained on the MIMIC-III dataset.

teacher uses to converge to a favourable 'initialisation', from which it then starts to train on bigger batches. In Figure 5.8 we see that the teacher seems to bring the student into a 'good initialisation' early and so the rest of training is on the bigger batches.

CIFAR-10

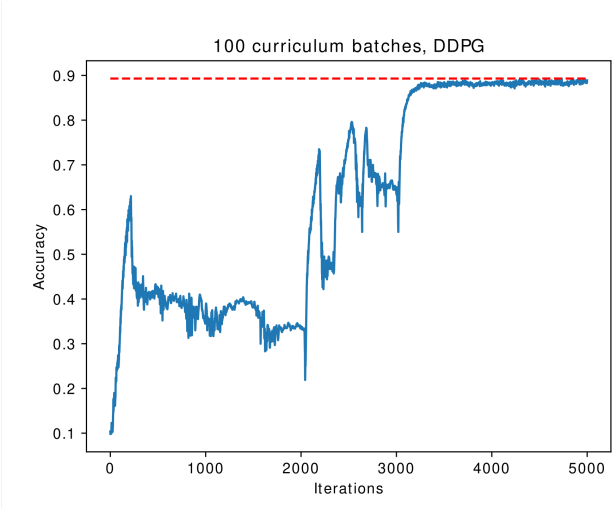


Figure 5.9: The performance of the student on the held-out test of the CIFAR-10 dataset while it is trained by the teacher. The red dashed line is the best performance achieved by this student.

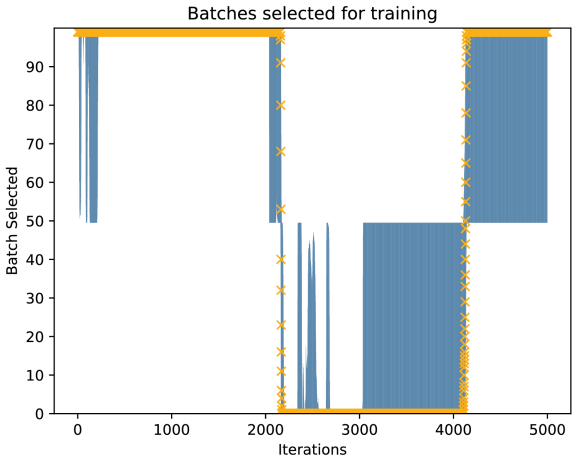


Figure 5.10: The actions generated by the policy of the teacher that has led to the performance of the student shown in Figure 5.9.

The performance of a student and the curriculum learned for training this student

on the CIFAR-10 dataset are shown in Figures 5.9 and 5.10. We see the teacher primes the student into an initial state before (at approximately iteration 3000) repeatedly presenting low entropy batches before progressing to high entropy batches. This is very similar to curricula that are commonly used in many studies on image recognition. Figures 5.11 and 5.12 show the curricula used for other students by the same teacher. It would seem that repeated presentation of low entropy batches before progressing to repeatedly presenting high entropy batches is most beneficial for training the image recognition students. This makes sense due to the need for feature extraction in order to generalise to other images. Training from a low entropy to a high entropy is also reported as improving the performance of convolutional neural networks in [43], [168].

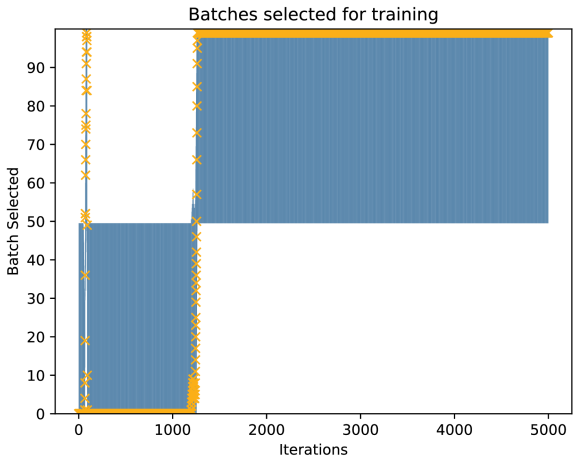


Figure 5.11: Curriculum generated for a randomly initialised student trained on the CIFAR-10 dataset.

Constrained Policy Learning

To demonstrate that the curriculum is indeed guided by the teacher and not simply providing an alternative optimisation trajectory, we train two identical teachers, one with no constraints and one that has been constrained in some way, and we compare the policies learned. We use a DDPG teacher where the constrained teacher uses 1

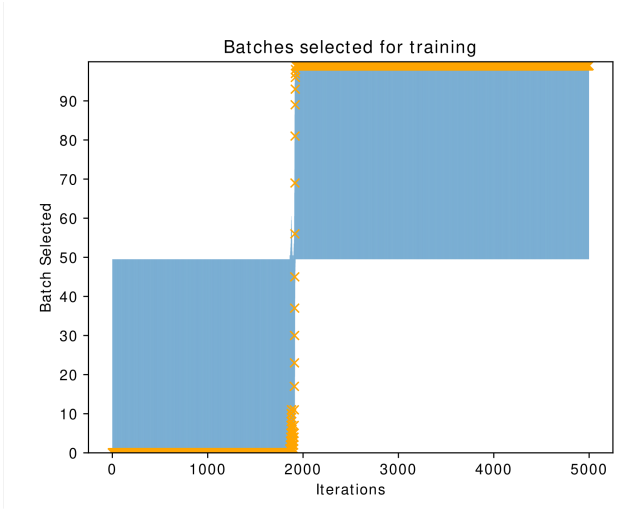


Figure 5.12: Curriculum generated for a randomly initialised student trained on the CIFAR-10 dataset. 5.4.

datapoint at a time with probability 0.999 (i.e., the teacher's second output (batch width) is almost always 0) and uses the real teacher output otherwise. The students of these teachers are trained on the CIFAR-10 dataset. From Figure 5.13, we see a

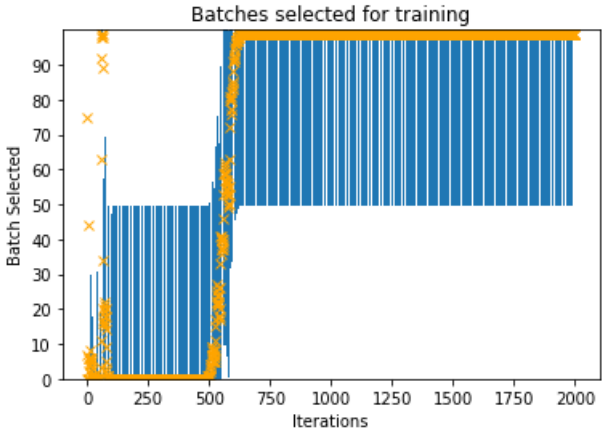


Figure 5.13: Actions generated by the policy of an unconstrained DDPG teacher on the CIFAR-10 dataset. Orange is the first output (where to select data from) and blue is the second output (how much data to include in the batch). If the batch selected is near zero then this is low entropy data and if it is near the top of the batch selection then this is high entropy data.

fairly typical curriculum with a relatively noisy starting phase going from low to high entropy then rapidly back to low. However the general trend is that low entropy data is used and trained with before progressing onto the higher entropy data. In Figure

Figure 5.14: Actions generated by the policy of a constrained DDPG teacher on the CIFAR-10 dataset where it is highly probable that only one datapoint will be selected at a time for training. We see that the selection is significantly more chaotic. Batch numbers near zero are low entropy data and batch number 100 contains the highest entropy data.

5.14 we see that we begin in a similar way as in Figure 5.13 (rapidly going from low to high entropy) but after this the smooth transition from low entropy to high entropy is not observed. It would seem that due to individual datapoints providing much noisier gradients than batch updates, the teacher must oscillate rapidly between high and low entropy data in an attempt to provide effective training. It is interesting to observe that the teacher in both cases samples along the entire entropy axis before converging to high entropy.

Another way of constraining our teachers is to reduce the learning rate of our student models. This will allow us to check whether the teacher will output the same strategies for students learning at different rates. We present the findings on the MIMIC-III and ORCHiD datasets below.

MIMIC-III

To constrain our students we first constrain our teacher (as done in the main paper) to select a batch width of zero with probability 0.999. Figure 5.15 shows the policy of the teacher when training the student on MIMIC-III data. When comparing these to typical MIMIC-III generated curricula (Figures 5.6, 5.7, 5.8), we see that there

Figure 5.15: The actions generated by the learned policy of a constrained teacher to train a student on the MIMIC-III dataset. The student has a learning rate of 0.02.

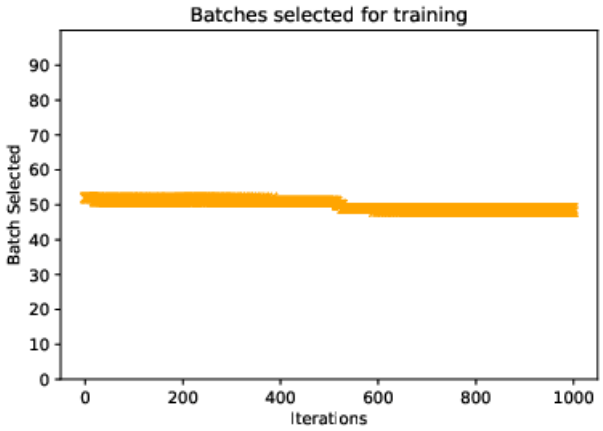


Figure 5.16: The actions generated by the policy of a constrained teacher to train a student that is also constrained with a lower learning rate of 0.002. The student has the same initial seed as that trained using the policy shown in Figure 5.15.

is no oscillation in entropy at the early stages of training and instead the teacher has learned to simply gradually step down in entropy. The student is trained with a learning rate of 0.02 and so in order to constrain this further we also reduce the students learning rate to 0.002, now constraining the student. We see from Figure 5.16 that the teacher begins training the student using similar data (at approximately batch 50 on the entropy scale), however due to the student's lower learning rate the downward stepping takes significantly longer. This is highly encouraging as it shows that the teacher is following the same strategy as used in Figure 5.15 albeit over a

longer number of iterations as we would expect.

IORD

In Figures 5.17 and 5.18 we utilise a DQN trained teacher on the IORD dataset. We initially train normally and then slow the learning rate of the student by 100 times for the same initial seed to see how this alters training.

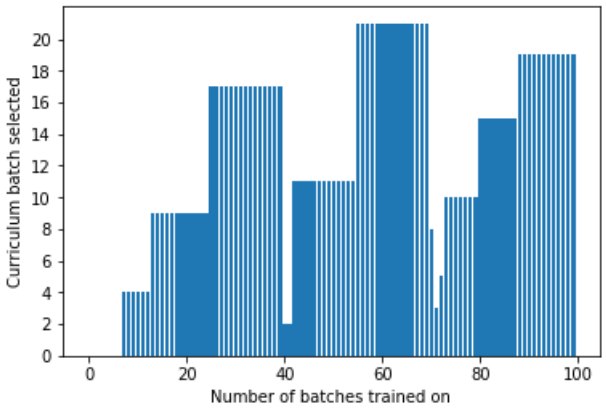


Figure 5.17: Actions generated by a DQN teacher with a learning rate of 0.01. At each iteration, anything shaded in blue is included in the batch used for training.

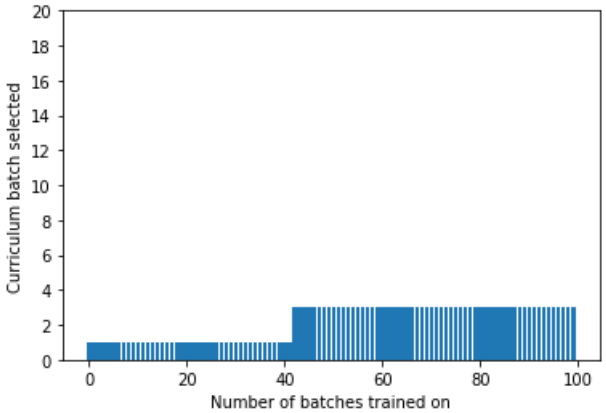


Figure 5.18: Actions generated by a DQN teacher with a learning rate of 0.001.

We can see in Figure 5.17 that a 'recurring low to high entropy' curriculum is implemented by the teacher as seen implemented by the DDPG teacher. These can be seen as the DQN equivalent of the high entropy spiking strategy found by the DDPG teacher. Where we see drops in the entropy of data being used seem to be locations

where the teacher is attempting to escape local minima. In Figure 5.18 we reduce the student's learning rate and we see that we still have a 'low to high entropy' curriculum but it is progressing much more slowly. Once again, this is due to the step size being smaller and therefore requiring more gradient updates to get the student network into a weight state that requires different batches for training.

Policy Stability for Similar Students

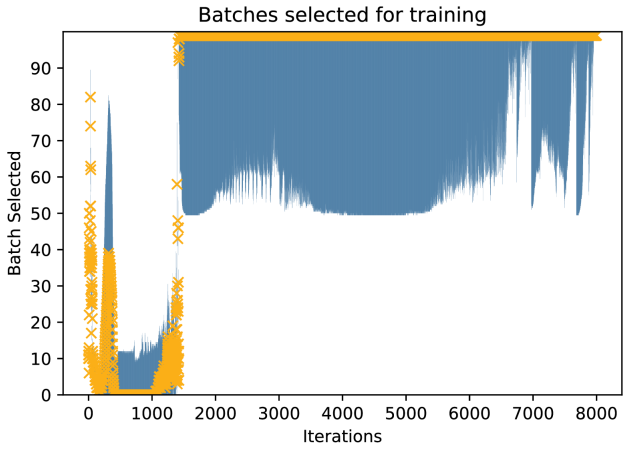


Figure 5.19: Actions selected by the policy of the teacher for a student to train on the Ward Admission dataset.

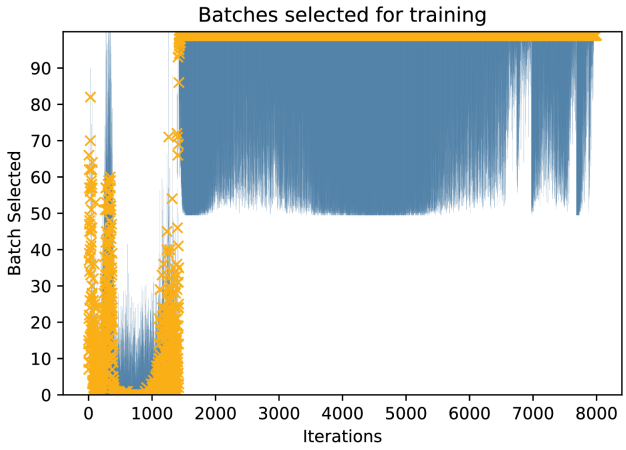


Figure 5.20: How the actions of the teacher shown in Figure 5.19 have changed by applying Gaussian noise to the states of the student.

To investigate the stability of the policies learned by the teacher we save the state

of the student at every iteration and corrupt this signal with zero-mean Gaussian noise with standard deviation 0.1. We see from Figures 5.19 and 5.20 that whilst the policies have changed slightly, the overall strategy of the policy is the same. This is encouraging as this is indicative of learned behaviour conditioned on the state as opposed to providing an alternative form of stochastic training. We now further demonstrate in this section that for all the tasks considered our teacher learned stable policies conditioned on the current state of the student. We present our findings on the ward admission location prediction from the IORD dataset, and validate our findings on the MIMIC-III and CIFAR-10 datasets as outlined below.

MIMIC-III

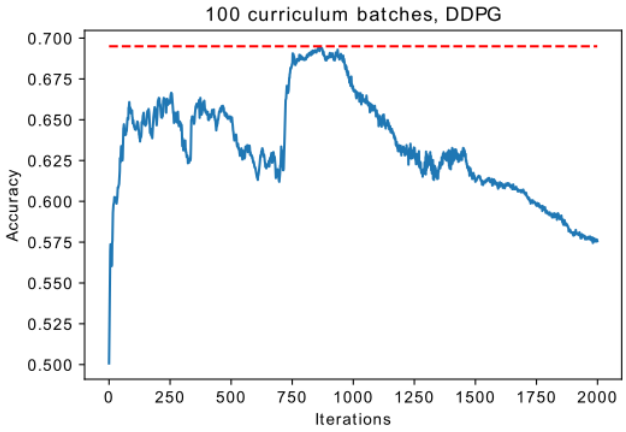


Figure 5.21: The performance of a student trained by the DDPG teacher on the MIMIC-III dataset.

We see that once again the teacher learns a policy of using low entropy data to initialise the student before increasing the size of the batch introduced to maximise performance. We now once again apply Gaussian noise to the states of the student.

In Figure 5.22 we see that the overall structure of the curriculum is the same as other MIMIC-III policies generated, beginning at high entropy and reducing to low to initialise the student before expanding the size of the batch. Figure 5.23 also shows this with a very similar curriculum to the one in Figure 5.22 being followed. This

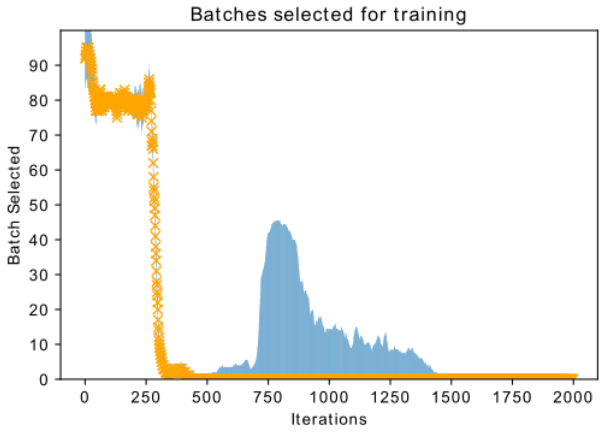


Figure 5.22: The actions used by the teacher to train the student with performance shown in Figure 5.21.

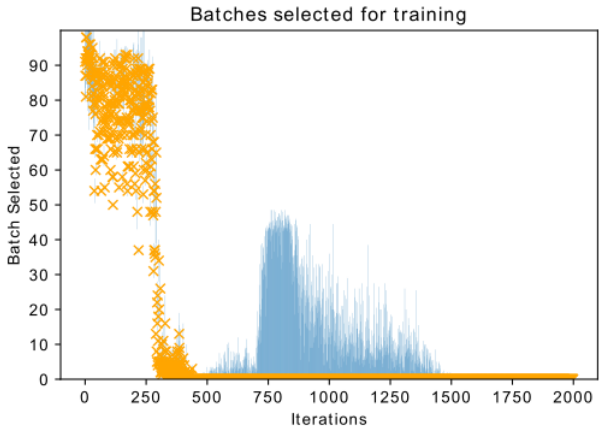


Figure 5.23: The actions taken by the teacher when the student has Gaussian noise applied to its states.

further encourages us that a strategy has indeed been learned by the teacher to train a student on the MIMIC-III dataset based on the weights of the student.

CIFAR-10

Once again we repeat the exercise on the CIFAR-10 dataset and observe the stability of the teaching policy based on the corrupted states of the student.

We see in Figures 5.24 and 5.25 that the same general policy is followed as that used in Figures 5.10, 5.11 and 5.12. As we only train for 500 iterations the policy ends at the point of transition to training on high entropy data. We see that corrupting

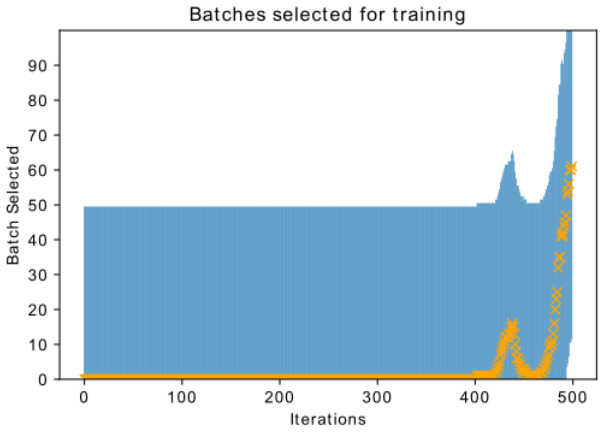


Figure 5.24: The actions used by the DDPG teacher to train a student on the CIFAR-10 dataset.

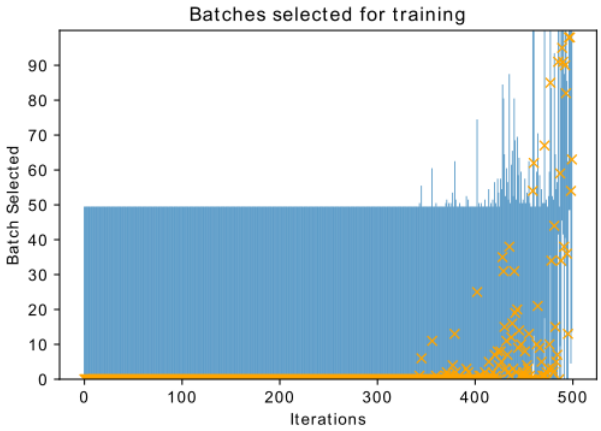


Figure 5.25: The actions of the teacher when the student has Gaussian noise applied to its states.

the students states with Gaussian noise has not significantly changed the policy of the teacher, providing further reassurance that the policy is not only stable but a learned function of the state of the student and not simply an alternative optimisation trajectory.

Policy Transfer between Tasks

We also investigated how transferable the trained teachers are between tasks. A teacher was trained on the Ward Admission dataset and transferred to the MIMIC-III prediction problem with no fine-tuning. The policy the teacher generated on the 'as

before unseen' training set is shown in Figure 5.27 and the corresponding result is shown in Figure 5.26.

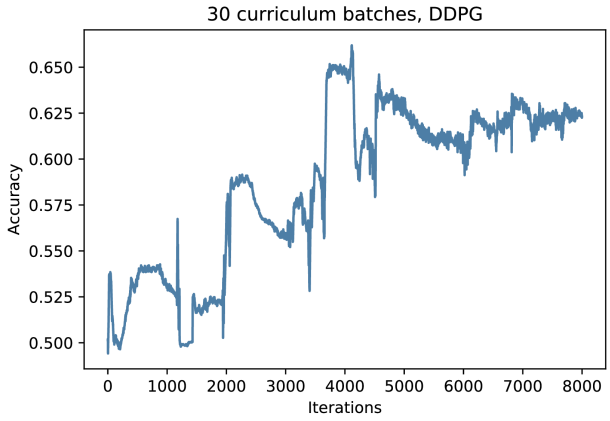


Figure 5.26: Performance of student on MIMIC-III dataset when trained by a transferred teacher taught to train students on the Ward Admission dataset.

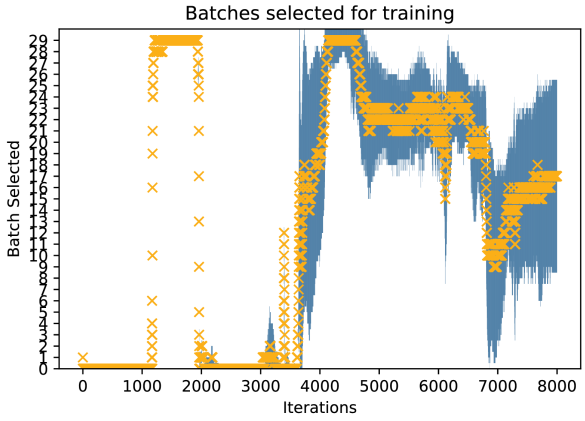


Figure 5.27: Actions selected by the transferred policy for the randomly initialised student which yielded the performance shown in Figure 5.26.

Here we see that the teacher once again uses the 'entropy spiking' technique to escape possible local minima resulting in the waves of improvement in accuracy shown in Figure 5.26, characteristic of curriculum learning. The performance attained is less than it would be when training the teacher directly using the MIMIC-III dataset (averaging 67% ± 1% versus 70% ± 1%, however it still outperforms the baseline methods and performs equivalently to some of the state-of-the-art methods (shown in Table 5.1). It is not currently clear why the transfer of teachers seems to be successful

but this is being investigated for various datasets.

Entropy Spiking Sampling

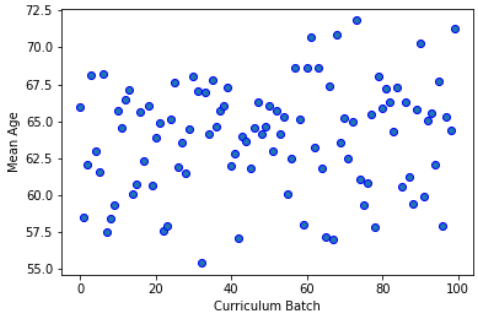
As seen in Figures 5.3 and 5.5, when teaching a student on ward admission location prediction, the teacher learns a policy of 'spiking entropy'. It is seen that the teacher uses large swings upwards (on the entropy and batch number axis) of batches selected while keeping their batch-width small. Utilising small batches induces a higher variance on the gradients when training. This may be indicative that the teacher has learned to use high-variance (in terms of gradient) batches in order to encourage movement of the student out of local minima (in the error versus parameter space) which corresponds to degradation in performance of the student. Upon degradation, the teacher then selects from lower entropy samples and widens the batch-width. This can be seen as a re-initialisation of the student, and in increasing the width of the batches used for training, providing a more stable training trajectory.

5.9 Analysis of Selected Patients

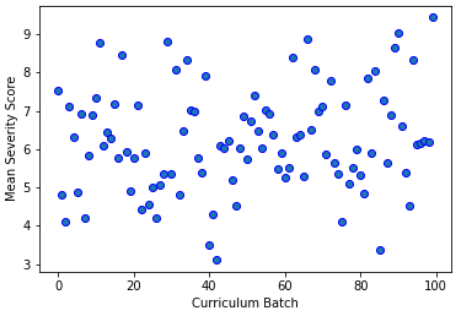
Having seen that the teacher develops strategies when trained using reinforcement learning, we now aim to gain some insights from these strategies. The actions of the teacher can be seen as a form of patient sampling to maximise the performance of the predictor. By gaining insights as to how the sampling corresponds to patients in the hospital, we can use the training strategy to guide clinical staff on how to prioritise data collection and from which patients. Whereas our approach in Chapter 4 focused on guidance as to which features to prioritise, in this chapter we aim to provide guidance on which patients to prioritise data collection from.

To do this, we first analyse how patient features vary along the Mahalanobis index.

In Figures 5.28(a) and 5.28(b) we see how a Mahalanobis curriculum of 100 batches distributes the mean ages and severity scores of the patients in our training set. The



((a)) Mean age across batches



((b)) Mean severity score across batches

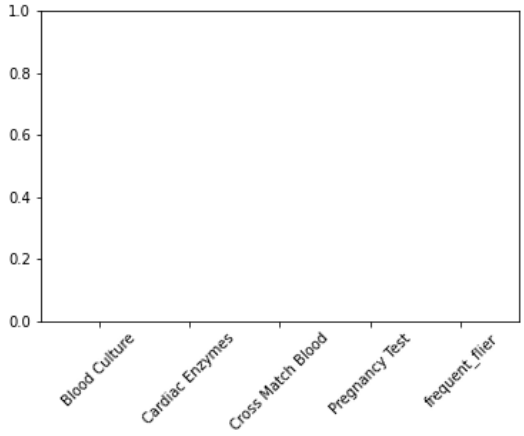
Figure 5.28: Plots to show how the mean age and severity score of the patients are distributed according to Mahalanobis curriculum batching.

severity score used here is different to the one used in the ED and is calculated post-admission meaning it could not be used as a feature in our training dataset. We see that there is very little correlation between position in the Mahalanobis curriculum and age or severity. As a result, we can exclude the possibility of the teacher selecting patients according to these criteria (i.e., ordering data according to age or severity of the patient does not provide us with a stronger performance during training).

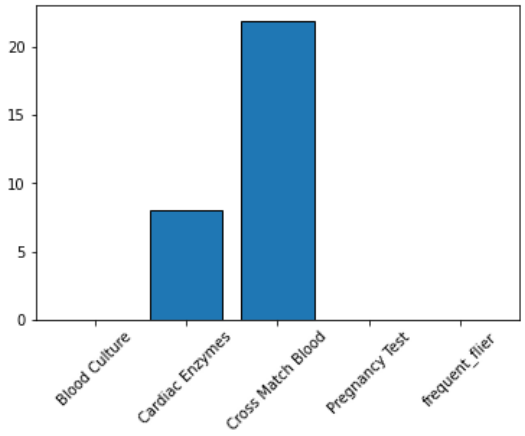
Analysis of Selected Features

As discussed in Chapter 3, it was found that the most useful features for this prediction problem were whether or not tests were requested for a blood culture, for cardiac enzymes, if a blood cross-matching was requested, if a pregnancy test was requested, and if the patient was a frequent admission. The previous diagnosis that the patient was admitted with was also an important consideration. We now assess how the teacher uses the Mahalanobis curriculum to select according to these features.

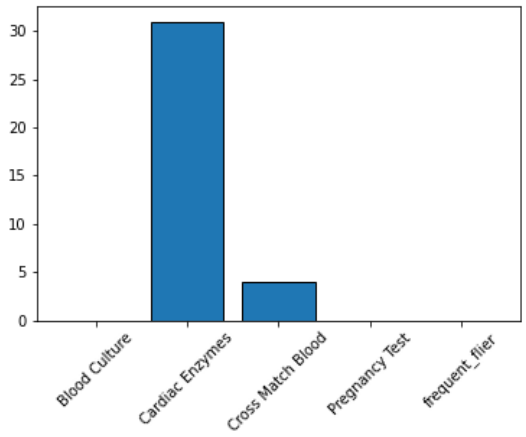
We see in Figure 5.29(a) that the patients in this category all have had no tests requested during triage in the ED. This signifies that the predictions for these patients are most likely being driven by their previous diagnosis code, based on the historic diagnosis code being chosen as one of the most important features for prediction previously. Note that we still use the full feature set as described in Chapter 4. We



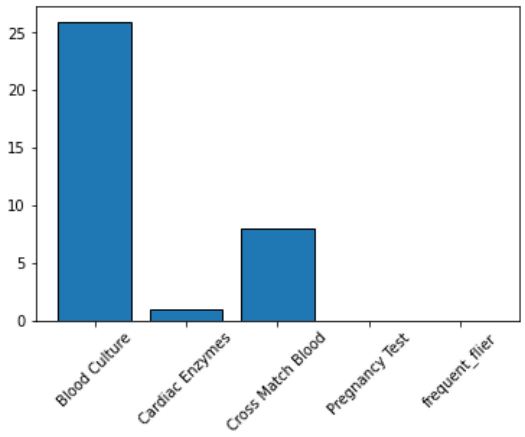
((a)) Test counts in the first (lowest entropy) batch



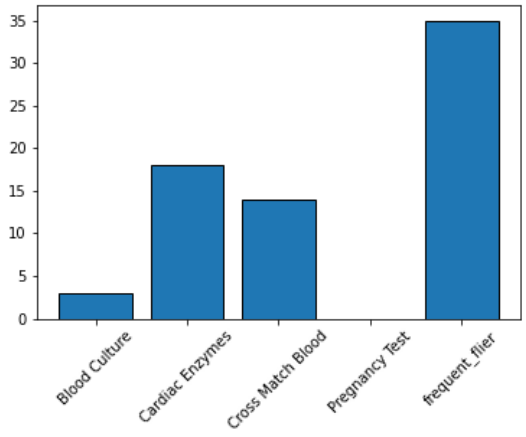
((b)) Test counts in the 32$^{\text{nd}}$ batch



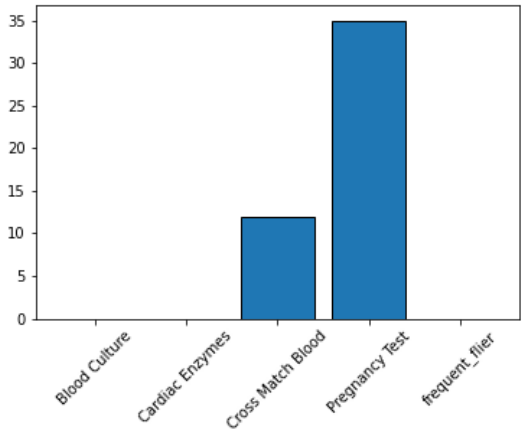
((c)) Test counts in the 55$^{\text{th}}$ batch



((d)) Test counts in the 70$^{\text{th}}$ batch batches



((e)) Test counts in the 80$^{\text{th}}$ batch



((f)) Test counts in the 90$^{\text{th}}$ batch batches

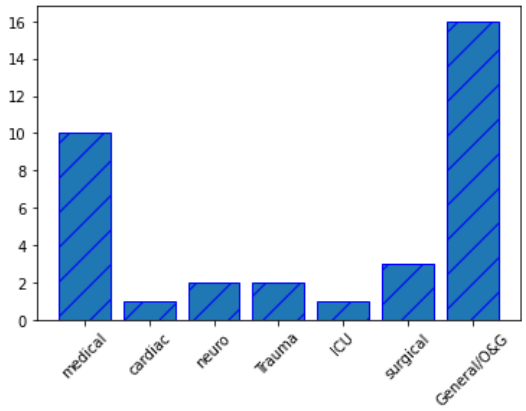
Figure 5.29: Plots to show how the features selected by our saliency method are distributed along our Mahalanobis index.

simply hypothesise based on the feature selection that the diagnosis code is mainly responsible for the classifications. This in addition to there being no tests requested renders these patients those that are more difficult to classify than those with requested tests. As a result, a low entropy start to training implies training with "more difficult" data, as hypothesised.

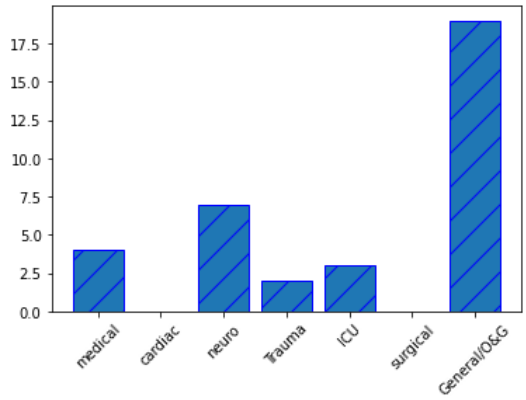
Figure 5.30 shows the corresponding labels that are included in each of the batches selected in Figure 5.29. We see from Figures 5.29(a) and 5.30(a) that a previous diagnosis code and no other tests is prevalent amongst all classes. We further see that the prevalence of blood type cross-matching tests is correlated with increased neuro, trauma and ICU admissions. The cardiac enzymes are highly correlated with cardiac admissions as was seen in Chapter 4. Blood culture tests are correlated with medical admissions as well as neuro and ICU. These visualisations further support the saliency method introduced in Chapter 4 and provide us with reassurance that the selected features correlate with the labels in this way.

Having analysed how the features and labels are distributed along the Mahalanobis curriculum, we now investigate how the teacher's learned policy exploits these patterns. In Figure 5.31, we show policy as generated in Figure 5.3 with red dots over the actions of the teacher that we sample to investigate the feature and label distributions.

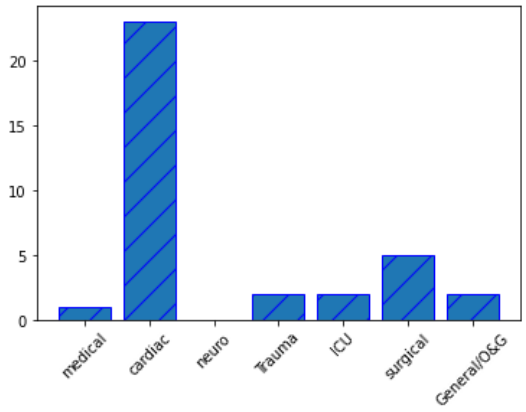
It can be seen in Figure 5.32 that the teacher's low entropy start means that the historic diagnosis code plays a greater role in training as can be seen by the patients with no tests requested. We also see that this corresponds to a greater prevalence of patients admitted to the surgical wards as well as those admitted to the neuro wards (Figures 5.32(b) and 5.32(d)). This supports the findings in Chapter 4 that admission to the neuro wards has a strong correlation with a previous diagnosis code. At the third sample the teacher selects a batch that contains a much broader range of classes (Figure 5.32(f)). This is associated with the converging of the performance of the teacher. The teacher selecting larger batch sizes at higher entropies may be due to the need to provide a more balanced batch in terms of class representation and therefore



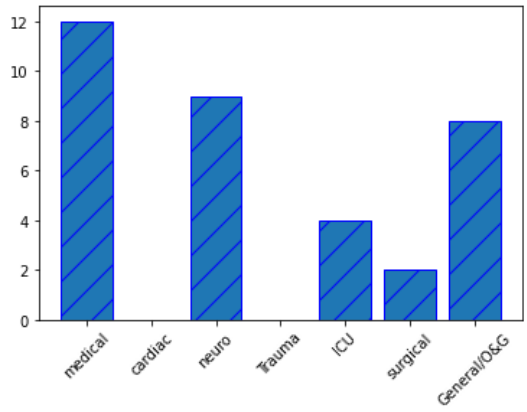
((a)) Label counts in the first (lowest entropy) batch



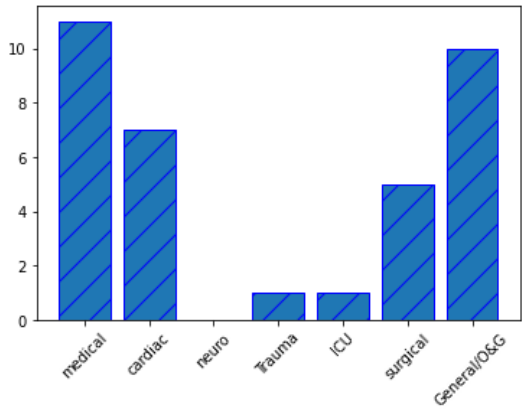
((b)) Label counts in the 32$^{\text{nd}}$ batch



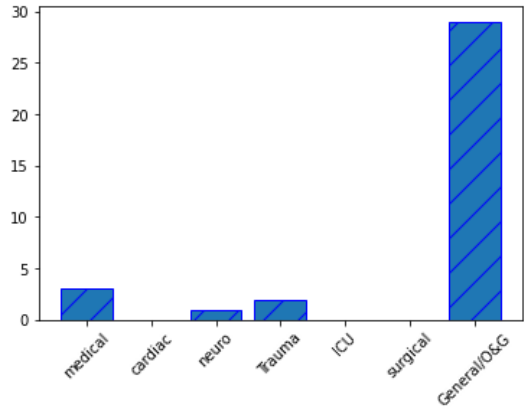
((c)) Label counts in the 55$^{\text{th}}$ batch



((d)) Label counts in the 70$^{\text{th}}$ batch batches



((e)) Label counts in the 80$^{\text{th}}$ batch



((f)) Label counts in the 90$^{\text{th}}$ batch batches

Figure 5.30: Plots to show how the labels are distributed along our Mahalanobis index.

stabilise training.

The fourth sample is during the 'swing' downwards in entropy, where the per-

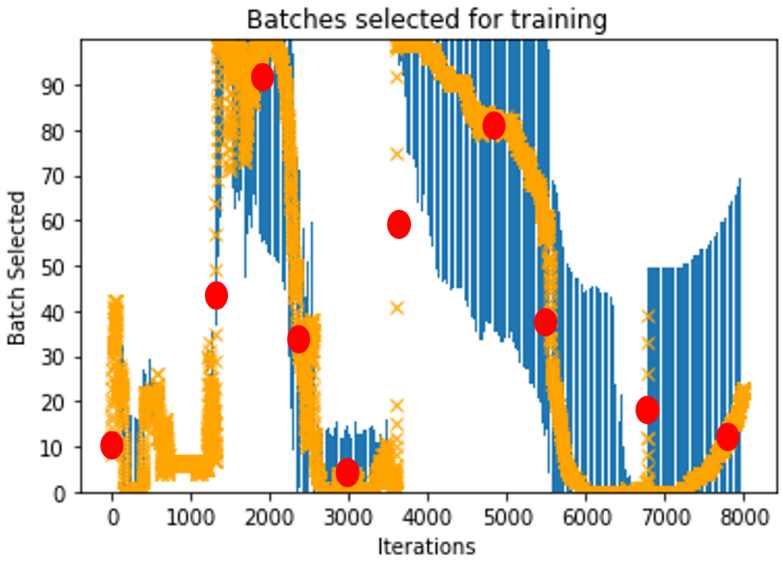


Figure 5.31: The actions generated by the teacher's policy in Figure 5.3 and the locations of our samples.

formance of the student increases. This is associated with once again a relatively balanced batch in terms of class representation, however this batch is more reliant on the historic diagnosis code for separation as opposed to the tests requested (Figures 5.32(g) and 5.32(h)).

The fifth sample represent the beginning stage of the first entropy spike. In this batch, the general class is still prevalent and there is still cross-class representation, however again with the diagnosis code being more important for separation (Figures 5.33(a) and 5.33(b)). The sixth sample occurs during the entropy spike. Here we see that the balance of classes has changed from previous training rounds. Whereas previously the general/O&G class provided a significant contribution to the composition of the batches, here the contribution is significantly reduced. Instead, the cardiac ward is dominant in this batch (Figure 5.33(d)). It is possible that due to the large reduction in the general ward class in this batch, this changes the target distribution presented to the model significantly and causes the large drop in performance that is associated with the entropy spike. As a result, when the general ward contribution is reinstated in the next two samples (Figures 5.33(f) and 5.33(h)), the performance

stabilises and improves once again.

The ninth sample is once again an entropy spiking sample. Figure 5.34(b) shows that once again the teacher has chosen a batch that has significantly changed the target distribution by reducing the contribution of general ward admission samples to the batch. This in turn causes a drop in the performance of the student. It is possible that in choosing smaller batches, the likelihood of selecting a target distribution that is sufficiently different to the overall task target distribution is greater. As a result, small batch sizes are generally used by the teacher in order to carry out the entropy spike. The final sample once again incorporates a good balance of classes and the performance once again increases. It is likely that the teacher selections according to class are not balanced due to the teacher reward being based on the validation set, which like the test set and unlike the training set, was not balanced according to class.

Clinical Utility

The results of this chapter may be employed in numerous ways in order to aid clinicians in their day-to-day work. The first use is in the model that has been developed. By presenting a student-teacher method that learns from the weights of the student, a clinician need only provide a dataset and define the input feature size and output size to the student before training the model. This should then provide the clinician with a well-performing model regardless of the domain they choose to train it in. This should make the use of machine learning and model tuning more approachable to clinicians who wish to do so.

The clinical utility of this work specifically for the problem at hand is in the prioritisation of collection of patient data. It was demonstrated that the teacher uses "low entropy" data initially before going to higher entropy data. The teacher also employs entropy 'spiking' where data that look out of distribution are used as the target in order to navigate out of local minima. Therefore, clinicians can use the Mahalanobis index in order to rank incoming patients. By seeing where they lie on the

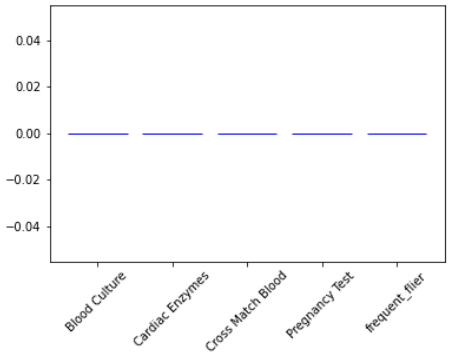
scale, they can prioritise whether the patient data will be useful for normal training or entropy spiking. A heuristic method of training and then spiking may also be used if data is collected specifically for this purpose in order to train a model in an cyclical way (much like the teacher). This approach can be mimicked in any of the domains the clinicians wish to apply a deep learning model. They would simply need to rank data according to the Mahalanobis (or any entropy metric) index and then prioritise collection of data that would allow them to carry out the entropy spiking strategy demonstrated by the teacher.

5.10 Conclusion and Limitations

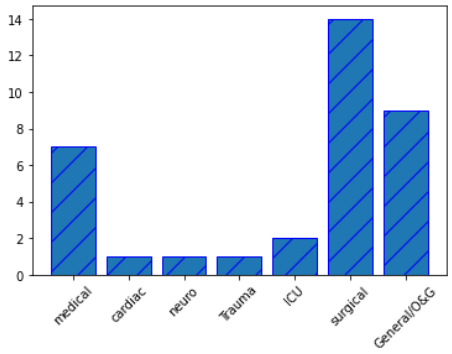
In this chapter we have further developed the method of predicting which type of hospital ward a patient will be admitted to after being triaged in the ED. We have done this by combining a teacher network that is trained using reinforcement learning based on its experience in training many student networks on the same task. We have shown that the 'teaching' neural network can learn from an encoding of the state of another network (in this case the student). In this way, an appropriate curriculum can be generated by the former in order to maximise the task performance of the latter. We have seen that the teacher develops strategies of moving out of local minima by sampling batches of data that show different output distributions to that of the overall training set. This is achieved by explicitly conditioning on the current state of the student, with policies learned that could not be using retrospective information on the student state. In doing so the teacher is able to degrade the performance (moving the weights of the student out of a local minimum) before training with samples that are more representative of the training set output distribution. It is hoped that the insights gained from using this approach will assist curriculum design as well as provide insight into how learning occurs in neural networks.

One of the limitations we face with this approach is that the representation of the

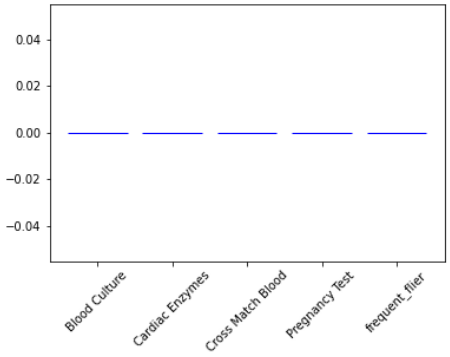
weights of the student is a form of dimensional reduction, and therefore may not be the best representation of the weights of the network from which to learn. To improve upon this, we propose that future works introduce an attention mechanism to identify which nodes in the network are predominantly responsible for the prediction, and use these weights as inputs to the teacher to remove redundancies from the input. There is also the problem that the number of inputs to the teacher relies on the size of the student where a student with M nodes in L hidden layers results in an input size of order M for the teacher. While this is better than explicit representation of the weights, which would result in an input size of order M^2, we actively seek better representations of the student's state. One potential method for achieving this is through the use of distillation networks, as seen in [92], allowing for a more compact representation of the student network and using the student-teacher methodology proposed in this work. We are encouraged however by the performance of the student-teacher setup on relatively small student networks achieving state-of-the-art performances. We also encourage experimentation with various curricula to rank data according to the dataset being considered. This changes the way that data are ranked, and therefore provides different combinations of data that can be grouped into batches. Alternatively, we could investigate having different outputs for the teacher networks and investigate methods of combining these outputs to select data for training.



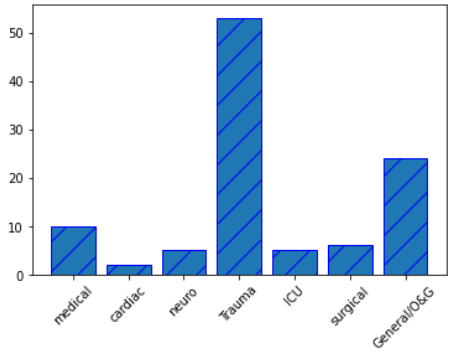
((a)) First sample features



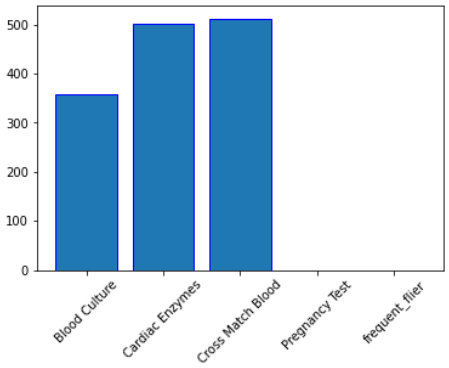
((b)) First sample labels



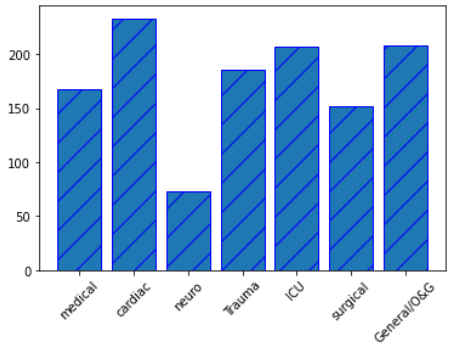
((c)) Second sample features



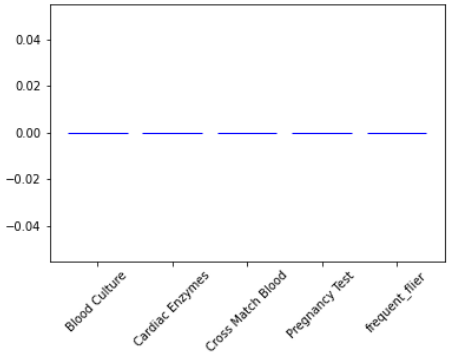
((d)) Second sample labels



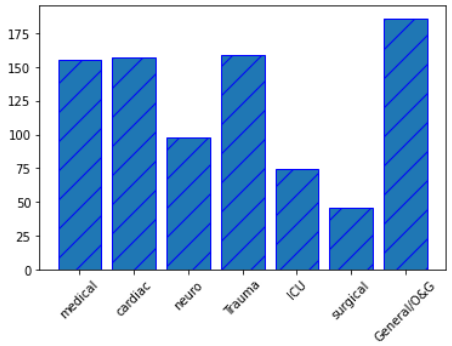
((e)) Third sample features



((f)) Third sample labels

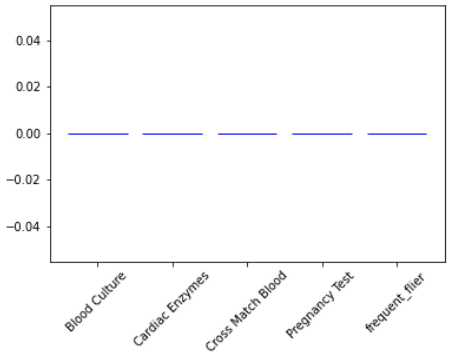


((g)) Fourth sample features

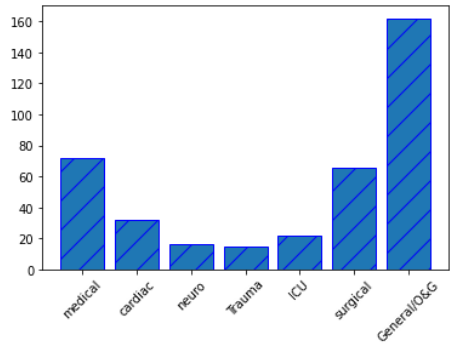


((h)) Fourth sample labels

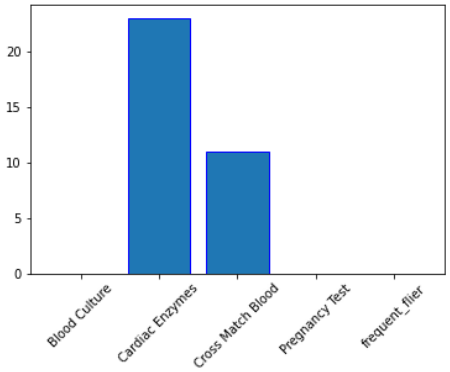
Figure 5.32: Feature and label distributions for the first four samples in Figure 5.31.



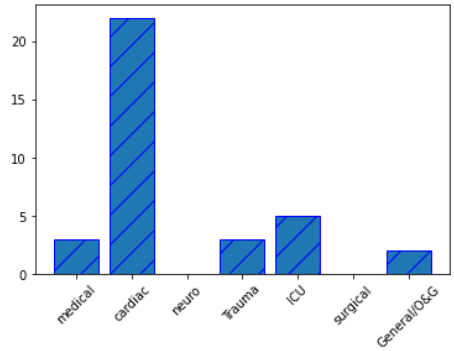
((a)) Fifth sample features



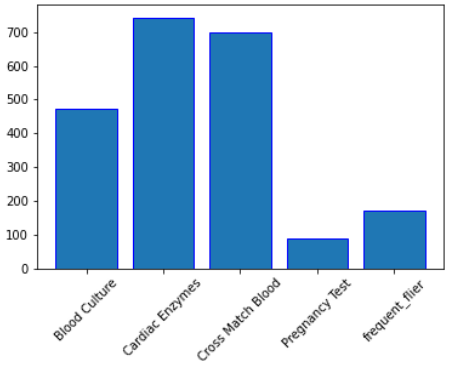
((b)) Fifth sample labels



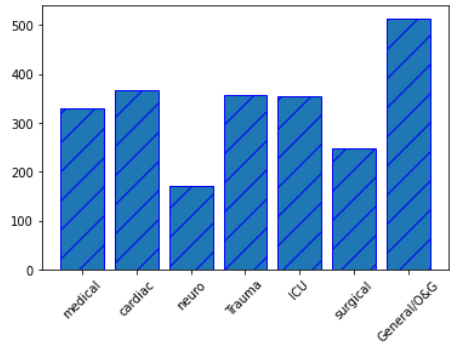
((c)) Sixth sample features



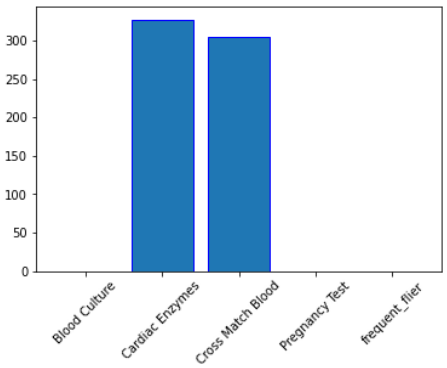
((d)) Sixth sample labels



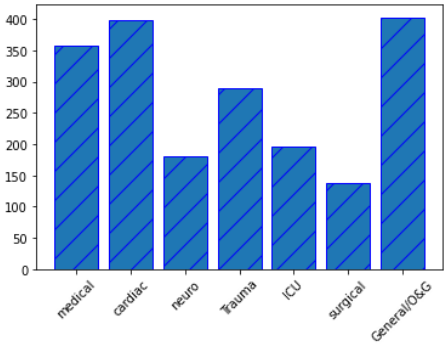
((e)) Seventh sample features



((f)) Seventh sample labels

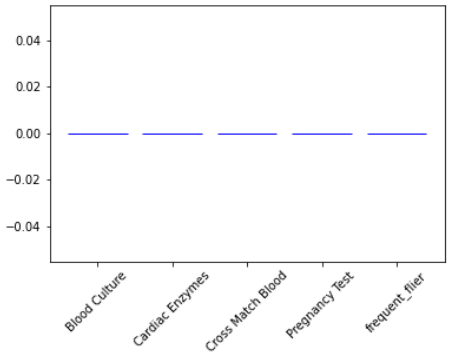


((g)) Eighth sample features

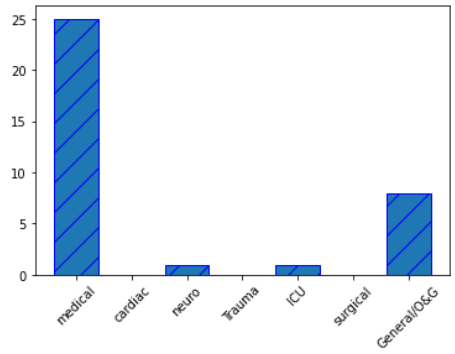


((h)) Eighth sample labels

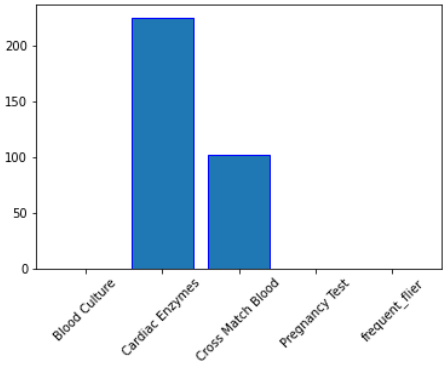
Figure 5.33: Feature and label distributions for the fifth to eighth samples in Figure 5.31.



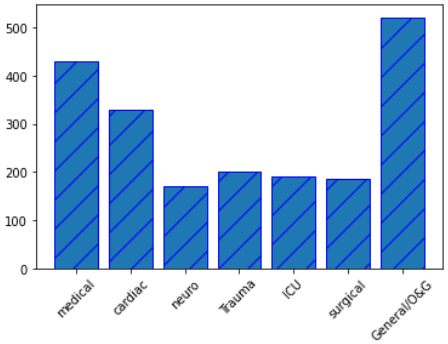
((a)) Ninth sample features



((b)) Ninth sample labels



((c)) Tenth sample features



((d)) Tenth sample labels

Figure 5.34: Feature and label distributions for the ninth and tenth samples in Figure 5.31.

Chapter 6

Scheduling Learning Using Meta-Gradients

Chapter Abstract

Given the abundance and ease of access of personal data today, the implementation of rigorous and robust processes to protect the security and privacy of individual data has become of paramount importance, particularly in the healthcare domain. In this chapter, we utilise patient data extracted from multiple independent data centres to train a machine learning model without sacrificing patient privacy. We develop a scheduling algorithm, in conjunction with the student-teacher algorithm presented in the previous chapter, deployed in a federated manner which allows a central model to learn from batches of data at each federal node. The teacher acts between data centres to update the main task (student) algorithm using the data that is stored in the various data centres. We show that the scheduler, trained using meta-gradients, can effectively organise training and as a result train a machine learning model on a diverse dataset without requiring that the local data are uploaded externally, concatenated or exchanged, thus improving patient data security. We achieve state-of-the-art performance and show how our method overcomes some of the problems faced by using federated learning alone, such as data and node poisoning. We further show how the scheduler can be used as a mechanism for transfer learning,

allowing different teachers to work together in training a student for state-of-the-art performance.

6.1 Introduction

Federated learning is a field that has emerged in recent years due to the increasing abundance of data available and the risks that this poses to individuals. Privacy - particularly of personal information - is of great importance and must be protected by researchers working in machine learning. In parallel, the emergence of electronic health records (EHRs) has allowed the digitisation of much personal information pertaining to the health statuses of individuals. EHRs are used in many machine learning research projects [161]. This often involves the transfer and storage of very sensitive information which increases the risk of data leakage. As a result, federated learning can minimise this risk by utilising the data at its source rather than transferring it to a separate research institution for processing.

Machine learning researchers working with EHR data will be familiar with the difficulty of gaining access to this information at the outset of a research project. It can be a very lengthy and complex process (for good reason) to gain access and utilise the EHRs of one healthcare institution, much less accessing the datasets of many. Federated learning offers an alternative in that it allows the patient data to be utilised while reducing the risk of their privacy being compromised.

Federated learning is not without its own limitations, however. The different datasets that are stored in the discrete nodes may have different underlying distributions due to their data collection processes, which can make machine learning across multiple domains difficult. Furthermore, there may also be disagreement between the datasets on labelling and strong imbalances of classes between nodes. These can all lead to difficulties in the training and convergence of the model being trained.

To overcome some of these limitations, in this work we propose the following setup.

Firstly, we use federated learning to i) protect the privacy of patients by minimising the transfer of their data away from the local institution and ii) improve the quality of our machine learning model by utilising a diverse range of datasets sourced from different hospitals. As we are 'blind' to the data at the nodes, we propose the use of the student-teacher network setup. This is because we have shown the student-teacher network to be capable of selecting appropriate training data to maximise classification performance on a given problem. This means this can be utilised on a dataset without needing guidance from a practitioner viewing the dataset. The teacher (the reinforcement learning agent presented in the previous chapter) will have access to the local servers (which host that particular hospital's dataset) and be able to select the appropriate data for the 'student' (our model) to be trained on at that time. The 'scheduler' will be responsible for directing the teacher to a given data centre to select data on which to train.

To summarise:

- *The Student*: - the machine learning model that we are training. This agent takes the same role as the student discussed in the previous chapter. This is a classification network that is independent of the number of datasets or nodes our federated system is comprised of.

- *The Teacher*: - a reinforcement learning agent that selects data from a data centre based on the state of the student. This essentially defines a curriculum at each training step for the student. This takes the same role as the teacher discussed in the previous chapter.

- *The Scheduler*: - a feedforward neural network trained using meta-gradients that directs the teacher to the appropriate data centre for training. This is also based on the state of the student at each training iteration (i.e., the inputs to the teacher and the scheduler are the same).

Section 6.2 discusses the related work that has been carried out, and Section 6.3

provides a detailed description of how our algorithm works. Section 6.4 details the datasets against which we benchmark our method. We then present the results in Section 6.5 and discuss their significance and interesting behaviours of our model in Section 6.6.

6.2 Related Work

Federated learning has been used by researchers to exploit larger pools of data for training [165], to preserve data privacy [190] and for distributing computational resource requirements [217]. Federated learning has also been used for healthcare applications to simultaneously utilise multiple datasets to train a model on patient data.

In this work we create a model that learns in a federated fashion through the interaction of a scheduler that is trained using meta-gradients and a student-teacher algorithm that is trained using reinforcement learning.

Meta-learning has been used effectively in [187] where the loss of a student model on a validation set was used as a signal to update the weights of a generative model. This work demonstrated the rapid and effective training of methods that exploit meta-gradients. Meta-learning was also used in [218], where the meta-gradients are used to tune the parameters of an actor-critic algorithm. As a result of the efficacy of this method in these domains we choose to use meta-gradients to schedule which data centre is selected. The gradients from this update are then used to update the student model.

The meta-learned scheduler chooses a node representing a data centre where a student-teacher algorithm is used to sample data. Student-teacher algorithms have been used in multiple works, with the general premise that one algorithm (teacher) is trained to train another (student) [125], [145]. These methods have also been used with a curriculum [43], where the curriculum is either pre-defined and exploited by

the teacher [196] or implicitly learned by the teacher during training [117].

Federated learning is a method of training a model (in our case a deep neural network) by using data from multiple centres, without having central access to each of them [130]. Local models at each of the data centres are iteratively updated and aggregated to form a global model. At each round of iteration, a central scheduler samples a subset, m, of local models, S_m, and sends them the current global model G^t. Each member of S_m then updates this global model using their local data to create an updated model, L^{t+1}. While this has been shown to work in many cases, [216] make the argument that there is inherent difficulty in updating neural networks in this manner. They argue that the permutation invariance of the summation operand renders averaging in the parameter space a naive approach. For meaningful averaging to be performed, the permutation must first be undone.

6.2.1 Compromising Federated Learning

One of the vulnerabilities of federated learning is that nodes being compromised can significantly affect the training of the global model [144]. Such attacks can either 'poison' the data found at one of the nodes (known as an *adversarial* attack), or significantly bias the model that is trained at one of the nodes, leading it to highly skew the aggregation step [215]. There is also the possibility of the attack being a *single-shot attack* or a *repeated attack* [200]. In the single shot case only one of the nodes is compromised, whereas in the repeated case multiple nodes can be compromised at any given time. Various studies discuss and summarise how federated learning can be compromised by introducing a backdoor into the training process [118], [194]. A backdoor is an attack that causes a classifier to produce unexpected behaviour if a specific trigger is added to the input space. An example is a sticker being added to an image and associating this with the incorrect label [118].

Defences against these attacks have been developed, with some authors using pruning of redundant neurons for the core classification task [153], or using outlier

detection to detect potential triggers [188], and re-training and pre-processing inputs [126]. In this work, we aim to overcome these limitations and build defence into the training procedure through using a student-teacher network that actively selects the data on which to train. By designing the scheduler to be 'rewarded' (note that this is not reward in the reinforcement learning sense, but providing a small error signal for good performance) for good performance on the secure validation set, we can overcome poisoning at local nodes.

6.3 Methodology

Our method is comprised of three agents in the training setup, the student, the teacher and the scheduler. In this section, the overall network architecture is displayed and each component of the setup is explained individually. A visual representation of the overall training process is provided in Figure 6.1.

6.3.1 Federated Learning

In the following sections we describe the techniques employed in federated learning.

Model Aggregation

As previously described, model aggregation is currently the most popular form of federated learning in use. This involves the training of n separate models at the n nodes in the system, and these local model differences in comparison with the global model are aggregated. The global model is then updated as:

$$G^{t+1} = G^t + \frac{\eta}{n} \sum_{i=1}^{n} \left(L_i^{t+1} - G^t \right) \tag{6.1}$$

where G is the global model at iteration t, η acts as a learning rate for replacing the global model with the aggregate of the local models, L_i.

Matched Averaging

In matched averaging, rather than take the difference between the entire model and aggregating these as a learning signal, the weights are considered layer-wise (in terms of the neural network layers) and the differences are taken layer-wise. This allows each layer to be treated independently. Because model aggregation is based on summation, there are many different combinations that can lead to the same result. To avoid this, a permutation matrix is introduced which multiplies the weights of each layer, thereby greatly reducing the probability of combinations providing a similar result. Note that this is very similar to the function of the reference vector introduced in Chapter 5 Section 5.4.5.

Cross-Device versus Cross-Silo

There are generally two different federated learning scenarios that are considered in the literature. The first is the cross-device scenario. This is the case where many, usually low-powered, devices are connected to a system and the data collected from them is used for training. An example is mobile phones connected to a central network; each phone carries significantly different data and is prone to dropping off the network during periods of low-signal strength to the cellular network. Such federated systems therefore need to be capable of operating with changing numbers of clients, as well as utilising data that are significantly differently distributed.

The cross-silo approach is less restrictive. In this case there are generally much fewer clients in the system, and it is assumed that all of the clients have adequate computational power. The data stored at each of the silos don't necessarily have to be the same, but it is assumed that there are sufficient quantities that the underlying distributions of the data should be more stable than they would be in the cross-device scenario. As hospitals have EHR systems with rich datasets and usually adequate computing facilities, we conclude that our problem is best addressed by a cross-silo approach.

6.3.2 The Overall Setup

The overall setup of our federated learning training routine is as follows: we have a scheduler that selects the node/centre from which data will be sampled (this can be one-hot or we can select multiple nodes). In the case of a one-hot setup, only one data centre is selected for data sampling. If a softmax is used on the output of the scheduler, proportional sampling from each data centre can take place. The teacher at the selected node can then select a batch of data according to the state of the student (note that this is the same as the way in which the teacher sampled data based on the student state in the previous chapter). The student at the node is a copy of the global student. (The global student is the version of the student model kept on the central server. A copy of this model is distributed to every node in the federated system after every scheduler update). The student is used to forward pass the batch of data selected by the teacher and return the loss. In the one-hot scheduler scenario, the loss is sent back to the global student model to update the weights via backpropagation. In the multi-node learning scenario, the losses from the models at all the selected nodes are aggregated and fed back to the global model for updating.

6.3.3 Data Pre-Processing

The first step we must take in order to exploit our teacher setup is to rank our data according to some metric. As shown throughout this thesis, the Mahalanobis distance has proved an effective entropy metric for the ward admission location prediction problem, so this is used once again for curriculum construction or data ranking. For image datasets the cosine similarity was used as suggested in benchmarking approaches.

As the tabular data found in electronic health record systems consist of multiple data types, we encode these using a denoising autoencoder as previously discussed. However, due to the federated nature of this setup, the trained encoder is now distributed to all the nodes so that the data at each node may be processed in the

same way for consistency across nodes.

6.3.4 The Teacher

For the student-teacher interaction we follow the setup in Chapter 5. The task of the teacher is to select a batch of data from the curriculum by selecting the index along the curriculum and the 'width' around that index to include in the selection. The following sequential steps are implemented:

- The data at each node is organised into N curriculum batches according to some metric H
- The teacher selects one or more batches for feeding into the student
- A pre-trained autoencoder is used to create a latent representation of the batch
- The student is trained on this batch and its performance on a separate validation set is recorded.

The teacher is a reinforcement learning agent and therefore is tasked with minimising the Bellman loss function as presented previously. As we also choose to use an actor-critic setup for the teacher, the action space and Q-function are separately parameterised. This allows a continuous action space and the actor that selects actions is updated using the following loss:

$$\nabla\theta_t^{\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q\left(s, a \mid \theta_t^Q\right) \mid_{s=s_i, a=\mu(s_i)} \nabla_{\theta_t^{\mu}} \mu\left(s \mid \theta_t^{\mu}\right) \tag{6.2}$$

where Q is the Q-function and μ is the policy. Notice that this time θ carries the subscript t. This corresponds to the task the teacher is assigned to learn how to teach. As the setup requires multiple teachers, separate training of the teachers is required.

The teacher can either be pre-trained, or jointly trained with the scheduler. The teacher can also either be trained on the dataset of one node and distributed to the

remaining nodes, or independently trained at each node. The latter is preferable due to the ability of the teacher to adapt to the dataset at hand. However, the former is useful when not all of the nodes in the federated system have access to adequate computational power. The intuition is that the curriculum strategy learned by the teacher should be general for the required task and thereby provide strong performance. This was seen with the teacher trained on the hospital admission location task providing a training strategy that proved effective on the MIMIC-III mortality prediction task.

6.3.5 The Student

The input to the teacher is the current state of the student. The student in this chapter is again a feedforward neural network that is tasked with classification. The state of the student is defined as a representation of the weights of the student. Given a matrix of weights, W^{ij}, between layers i and j of the network, for each row, $W^{ij}_{n:}$, we take the inner product of the row with a fixed reference vector a. From this inner product we extract $|\langle W^{ij}_{n:}, a\rangle|$ and $\angle(W^{ij}_{n:}, a)$ for $n = 1, 2, \ldots, M_i$ where M_i is the number of hidden nodes in layer i. These values are concatenated to represent the row and this process is repeated for all rows to build the vector. For more hidden layers the process is repeated until we have one vector representing the network. This provides us with a representative vector, $\mathbf{v} \in \mathbb{R}^{2\left(\sum_l^h M_l\right)}$, where h is the number of hidden layers. This vector is fed to the teacher to understand the state of the student. This is the same representation used in the previous chapter.

6.3.6 The Scheduler

The scheduler is the last of our agents in the training setup. This is also a feedforward neural network that takes the student state as input and selects from which of the nodes the training data should come at the current iteration of training. This agent is trained

using the meta-gradients generated from validation losses in a similar way to that described in [187]. There is an 'inner' loop of training, in which the student-teacher interaction takes place. In the 'outer' loop, we aggregate the losses on validation sets at each node and use this aggregated loss as the signal to update the weights of our scheduler. The aggregation of the losses from the distributed validation sets is important, because using the signal of only one validation set may result in overfitting to the distribution on that node. It is also important from a security perspective; should the validation set being used for loss calculation be poisoned, this would result in catastrophic updates for the global model. However, by taking an aggregate of losses across independent validation sets, we encourage the scheduler to select in a way that benefits performance on all of these. Figure 6.1 shows diagrammatically

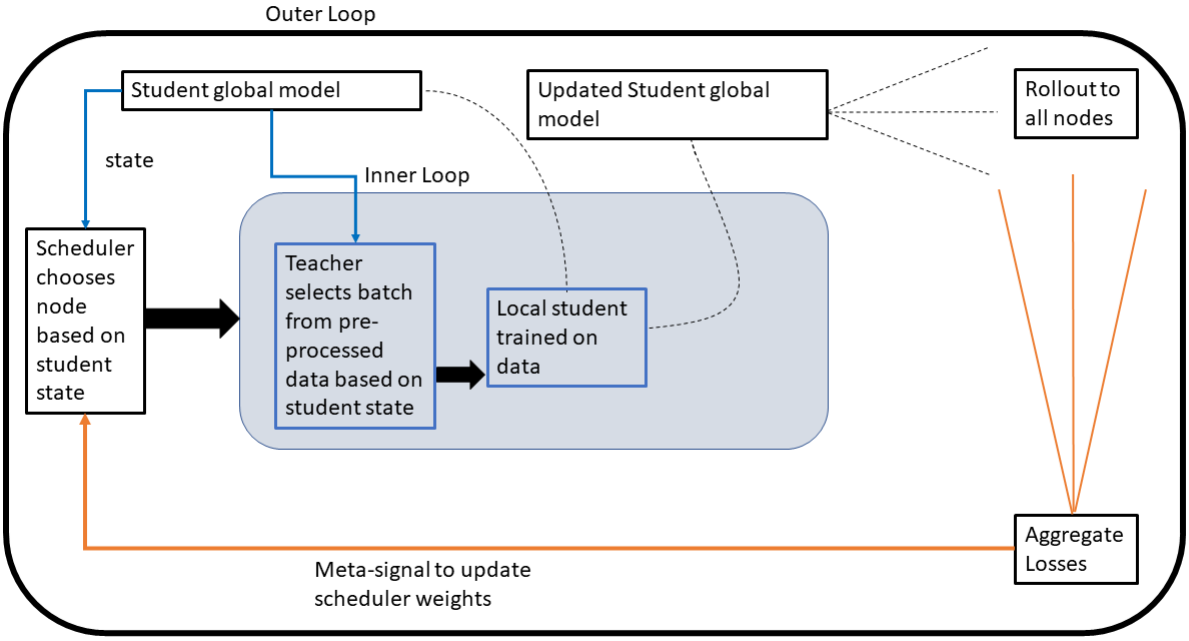


Figure 6.1: A diagram displaying what happens in the training routine every iteration. The black arrows indicate selection, the blue indicate the extraction of state, the dashed indicate the transfer of a model and the orange indicate the movement of losses. The inner loop occurs at each node.

the training procedure at every iteration of training. In the outer loop the scheduler selects the node(s) to use for data selection. The global student model is 'sent' to the

node, the teacher at the node selects the data in the inner loop and trains the student network on this. The student is then sent back to the central node and distributed to all nodes (note that the local student models are only used to get validation accuracies reported). The students are tested on separate validation sets at each node and their individual losses are aggregated by summing them. These are then used as the loss to update the scheduler. The inner loop loss function is dependent upon the target task with crossentropy used for classification and mean squared error used for regression. The scheduler is updated as:

$$\theta_{sc}^{t+1} = \theta_{sc}^{t} + \omega \sum_{0 \leq t' \leq t} \alpha^{t-t'} \nabla \left(\mathcal{L}_{inner} \right) \tag{6.3}$$

where $\mathcal{L}_{inner}$ is given by:

$$\ell_{inner} \left(T \left(\mathbf{s}_t; \theta_{te} \right), \mathbf{y}_t^*; \theta_{st} \right) + \sum_{n=1}^{N-1} \ell_{inner} \left(\mathbf{x}_v^n, \mathbf{y}_v^n; \theta_{st} \right) \tag{6.4}$$

where θ_{sc} are scheduler weights, ω is a learning rate for stochastic gradient descent, α is a momentum hyperparameter, ℓ_{inner} is our local task loss function, T is the teacher network taking as input the student state at iteration t, $\mathbf{s}_t$, $\mathbf{y}_\mathbf{t}^*$ is the ground truth associated with the teacher selection and θ_{te} and θ_{st} the teacher and student parameterisations respectively. $\mathbf{x}_v^n$ and $\mathbf{y}_v^n$ are the features and labels of the validation set of node n respectively.

Entropy Loss

As the scheduler is trained using meta-gradients, the model may converge after a set number of iterations. While we would like convergence of the student (i.e., the task solving model), we do not necessarily need the scheduler to converge. In fact, using the loss on the validation set as the signal to update the scheduler automatically prevents the scheduler from converging. This is because, should the scheduler converge on selecting a particular data centre, the validation scores on the other data centres will

deteriorate thereby increasing the aggregated loss on the validations and providing an update signal to the scheduler. This, however, was found in practice to require many iterations of training, therefore in order to encourage the exploration further, we add an *entropic loss* term to the scheduler. This takes the form:

$$\ell_{ent} = \frac{1}{H\left[S\left(s_t; \theta^t_{sc}\right)\right] + \epsilon} \tag{6.5}$$

where H is entropy, $S\left(s_t; \theta^t_{sc}\right)$ is the softmax output of the scheduler and ϵ is a small positive value (we use 10^{-5}) to prevent potential division by zero. ℓ_{ent} is then added on to the end of the expression shown in Equation 6.4 to discourage fast convergence of the scheduler.

For tabular classifications, we define the student to be a feedforward neural network consisting of 2 hidden layers and 50 nodes in each layer. These are all activated with ReLU activations. For the image recognition tasks we initialise a student that has 4 convolutional layers with 32 filters of size 3x3 in the first two layers and 64 of these in the second two. These are all activated by ReLU and maxpooled and are followed by 3 feedforward layers of size 50 nodes each.

For the results reported in Section 6.5, we use a pre-trained teacher (i.e., the teacher has been trained using reinforcement learning on different students for the same classification problem). The teacher has 2 hidden layers and 150 nodes each activated by ReLU apart from the final layer (of size 2) which is activated by a tanh function. For the scheduler we use a feedforward neural network with 2 hidden layers and 100 nodes in each layer activated by ReLU. The output is activated by a softmax of size the number of nodes in the federated system.

6.4 Datasets

The same dataset for our main investigation, the ward admission location prediction, is used once again. This is built from the EHR of the OUH NHS Trust. In order to

treat this in a federated fashion, the data are split randomly into N datasets so that each can act as a separate machine in a federated system. A training set of 60% of the dataset is used and balanced with 20% for validation and 20% for testing. The training and validation sets are then evenly divided according to the number of nodes in the federated system.

CIFAR-10

To test our methodology on the image space we also report results on the CIFAR-10 image recognition dataset [44]. We use 40,000 training examples for the training set and 10,000 each for validation and test set examples. We once again randomly divide the dataset into N datasets to mimic a federated learning system. It should be noted that in Table 6.1 the CIFAR-10 dataset has been split into three samples of size 15000, 15000 and 10000 due to the possibility of the teacher selecting a full training set batchsize and memory constraints.

MNIST

As previously, we utilise the MNIST dataset [15] another publicly available dataset against which to assess our results. We use 30,000 examples for training, 10,000 for validation and 10,000 for the test set. Once again the dataset is partitioned into N datasets to emulate the federated learning approach. The datasets have purposefully been chosen to be quite different to one another, in order to investigate how the scheduler can combine the strengths of different teachers in order to train a student. However, the Results also consider the use of teachers that are trained to teach very similar tasks.

eICU

In order to validate our results on real-world data collected from different hospitals, we introduce the eICU dataset [158], also hosted on Physionet [19]. The task here is

mortality prediction (binary classification) based on features extracted from admission to the ICU as is done in [214]. As this dataset contains identifiers for individual hospitals, we are able to create nodes corresponding to each hospital. The features selected are as outlined in Chapter 3. We choose to learn from the eight hospitals with the largest populations in the dataset, generating 8,594 instances. We sample 60% from each node to keep as the training set, and keep 20% for the validation set and the final 20% as the test set. As per usual, the validation set is kept on the local node for performance aggregation during the scheduler training.

6.5 Results

As our work lies in the intersection of two research areas within the field of machine learning (namely federated learning and student-teacher learning), we choose to use baselines from both of these fields as comparators. From the student-teacher learning side, we will assess how our method compares in terms of final model performance only. For the federated learning comparison we will compare not only the final model performance but also the robustness of the method to attack.

Final Model Performance

Table 6.1 shows how the performance of our federated learning method (FLST) compares to other state-of-the-art classification methods. The baselines we use are the reinforcement learning trained student-teacher setup without scheduling [196] (RLST), two state-of-the-art methods used for classifying tabular data (DeepFM [119] and Deep+CrossNet [138]), and two state-of-the-art classifiers for image recognition (GPipe [169] and DenseNet [148]). As baselines, we train a standard feedforward neural network (a convolutional neural network for the image datasets) using stochastic mini-batch training (SMBT) and a curriculum (CURRIC) for comparison. For the OUH (IORD) dataset, not only does the federated learning system outperform the

Table 6.1: Average classification accuracies and standard deviations for various baseline and state-of-the-art methods on the Ward Admission (tabular), MNIST (image) and CIFAR-10 (image) datasets and the AUC score on the eICU dataset. All models are averaged over the same five seeds, apart from those highlighted with * which indicates that the accuracy reported from the cited text is quoted.

Method	OUH Acc (SD)	MNIST Acc (SD)	CIFAR-10 Sample 1 Acc (SD)	CIFAR-10 Sample 2 Acc (SD)	eICU AUC (SD)
SMBT	0.45 (0.01)	0.91 (0.01)	0.65 (0.02)	0.65 (0.02)	0.80 (0.02)
CURRIC	0.48 (0.02)	0.93 (0.02)	0.68 (0.01)	0.68 (0.01)	0.81 (0.01)
DeepFM	0.59 (0.01)	N/A	N/A	N/A	0.81 (0.01)
Deep+CrossNet	0.58 (0.02)	N/A	N/A	N/A	0.82 (0.02)
DenseNet* [148]	N/A	**0.99** (0.01)	0.96 (0.01)	0.96 (0.01)	N/A
GPipe* [169]	N/A	**0.99** (0.01)	**0.99** (0.01)	**0.99** (0.01)	N/A
RLST	**0.62** (0.02)	0.95 (0.02)	0.90 (0.01)	0.90 (0.01)	0.85 (0.02)
FLST	0.60 (0.02)	0.94 (0.01)	0.91 (0.01)	0.90 (0.01)	**0.86** (0.01)

Table 6.2: Average classification accuracies and standard deviations for various baseline and state-of-the-art methods on the hospital ward admission (tabular), eICU (tabular), CIFAR-10 (image) and MNIST (image) datasets. All models are averaged over the same five seeds. We also show how the models perform when subjected to a data poisoning attack and a local model poisoning attack.

Method	Accuracy Acc (SD)	Model Poisoning Acc (SD)	Data Poisoning Acc (SD)
FedAvg (hospital)	0.55 (0.01)	0.33 (0.01)	0.47 (0.02)
FedMA (hospital)	0.56 (0.02)	0.45 (0.02)	0.52 (0.01)
FLST (hospital)	**0.60** (0.02)	**0.59** (0.01)	**0.59** (0.02)
FedAvg (CIFAR-10)	**0.93** (0.01)	0.62 (0.01)	0.65 (0.02)
FedMA (CIFAR-10)	0.93 (0.02)	0.77 (0.02)	0.68 (0.01)
FLST (CIFAR-10)	0.90 (0.01)	**0.85** (0.01)	**0.87** (0.02)
FedAvg (MNIST)	**0.95** (0.01)	0.77 (0.01)	0.73 (0.02)
FedMA (MNIST)	**0.95** (0.01)	0.83 (0.01)	0.81 (0.02)
FLST (MNIST)	**0.95** (0.02)	**0.86** (0.02)	**0.88** (0.01)
FedAvg - AUC (eICU)	0.80 (0.01)	0.65 (0.07)	0.69 (0.12)
FedMA - AUC (eICU)	0.82 (0.01)	0.71 (0.09)	0.72 (0.04)
FLST - AUC (eICU)	**0.86** (0.02)	**0.80** (0.04)	**0.81** (0.03)

baselines, but it also outperforms current state-of-the-art models in tabular learning.

This is similar to what was seen in the case of the student-teacher and is most likely

due to the strategies that the teachers use when training the student. It should be noted that the teachers at each node are the same for this experiment. While it outperforms the baselines it does not outperform the student-teacher setup itself. This is likely due to the conflicting requirements at each node. Each node has slightly different underlying distributions (this will be controlled in later experiments), and so the performance of the student can decrease before it increases once again. As a result, convergence to a strong performance generally takes longer in the federated setup. However, this is with the added benefit of providing a more secure training routine (from model and data poisoning).

The federated setup once again shows a strong performance on the MNIST dataset. It performs comparably with the student-teacher alone. While it performs worse than the baselines, it should be noted that the baselines are highly tuned and curated architectures for image recognition and consist of millions of parameters. In comparison, the convolutional neural network used in the federated setup consists of 4 convolutional layers and 3 linear layers. For the much simpler model to provide an almost comparable performance shows the power of strategic sampling for training a network.

For the CIFAR-10 dataset, a similar pattern is seen as with the MNIST dataset. A strong performance is attained, but this is not comparable with a highly-tuned, highly parameterised network. However, it is still a strong performance compared to the baselines and approaches state-of-the-art performance. For the eICU dataset the federated setup greatly benefits learning from multiple hospitals, even improving upon the student-teacher setup described in Chapter 5 in terms of final performance. We also observe that the score is not altered greatly when the local models are poisoned, and is even less impacted when the local data are compromised. This is highly encouraging for real-world utility of this algorithm, as clinical test results are often recorded erroneously in hospitals and as such are a real-world instantiation of data poisoning.

It has been seen that the federated system is capable of producing a performance

that competes with state-of-the-art models that are trained in a centralised manner. Figure 6.3 shows the scheduler outputs during training for the hospital admission problem, and Figure 6.4 shows the performance of the student and the actions of the teacher during this training. We see that different data centres take precedence for training with time and how this leads to a steadily increasing performance of the student.

Learning from Multiple Hospitals

When training on the eICU dataset, we utilised data from individual hospitals as the separate nodes in the federated system. An essential question that arises is how the increasing the number of nodes in the system affects the final performance of the student.

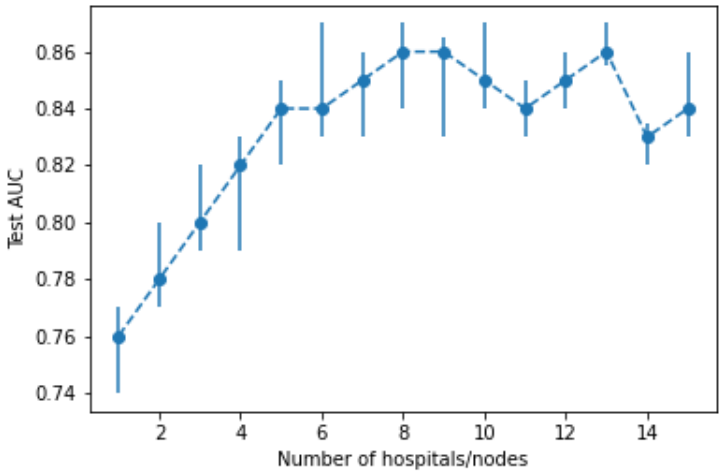


Figure 6.2: The number of nodes in the federated system (i.e., number of hospitals) versus the student performance at test time. Note that in this figure the seed of the scheduler is held constant, and the error bars correspond to the scores generated from using different hospitals for training. So for N=k nodes, the error bars correspond to training only using data from ten different combinations of k hospitals.

In Figure 6.2 we carry out an ablation on the number of nodes in the system and how it affects performance. In this plot, the scheduler is held at constant seed and the error bars are generated due to selecting different hospitals to train on. There

are a total of 208 hospitals in the eICU dataset, but we only select from the fifteen which have the highest volume of data recorded. As a result, Figure 6.2 shows error bars for the difference in performance when ten different combinations of hospital data are used. We can see that with more data being used in the system, the final performance generally increases. We also see that the variance in the performance starts to decrease with the increase in the number of nodes. Given the increase in the volume of data being trained with as nodes are added, this aligns with expectations. The performance plateaus at approximately 8 nodes, indicating that adding more nodes to the system may not necessarily be beneficial for performance. This could be useful for real-world application as it will allow practitioners to prioritise data centres with the highest quality data for their federated systems.

Federated Performance

Table 6.2 shows how the proposed setup compares against other federated learning algorithms. We see in the case of the hospital admission location prediction, the student-teacher federated setup outperforms the state-of-the-art federated baselines. In fact, for all datasets in the presence of data poisoning or model poisoning, the student-teacher federated setup consistently outperforms the state-of-the-art federated baselines. However, for 'vanilla' training, the training routine provides a marginally worse final performance from the student. This is in fact a trade-off that is required for the enhanced security against poisoning. In the case of FedMA and FedAvg, a model aggregation procedure is employed whereby distributed models are trained until convergence before an aggregate of these models is used to provide the overall end model. In the case of the student-teacher federated setup, learning is carried out on-the-fly. As a result, convergence to a strong final performance takes longer than it would if each node were able to train a model locally. This is due to the increased communication cost between the data centres. After every update, the central node must communicate the new global model to all nodes. The nodes must all feedback

the losses of the new model on their validation sets. These steps are repeated every time the scheduler and the student are updated, and as a result, a strong performance generally takes a greater number of iterations than a model aggregating approach would. However, with a long computational budget, the equivalent performance to the student-teacher setup alone can be achieved.

This problem can be exacerbated by having many nodes in the federated system. The more nodes there are, the more the trained model needs to satisfy the independent validation sets. As a result, this will require longer training in order to allow the scheduler to identify the best data centres to make this training effective across all nodes.

6.6 Implicit Defensive Setup

Robustness of Federated Training Routine

Table 6.2 showed how our method performs when compared to other federated learning algorithms. For our baselines we used FedAvg [130] in which the local models at each node are aggregated before being averaged, as well as FedMA [216], which constructs a shared global model in a layer-wise manner by matching and averaging hidden elements (such as neurons and hidden states). In order to investigate how performance deteriorates when exposed to different backdoor attacks, data poisoning and model poisoning were employed. The data poisoning was carried out by randomising the features of every datum at node c, such that $\mathbf{x}_n^c \sim \mathcal{N}(0,1)$. With respect to the model poisoning scenario, consider the set of weight matrices, $\mathcal{W}_t^c$ between each layer of a teacher, t, at node c. For each weight matrix, $\mathbf{w}_i$, we assign randomly sampled values for each element such that $\mathbf{w}_i \sim \mathcal{N}(0,1) \quad \forall\, \mathbf{w}_i \in \mathcal{W}_t^c$. It was hypothesised that through the use of the scheduler, this approach provides an added layer of redundancy in the system thereby allowing attacks to be avoided after their implicit detection through degraded performance on the validation sets stored at all nodes. In order to

further test this, this section will focus on robustness verification through experimental setups.

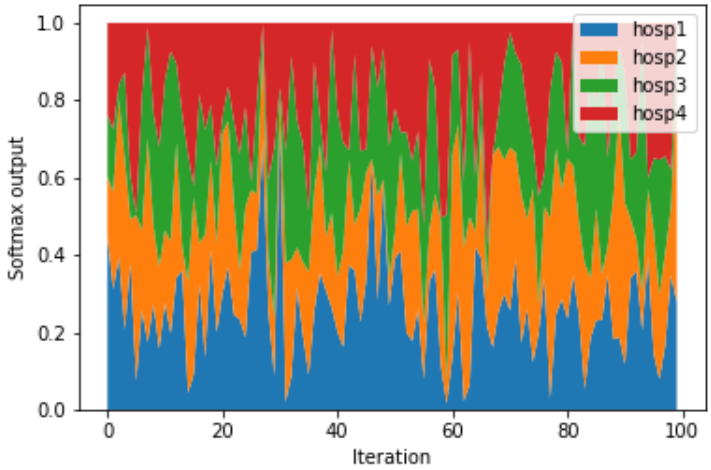


Figure 6.3: A four-node federated system being scheduled for training a student on the hospital admission location prediction problem. Each colour represents a different node being selected for training.

Figure 6.4: The performance of the student and actions taken by the teachers (scheduled according to Figure 6.3) at each node to train the student. The orange 'x' and blue bar indicate the first and second outputs of the teacher respectively (index of data along the curriculum that is selected and how much data around this to include in that batch). The red line shows the performance of the student on the held-out test set on the hospital admission problem.

Iterative Learning

In the proposed approach, local models are exploited to select data and extract gradients that are used to update a global model. The advantage of this approach is that it provides flexibility for a poisoned node within the federated system to be discounted or unused. This can also be done on-the-fly without the need to inspect either the local models after each training run or the data stored at each node. Figure 6.3 shows how the scheduler selects in a non-poisoned scenario. In this setup, the one-hot selection is used, meaning that the largest softmax output is chosen as the data centre to learn from. We observe that the largest contribution changes regularly, meaning the various data centres are regularly being exploited for their data. Figure 6.4 shows how the student performance changes with each iteration of training. The performance on the held-out test set is assessed as the student trains, but is never used for any updates. This provides a state-of-the-art performance but what is particularly interesting is that the confederation of teachers have each chosen to use low entropy data while marginally varying the batch width. However, as the training set is now split equally across four nodes, the insights that were gained in the previous chapter may no longer apply here due to the underlying distributions (and ranking along the Mahalanobis index) now being different.

Selecting the Appropriate Teacher

The design of the first set of experiments, allows us to investigate whether the scheduler will be able to select the appropriate teacher for the specific learning task. The scheduler's task is to select a teacher from three different nodes to train the student. Three pre-trained teachers for separate tasks (hospital admission location, CIFAR-10 and MIMIC-III mortality prediction as presented in the previous chapter [107]) were saved to the nodes and the scheduler was able to choose from these in order to source the data for training. The data itself is CIFAR-10 data, and only the teacher is changed for these experiments. The student being trained by the scheduler

shown in Figure 6.5 is being trained on the CIFAR-10 image recognition task. Figure 6.5 shows how with training, the scheduler learns to assign the teaching job to the node that contains the teacher trained to teach CIFAR-10 learning. Initially, the teachers all have the same softmax output probability of approximately 0.3. As training progresses, the response from the scheduler becomes entirely dominated by the node in the federated system that corresponds to the appropriate teacher for the task. From this it is seen that this approach is sufficiently robust to select the appropriate teacher for the learning task.

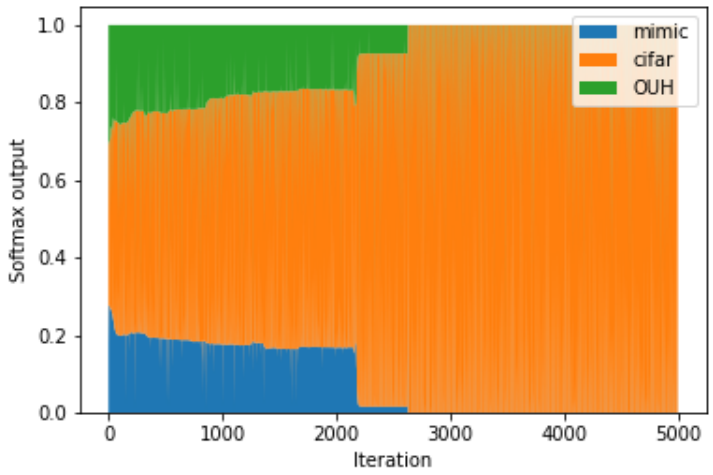


Figure 6.5: Scheduler selection as the student is trained in a 3-node system. The student is being trained on CIFAR-10 and the scheduler learns to use the CIFAR-10 teacher to teach the student.

We repeat this analysis, this time with a student focusing on the ward admission dataset. Figure 6.6 shows how the scheduler selection develops for this problem.

Once again, it can be seen that the scheduler response becomes dominated by the teacher that is specifically trained to teach the ward admission prediction problem. This is again implemented in a one-hot scheduler fashion, such that the maximum softmax output is selected as the data centre from which to train. Although there is some increased allocation of probability mass to the MIMIC and CIFAR teachers towards the end of the graph, this is probably due to the entropy loss, and with further

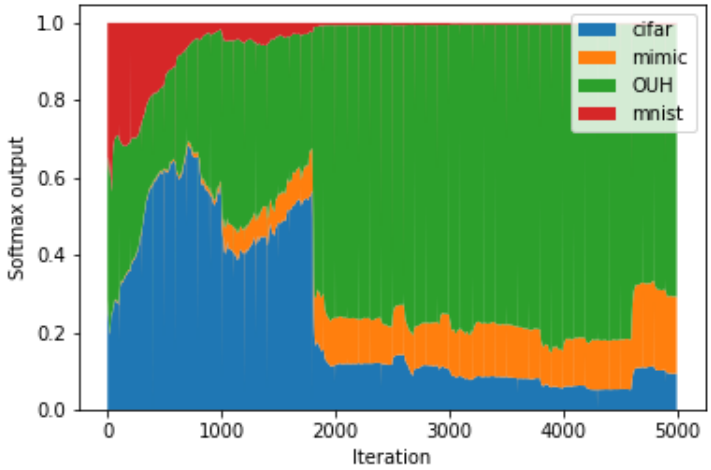


Figure 6.6: Scheduler selection as the student is trained in a 4-node system. The student is being trained on the IORD ward admission location problem and the scheduler learns to use the teacher associated with this.

training the probability mass would once again be allocated to the ward admissions teacher.

Scheduler Losses

The scheduler loss is defined as the aggregation of the independent validation losses at each of the nodes. As a result, it must be ascertained that these losses do not monotonically decrease (as is usually desired in the training of neural networks), as this would not incentivise the selection of alternate data centres from which to train.

As seen in Figure 6.7, the loss does not monotonically decrease. Where the training of the student does not improve the performance on all of the validation losses, we see increases in the loss. As the scheduler experiments with different selections, the losses decrease and increase. Eventually however, with enough training, the losses begin to converge to a low value indicating that a strong performance is achieved on all of the independent validation sets.

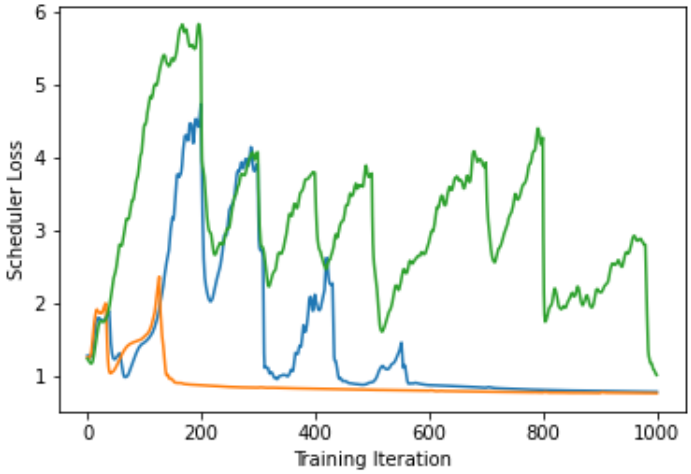


Figure 6.7: How the scheduler loss changes with training for a student trained on the MNIST dataset. Three separate scheduler initial seeds are plotted.

Compromised Data at Nodes

This set of experiments investigated the robustness of the proposed method to attack through poisoning of data at local nodes in the federated system. By scheduling training through the use of meta-gradients, it was hypothesised that there would be an additional layer of redundancy which would prevent immediate poisoning of the model. In this experimental setup, the dataset is again split randomly for each node (the size of the dataset at each split is also random). We keep only one node clean, and the rest of the datasets on the other nodes are replaced with random values according to a normal distribution with zero mean and unit variance. Figure 6.8 shows learned scheduling for a student being trained on the MNIST dataset. We see how the scheduler begins by selecting a corrupted dataset, before quickly transitioning to selecting another corrupted dataset. The scheduler then selects a dataset with clean data and this selection dominates for the rest of training. In Figure 6.9, we see that due to the initial training on corrupted data, the test-set performance degrades. However, as soon as the scheduler learns to use the clean data, the performance improves rapidly. It can also be noticed that the performance achieved is below state-of-the-art. This is likely due to there being a much smaller diversity in training data due to node

corruption.

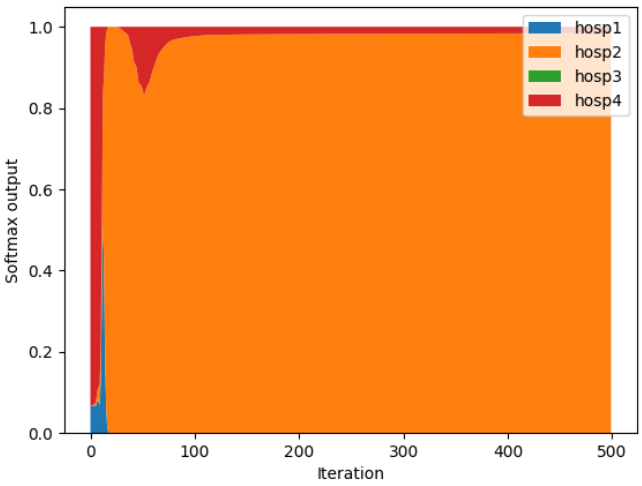


Figure 6.8: The scheduler selection for the different nodes in the federated system. We see that the scheduler learns to select the only clean dataset for training. The magnitudes of the different colours indicate the softmax output of the scheduler.

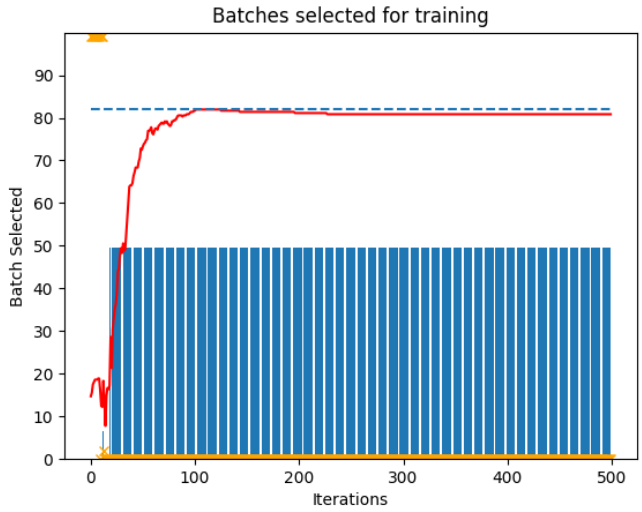


Figure 6.9: The teacher data selection as well as the performance of the student on the held-out test set for the MNIST digit recognition problem. The orange 'x' and blue bars represent the first and second outputs of the teacher respectively. The dashed line shows the maximum accuracy achieved. The red line is the performance of the student on the held-out test set.

Compromised Local Models at Nodes

The next experiment conducted is how our system trains when there are compromised local models (whether these are the student or the teacher). In the case of corrupting the teacher, the weights of a teacher at a local node are replaced by random samples from a normal distribution with zero mean and unit variance. Within this setup, the effect of having a compromised student or teacher is the same: a high loss which encourages the scheduler to change its selection. Figure 6.10 shows how the scheduler selects nodes from which to train. We see that, initially, nodes 1 and 4 are used to train before the scheduler attempts to use the poisoned teacher. After repeated reductions in the federated validation sets, the scheduler rapidly changes its selection, first favouring node 4, then, increasingly, node 3. With further training, we see the continuation of this rapid removal of training using node 1 whenever this node is encountered.

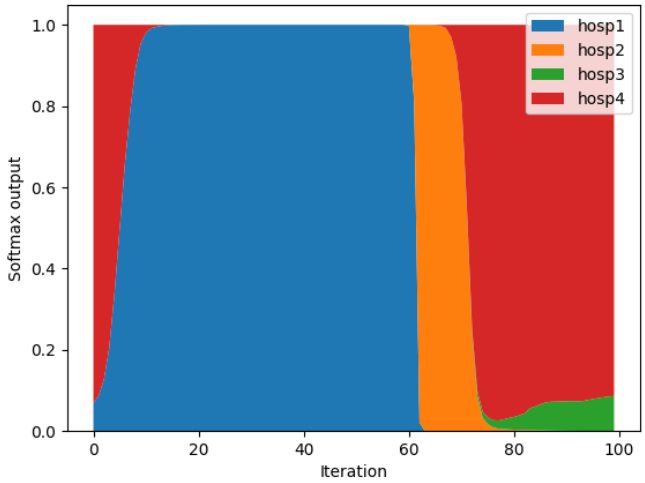


Figure 6.10: Scheduler selection for a four-node federated system. The hosp1 node (orange) corresponds to the poisoned teacher (randomised weights). We see that after brief selection, the scheduler reduces the contribution of this node. The student is being trained on the MNIST task.

Federated Learning for a Live Hospital Environment

To further assess the robustness of the proposed method, an experiment treating the hospital as a live environment is used. This is important because randomly splitting the data for each node may not be entirely characteristic of how real-world data from multiple hospital data centres would be distributed. There can be systematic differences due to the equipment that is used to take measurements, or the culture of measurement at the hospital, as well as downtime of machines meaning that recording is not possible. For this experiment, the data at two nodes are 'poisoned' in a systematic way that may mimic how data could be recorded in a real hospital environment. To achieve this, the data are separated into four nodes according to the time period during which the patient was admitted to the ED. These four nodes are patients admitted: (i) between midnight and 6 a.m., (ii) between 6 a.m. and noon, (iii) between noon and 6 p.m. and (iv) between 6 p.m. and midnight. The simulation mimics the scenario where during the morning hours, historic patient records are not available and the historic diagnosis code is therefore replaced by the generic historic diagnosis code corresponding to headache. Figure 6.11 shows the scheduler's selections for this setup. We observe that the scheduler begins by using the data from the morning but rapidly changes to using afternoon data with which it has success and therefore continues to train with. There is then a cycling between the use of afternoon and evening data while neglecting to use any data recorded during the morning. Once again, this performance is conditional on the validation set at each node being secure and unchanged. As a result, this indicates that some curation is required by a user for the validation sets to ensure that these are accurate. However, even with less curation of the training set, the scheduler may still be allowed to choose the centres from which it will gain greatest benefit for the student.

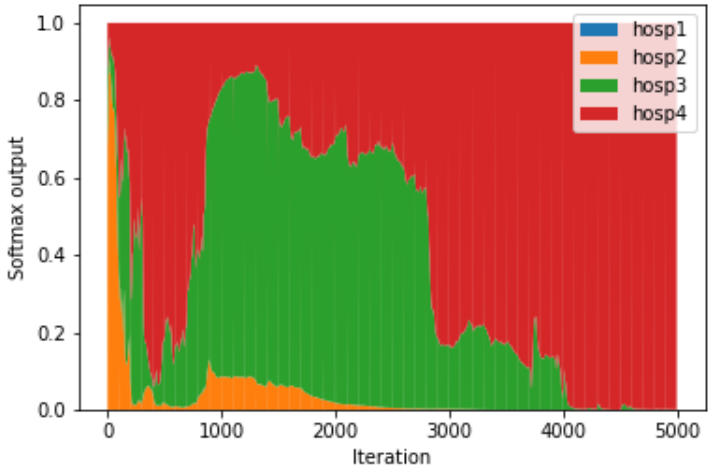


Figure 6.11: Scheduler selection in a system where data has been systematically poisoned according to the time it was recorded in order to simulate a database outage. Hosp 1 (node i) and 2 (node ii) represent the patients admitted in the morning hours, and 3 (node iii) and 4 (node iv) the afternoon and evening admissions respectively.

A Federation of Similar Teachers

This experiment investigated how a federation of similar teachers interact in order to train a student. In this approach, the ward admission prediction problem was split into separate prediction problems. Instead of having one, seven-class, classification, the design was modified so that each class would be a distinct binary classification, thus creating seven separate classifiers. A teacher was trained for each classification problem with the same feature space that was used for the seven-class classification and saved. The federated system was then split into seven nodes with the training set split across the seven nodes. The training set was still in the seven-class format, the binary classification format was simply to produce teachers capable of training to classify each class. Figure 6.12 shows how the scheduler selected data centres and the associated teachers.

We observe that node 6 (corresponding to the surgical admissions teacher) dominates initially before node 3 (neuro admissions) takes precedence. These then give way to node 4 (trauma admissions). Figure 6.13 shows the overall unbalanced class distribution for the training data set.

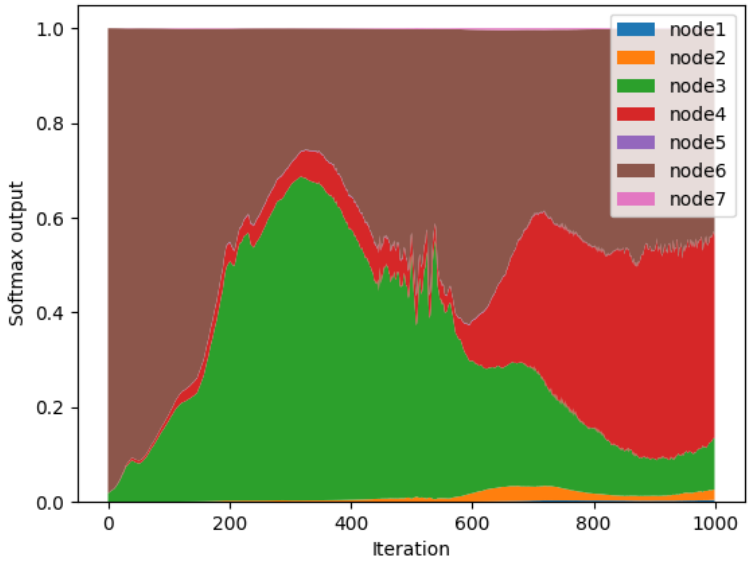


Figure 6.12: Scheduler selections for the similar teachers setup. We see that the teachers corresponding to teaching on underrepresented tasks are prioritised first. Presence of the more predominant classes in all of the nodes indicates that learning overall covers the range of classes.

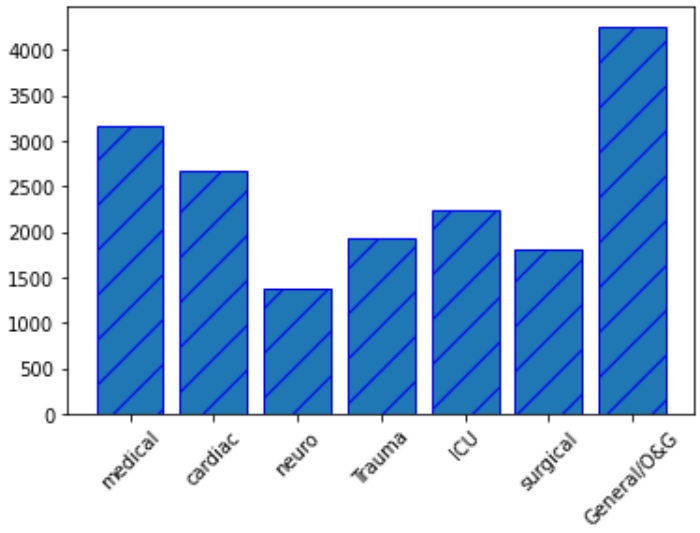


Figure 6.13: The total unbalanced class distribution through the training set. The validation set and test sets have been kept according to this unbalanced distribution.

It can be seen that the scheduler has learned to sample the teachers that train according to the most underrepresented classes in the dataset. This is due to the validation and test sets being kept to the same class distribution as that shown in

Figure 6.13. It should be noted that the data at each node is not split according to class, only the teacher is class specific. As a result, there is sufficient representation of the more prevalent classes and the selection of the underrepresented class teachers ensures that these classes are learned as well as the others.

Transfer Learning

Following the use of similar teachers interacting with the scheduler, we see that there is potential for exploiting transfer learning to train students on tasks for which there are no trained teachers. This is because the teachers trained in the previous section were all trained on different tasks to the one that they were selected for by the scheduler. We therefore hypothesise that through exploiting the various teacher's skills in succession, we can provide the gradients that the student needs in order to master the unseen task. This is similar to the approach used in the previous chapter, where a teacher taught on a similar task was shown to be able to provide a good strategy to train a student. Here, the scheduler is capable of changing its selection based on performance of the student in a real-time fashion, the scheduler may be able to select the teachers that would sample and provide gradients to the student that would be most advantageous to the final performance.

6.6.1 Clinical Utility

It is believed that this federated style of learning can provide clinicians with an opportunity to train their models without needing explicit access to another hospital's dataset. The only human input required would be that which ensures that the training sets at each node in the system are formatted in the same way.

This method could also be helpful to clinicians as it is capable of sampling data without the need for human input. This means that erroneous data will not be learned from, and so will not require large efforts in the analysis and discarding of data that seems to the user to be out of distribution. This, however, relies on having a

large enough dataset that enough 'accurate' data is present. It also requires that the validation sets at each site are curated and investigated by the users and kept in a secure location to avoid poisoning or attack.

6.7 Discussion and Conclusion

In this work we have shown that using a federated system, with teachers at the nodes that select the data to provide gradient updates to a central model, can achieve state-of-the-art performance as well as protect the privacy of patients. We have further shown that the setup provides some protection against attacks on data stored at each node or the local models being used at each node.

However, there remain some challenges associated with this approach that need to be addressed in order for it to become practicable. The first is that the training of the centralised model (student) is inherently unstable due to the continual training that this setup expects. It is expected that the student performance will converge with training, however should data poisoning occur, the scheduler will need to continue a few iterations of training in order to recover a well-performing model. Presenting the problem in this way also provides flexibility for growing datasets at each node. To enable this, taking the straightforward process re-sorting the curriculum at each node would be the only requirement in order that the scheduler and teachers could be used as before. It is possible however that should all validation set performances exceed a certain defined threshold of performance, a flag can be raised to stop model training.

Another limitation is the need for centralised control. It is important in this setup that all of the hospitals communicate their responses to the central node for actions to be taken by the scheduler. In the case of large institutions such as hospitals, this may be acceptable, but is unlikely to be so for faster-paced learning environments such as learning from mobile phones, where interruptions to communication can be frequent. However, upon re-connection to the federated system, any reductions in performance

to the whole system will be used as signals to improve the scheduler selection, and with training the performance should recover.

Furthermore, in order to ensure diversity in selections by the scheduler we introduced the entropy loss that discouraged convergence on one selection. This may be a naive way of encouraging diversity in selection and we believe that there may be better additive losses and regularisation terms that can be used to design a loss function that will serve the purposes of the scheduler better.

For further protection against attack, sentry agents (much like the teachers) could also be trained to detect any anomalies or designed attacks within the batches selected by the teachers, before the losses are passed onto the central node. This would reduce the burden of scanning the entire dataset at the node before training.

It is possible however that this setup can be used to continually learn even with distributional shift with time. Should a small curated validation dataset be maintained by the local hospitals including the latest data from the hospital, the scheduler's losses will be derived from these new datasets instead. As a result the scheduler will learn to sample in a fashion that would allow the student to maximise its performance on these validation sets. The training sets would also need to be updated with new information, however these would not need to be curated with the same precision as the validation sets.

To conclude, we believe we have presented a promising direction for federated learning between large institutions such as hospitals. With further work, we believe that we can develop this into a robust system that can continually learn from growing datasets while maintaining a state-of-the-art performance for the improvement of hospital patient flow.

Chapter 7

Conclusion

Patient flow has been recognised as a key bottleneck that must be tackled for improved patient throughput in hospitals. The greater rate of inflow of patients than outflow renders bed space and resource increasingly rare commodities and so planning is essential to make more efficient use of them. While this thesis has focused on the John Radcliffe Hospital in the Oxford University Hospitals NHS Trust, this is part of a wider trend across the UK. The methods developed in this thesis contribute towards these efforts of more efficient use of hospital resource.

7.1 Summary of Thesis

In Chapter 1 we introduced the concept of patient flow and described the effect that poor patient flow has on hospitals. We presented the typical structure of a hospital and how emergency admissions are processed. We formally defined patient flow and saw that the definition is comprised of two components; The first is related to the need to move patients in an efficient manner with respect to resource. The second is that the quality of care and satisfaction of the patient must be maintained. We discussed the need to create a predictive rather than a reactive system to manage patient flow, as this was beneficial for both resource utilisation as well as for the care of the patient.

In Chapter 2 we reviewed the literature surrounding the use of prediction in patient flow. We saw that patient flow can be split into four distinct subsections: admissions to the ED, movement from the ED to inpatient admission (termed crossing the ED-inpatient interface or EDii), movement of inpatients or resource within the hospital, and discharge from hospital. We saw that there is a rich literature across the spectrum of these subcategories but with little uniting them. Our first contribution in this thesis was to create a summary of the work published in using machine learning for patient flow and mapping the landscape of work completed in this field. We also identified that predicting what type of ward a patient would be admitted to in hospital would be very useful clinically for resource allocation. We therefore chose to tackle this problem for the remainder of the thesis.

In Chapter 3 we analysed the IORD dataset that was extracted from the Oxford University Hospitals NHS Foundation Trust. We analysed the data according to the four subcategories of patient flow that we had identified in the previous chapter. We showed that there was sufficient data to carry out the prediction of location of admission to hospital from the ED and that this problem was the most actionable with accurate predictions. We then went on to show how the training dataset that was used for the remainder of the thesis was constructed. A future extract of the IORD dataset should be one that incorporates the time-series streams of the patient data. This will allow a time-series model to be created so that a richer input can be used for prediction. An important question that will arise from this is how the hospital will handle a prediction that may change with more data being available as the time-series unfolds. Something that may help with this is investigation of confidence measures for the network outputs. An output from the network that ranks the potential next ward and confidence in this prediction may give clinicians more actionable information rather than a discrete prediction.

In Chapter 4 we introduced the concept of curriculum learning for gradient based learning models. This was based on the Mahalanobis curriculum that was also

introduced in this chapter. We demonstrated through experiments that sampling the data according to a specific ordering was beneficial for the test-time performance. A multi-armed bandit was then also implemented in order to allow the model to sample according to an order that was not pre-specified by the user which we found also provided improvements in the performance. In addition, we introduced a saliency term to provide some interpretability to the predictions being made. It was found that a large learning signal came from the tests requested by the triage nurse, as well as the previous diagnosis code that the patient may have. In order to further test this model, we investigated its performance throughout the year (as the distribution for the reason for admission is not stationary with respect to time of year), and found that the model performed consistently across the span of four years. Something that may be useful to clinicians is a temporal model, one where different extracts of the dataset are used for training according to time of year. This may help with having a more accurate model but will invevitably introduce biases to the model. A more useful approach may be in carrying out representation learning of ED admission patient types using some form of metric or contrastive learning. These methods encourage the embedding of similar patients in similar locations in the latent space of the network. However, once again this can be biased by various features and so investigation would be required to find the best way of making patient comparisons prior to embedding them in a similar location.

In Chapter 5 we expanded this approach of sampling the data by introducing this now as a reinforcement learning problem. We introduced the student-teacher setup, whereby a student (predicting the location of admission) is trained by a reinforcement learning agent. In expanding our approach in this way, the data sampling became conditioned on the current state of the student (whereas with the multi-armed bandit the sampling was retrospective) and this led to a much stronger performance of the student at test-time. An investigation of both discrete and continuous reinforcement learning algorithms was carried out in order to provide the teacher with maximal

flexibility of selection when teaching the student. It was found that for different problems, different strategies were implemented by the teacher, meaning that heuristic strategies of training for problems could be learned from the teacher's behaviours. An area for further exploration is in the utilisation of multiple teachers at the same time and combining their selected batches as one batch to be used for training. Another important investigation would be in using the student-teacher setup to train a reinforcement learning agent. In this case it would be important to sample through the experience replay buffer of the student in order to present batches for the student to learn from.

In Chapter 6 a federated learning approach of learning from multiple private datasets was introduced. This provided a much larger combined dataset to learn from without sacrificing the privacy of the datasets. We showed how the scheduler can schedule which hospital to learn from and works in conjunction with the teacher at each node. This allowed for the sampling of data which, when trained on, improved the overall performance of the student across all validation sets. Furthermore, we demonstrated how if there are systematic inaccuracies in the data that had been recorded at one site, or active poisoning by an attack, the scheduler learned to stop using this node. This was achieved through the feedback that the scheduler received, namely the meta-gradient based on the loss of the student at each validation set at each node. This not only allowed us to exploit student-teacher learning to maximise performance, but also built robustness and defence into the training setup by allowing a selectiveness over which data could be sampled. Further investigations should allow for sampling from multiple nodes at the same time as well as the mixing of teachers to produce a hybrid teacher.

7.2 Limitations and Future Work

The methods proposed in this thesis are not without their limitations. Throughout this thesis we have attempted to present a balanced argument for the methods proposed, while also highlighting the limitations associated with each. We here summarise the limitations and future work that could be conducted to further progress these methods and aid in providing hospitals with more efficient patient flow.

The first limitation is that the model we have introduced is a static one. This was purposefully designed with the intent of maximising the performance of the predictor based on the first information available at triage, so that there would be as much time as possible to prepare bed space if it was needed. However, it could be more useful using a time-series of information gathered as the patient waits in the ED. It is likely that conditions within the hospital change as the patient waits in the ED and a time-series model to monitor this information would probably provide a more accurate idea of the location of admission. This is due to the richer context that a time series provides than a static snapshot. A future model should incorporate some form of recurrency to capture these patterns whether through a recurrent model or an attention-like model such as a transformer to learn the time dependencies between the data in the time-series. One of the challenges that may be faced with this approach is how it should be implemented. An online model being used in that hospital that is capable of changing predictions with more information being made available can cause scheduling issues with bed preparation. Therefore it is not necessarily clear that a more accurate time-series model would be beneficial in such a case where a decision must be made and stuck with. However it is possible that the time-stamps could be limited such that any information available after a certain threshold of waiting cannot be used to mimic taking the decision to prepare bed space. Future work could also investigate the use of a model that could quantify uncertainty in the predictions made. A future project should re-frame the ward admission as a time-series, limit the

time-series so that it does not span up to the time of admission and extract a new feature set from the IORD database, where the features extracted for this thesis were done to summarise as much information for a static model as possible.

It should also be noted that patients are admitted to hospital not only based on their needs but also on the location of specialised equipment that may be needed to assess their condition. As a result, future work could make use of this information to predict if specialised scans or measurements are required and further refine the prediction. This would require the integration of the hospital machine maintenance database to understand the use and locations of each of these specialised machines. Therefore a future project should focus on the integration of the maintenance database with the IORD database and alignment with which patients used the machines and the corresponding timestamps.

As part of the preprocessing of the dataset, one of our criteria was the removal of patients who had "obvious" admissions. An example is removal of patients who had an ICD-10 code associated with a heart attack who were admitted to the cardiac wards immediately. However, this is a somewhat naive and heuristic method of filtering out the "easy" cases, and a more systematic way of carrying out this filtering should be investigated. Future work could investigate if there are any other data that could be used to infer admission to a specific ward, other than the diagnosis code that was subsequently assigned to the patient. A future project should also include a method of encoding the patients diagnosis history. Having a way of summarising this information will provide much greater context for the machine learning algorithm to classify their likely admission location.

This work was carried out with the assumption that a full feature set of patients should be available upon triage. However, it is common for there to be missing data in hospital admissions. As a result, future work could focus on the creation of a method that can model the joint distribution over the features of the patients. Should data be unavailable, the missing elements of the patient feature set can be sampled

from the joint distribution by conditioning on the available data and then sampling. Furthermore, the saliency method presented is not a 'causal' mechanism that explains why the predictions were made, only that there are correlations between those features selected being weighted larger than others. A future project could investigate the causal nature of the predictions being made in this model, being able to generate any missing features and expanding the prediction from admission location to the first two admission locations. It is possible that in expanding the model gradually in this way, a fuller picture of the likely resources that will need to be utilised in care for the patient will be captured.

As discussed in Chapter 5, the student-teacher model can be limited in its use due to the representation of the students. Due to a representation of the weights being used, the larger the student grows, the larger the input to the teacher becomes. This then requires significantly more data to be generated in order to train the teacher. While this is possible, it is not efficient and is unlikely to be desirable for a training methodology. As a result, we believe future work should focus on more compact and efficient representations of the student. There is also more information that could be used to represent the student, such as its learning rate and the activations used at the nodes. However representations of these can be difficult to encode particularly for activation functions. An alternative could investigate the use of model distillation for large students to be represented by smaller ones, and using the distilled model with the student-teacher methodology. This is generally done using a mimicking setup where the loss function of the student is to mimick the outputs of the teacher for the same inputs in order to replicate the behaviour of the teacher. There are some issues with this approach in that, a distilled model is trained simply to mimic the behaviour of a larger model for the same problem. Whereas the larger model has extracted features from the data in order to perform well on the task, the distilled model potentially memorises the input-output pairs. A future project should consider this investigation, and integrate a form of model distillation in the full training loop of the

student-teacher setup outlined in this thesis. This would incorporate the distillation loss in the joint training routine so that the small student can replicate the outputs of the larger trained model and the student-teacher setup can be used with that model.

Another limitation of the student-teacher method is in the design of the output heads. In the setup proposed only two heads were allowed; one of the index of a datum, and another to select a batch around this datum. Future work could investigate the incorporation of multiple double-headed outputs. This would allow for the selection of data from opposite ends of the entropy spectrum within the same batch, and may provide significantly more flexible training.

A further project to be investigated is in creating a generative student-teacher setup. With the continuous agent, any value along the batch index can be selected, including non-integer values. Having a generative model that is capable of generating the batch according to the selection of the teacher would allow us to present a batch to the student at exactly the entropy that was selected as opposed to selecting the closest existing entry. This could be done through training a separate variational autoencoder that reconstructs the batches according to their Mahalanobis distance (which would allow us to approximate a Gaussian in the variational model to generate the batches) or through the use of meta-gradients. The reward used for the teacher could also be fed back as a signal for the generative model in order to train it.

One of the limitations of the federated approach proposed in this thesis is the requirement for continual updates. Whilst we have argued that this provides greater robustness due to erroneous or poisoned data that may be at the nodes, it also requires more time for training. As a result, should poisoning occur, it may take many iterations of training before the scheduler stops using the node for training (as seen in sub-section 6.10, where twenty training updates were required before the poisoned node was dropped).

There is also the issue of messages from all nodes being required prior to updating the scheduler. In real systems, there can often be interruptions in communication. As a

result, future work could investigate whether the scheduler can maintain performance of the student by ignoring a message from a node if it takes too long to arrive. Alternatively, investigations into other ways of updating the scheduler could be investigated that do not rely on constant communication.

Furthermore, the additive loss that was used in the scheduler's loss function, the entropy loss, could be improved upon. This loss simply encourages greater diversity in selection. However, designing an additive loss that is based on the task that is being solved could be more useful, as opposed to simply encouraging diversity in selection.

It is hoped that this thesis may contribute to improving the efficiency with which healthcare is delivered. In doing so, we hope that it will help not only to improve expenditure in hospitals but also to improve the quality of patient care.

7.2.1 Future Projects

Integration of Discharge Prediction

Throughout this thesis we have discussed how resource can be predicted upon admission of patients to the ED. We have also seen in the literature review how discharge rates have a great impact on the resource available to treat patients in hospitals. As a result, one of the future projects that should be investigated is the integration of a discharge predictor alongside the admission predictor that was presented in this thesis. Access to this information will allow hospital staff to plan how many patients are likely to leave the hospital and how many are likely to be admitted. This provides staff with more flexibility in terms of admission options than making the assumption that the patient stays within the hospital will remain constant. This is also a step towards introducing a dynamic model that can be used to predict real-time patient flux.

Simulation of the Trust

After implementing and trialling the predictions in hospital, we propose the expansion of these models to the Trust level. This would incorporate all five hospitals that constitute the Oxford University NHS Hospitals Trust. We envision a dynamic model for each of these hospitals, with an admission location predictor, as well as a discharge predictor. These predictors could be trained using the local data from the hospital or through some form of transfer learning and fine-tuning with local data. These would be implemented in a way so that the real-time flux of the patients could be monitored. This would create a digital twin of the Trust, allowing for monitoring as well as prediction of resource utilisation across all of the individual hospitals. Simulations could then be used to assess hypothetical scenarios and help management plan for resource utilisation or staffing rotas. It may also help with the transferal of patients to other hospitals in the Trust when certain hospitals are overburdened. We envision then aggregating the outputs from each hospital to represent the requirements of each Trust, creating a hierarchical model whereby each Trust is treated as a node as opposed to each hospital. Such an approach could also help for budgeting and planning at a higher level than the day-to-day running of the hospital. With incorporation of the suggested future work, a flexible and robust tool could be developed in order to aid hospital staff with their daily (and potentially longer-term) planning to improve patient flow.

References

[1] L. S. Rosenfeld, F. Goldmann, and L. A. Kaprio, "Reasons for prolonged hospital stay: A study of need for hospital care," *Journal of Chronic Diseases*, vol. 6, no. 2, pp. 141–152, 1957.

[2] P. J. Feldstein and R. M. Severson, *The demand for medical care*. 1964.

[3] J. Coid and P. Crome, "Bed blocking in bromley.," *British Medical Journal (Clin Res Ed)*, vol. 292, no. 6530, pp. 1253–1256, 1986.

[4] A. E. Roth, *The Shapley value: essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.

[5] G. Laffel and D. Blumenthal, "The case for using industrial quality management science in health care organizations," *Journal of the American Medical Association*, vol. 262, no. 20, pp. 2869–2873, 1989.

[6] D. M. Cutler, J. M. Poterba, L. M. Sheiner, L. H. Summers, and G. A. Akerlof, "An aging society: Opportunity or challenge?" *Brookings papers on economic activity*, vol. 1990, no. 1, pp. 1–73, 1990.

[7] B. D. McCarthy, J. B. Wong, and H. P. Selker, "Detecting acute cardiac ischemia in the emergency department," *Journal of General Internal Medicine*, vol. 5, no. 4, pp. 365–373, 1990.

[8] M. Gori and A. Tesi, "On the problem of local minima in backpropagation," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 1, pp. 76–86, 1992.

[9] J. R. Quinlan, "Combining instance-based and model-based learning," in *Proceedings of the tenth international conference on machine learning*, 1993, pp. 236–243.

[10] K. Styrborn and M. Thorslund, "Bed-blockers: Delayed discharge of hospital patients in a nationwide perspective in sweden," *Health Policy*, vol. 26, no. 2, pp. 155–170, 1993.

[11] D. Tandberg and C. Qualls, "Time series forecasts of emergency department patient volume, length of stay, and acuity," *Annals of Emergency Medicine*, vol. 23, no. 2, pp. 299–306, 1994.

[12] D. A. Thompson, P. R. Yarnold, D. R. Williams, and S. L. Adams, "Effects of actual waiting time, perceived waiting time, information delivery, and expressive quality on patient satisfaction in the emergency department," *Annals of Emergency Medicine*, vol. 28, no. 6, pp. 657–665, 1996.

[13] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.

[14] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications*, vol. 13, no. 4, pp. 18–28, 1998.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[16] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*. MIT Press Cambridge, 1998, vol. 135.

[17] C. Gaskett, D. Wettergreen, and A. Zelinsky, "Q-learning in continuous state and action spaces," in *Australasian Joint Conference on Artificial Intelligence*, Springer, 1999, pp. 417–428.

[18] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The mahalanobis distance," *Chemometrics and Intelligent Laboratory Systems*, vol. 50, no. 1, pp. 1–18, 2000.

[19] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, e215–e220, 2000.

[20] J. Oliveira-Filho, A. R. Massaro, F. Yamamoto, L. Bustamante, and M. Scaff, "Stroke as the first manifestation of calcific aortic stenosis," *Cerebrovascular Diseases*, vol. 10, no. 5, pp. 413–416, 2000.

[21] H. Batal, J. Tench, S. McMillan, J. Adams, and P. S. Mehler, "Predicting patient visits to an urgent care clinic using calendar variables," *Academic Emergency Medicine*, vol. 8, no. 1, pp. 48–53, 2001.

[22] G. A. Skowronski, "Bed rationing and allocation in the intensive care unit," *Current Opinion in Critical Care*, vol. 7, no. 6, pp. 480–484, 2001.

[23] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The nonstochastic multiarmed bandit problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002.

[24] O. Miro, M. Sanchez, G. Espinosa, B. Coll-Vinent, E. Bragulat, and J. Milla, "Analysis of patient flow in the emergency department and the effect of an extensive reorganisation," *Emergency Medicine Journal*, vol. 20, no. 2, pp. 143–148, 2003.

[25] S. Stewart, K. MacIntyre, S. Capewell, and J. McMurray, "Heart failure and the aging population: An increasing burden in the 21st century?" *Heart*, vol. 89, no. 1, pp. 49–53, 2003.

[26] A. Xavier, "Hospital competition, gp fundholders and waiting times in the uk internal market: The case of elective surgery," *International Journal of Health Care Finance and Economics*, vol. 3, no. 1, pp. 25–51, 2003.

[27] B. Bucheli and B. Martina, "Reduced length of stay in medical emergency department patients: A prospective controlled study on emergency physician staffing," *European Journal of Emergency Medicine*, vol. 11, no. 1, pp. 29–34, 2004.

[28] S. Stevens, "Reform strategies for the english nhs," *Health Affairs*, vol. 23, no. 3, pp. 37–44, 2004.

[29] P. Walsh, S. J. Rothenberg, S. O'Doherty, H. Hoey, and R. Healy, "A validated clinical model to predict the need for admission and length of stay in children with acute bronchiolitis," *European Journal of Emergency Medicine*, vol. 11, no. 5, pp. 265–272, 2004.

[30] M. D. Bayley, J. S. Schwartz, F. S. Shofer, M. Weiner, F. D. Sites, K. B. Traber, and J. E. Hollander, "The financial burden of emergency department congestion and hospital crowding for chest pain patients awaiting admission," *Annals of Emergency Medicine*, vol. 45, no. 2, pp. 110–117, 2005.

[31] J. Leegon, I. Jones, K. Lanaghan, and D. Aronsky, "Predicting hospital admission for emergency department patients using a bayesian network," in *AMIA Annual Symposium Proceedings*, American Medical Informatics Association, vol. 2005, 2005, p. 1022.

[32] S. Saint Lamont, "See and treat: Spreading like wildfire? a qualitative study into factors affecting its introduction and spread," *Emergency Medicine Journal*, vol. 22, no. 8, pp. 548–552, 2005.

[33] J. Vissers and R. Beech, *Patient Flow Logistics in Health Care*. Psychology Press, 2005.

[34] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[35] K. J. Brasel, H. J. Lim, R. Nirula, and J. A. Weigelt, "Length of stay: An appropriate quality measure?" *Archives of Surgery*, vol. 142, no. 5, pp. 461–466, 2007.

[36] A. M. De Bruin, A. Van Rossum, M. Visser, and G. Koole, "Modeling the emergency cardiac in-patient flow: An application of queuing theory," *Health Care Management Science*, vol. 10, no. 2, pp. 125–137, 2007.

[37] P. Thungjaroenkul, G. G. Cummings, and A. Embleton, "The impact of nurse staffing on hospital costs and patient length of stay: A systematic review," *Nursing Economics*, vol. 25, no. 5, p. 255, 2007.

[38] J.-Y. Yeh and W.-S. Lin, "Using simulation technique and genetic algorithm to improve the quality care of a hospital emergency department," *Expert Systems with Applications*, vol. 32, no. 4, pp. 1073–1083, 2007.

[39] A. Börsch-Supan, *First results from the Survey of Health, Ageing and Retirement in Europe (2004-2007): Starting the longitudinal dimension*. Univerza v Ljubljani, Ekonomska fakulteta, 2008.

[40] J. Boyle, M. Wallis, M. Jessup, J. Crilly, J. Lind, P. Miller, and G. Fitzgerald, "Regression forecasting of patient admission data," in *2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, IEEE, 2008, pp. 3819–3822.

[41] T. Rotter, J. Kugler, R. Koch, H. Gothe, S. Twork, J. M. van Oostrum, and E. W. Steyerberg, "A systematic review and meta-analysis of the effects of clinical pathways on length of stay, hospital costs and patient outcomes," *BioMed Central Health Services Research*, vol. 8, no. 1, p. 265, 2008.

[42] L. van der Maaten and G. Hinton, "Visualizing high-dimensional data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[43] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, ACM, 2009, pp. 41–48.

[44] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[45] L. M. Schweigler, J. S. Desmond, M. L. McCarthy, K. J. Bukowski, E. L. Ionides, and J. G. Younger, "Forecasting models of emergency department crowding," *Academic Emergency Medicine*, vol. 16, no. 4, pp. 301–308, 2009.

[46] J.-S. Tancrez, B. Roland, J.-P. Cordier, and F. Riane, "How stochasticity and emergencies disrupt the surgical schedule," in *Intelligent Patient Management*, Springer, 2009, pp. 221–239.

[47] S. Au-Yeung, U. Harder, E. McCoy, and W. Knottenbelt, "Predicting patient arrivals to an accident and emergency department," *Emergency Medicine Journal*, vol. 26, no. 4, pp. 241–244, 2009.

[48] V. M. Arora, M. L. Prochaska, J. M. Farnan, M. J. D'Arcy V, K. J. Schwanz, L. M. Vinci, A. M. Davis, D. O. Meltzer, and J. K. Johnson, "Problems after discharge and understanding of communication with their primary care physicians among hospitalized seniors: A mixed methods study," *Journal of Hospital Medicine*, vol. 5, no. 7, pp. 385–391, 2010.

[49] R. Ding, M. L. McCarthy, J. S. Desmond, J. S. Lee, D. Aronsky, and S. L. Zeger, "Characterizing waiting room time, treatment time, and boarding time in the emergency department using quantile regression," *Academic Emergency Medicine*, vol. 17, no. 8, pp. 813–823, 2010.

[50] M. A. LaMantia, T. F. Platts-Mills, K. Biese, C. Khandelwal, C. Forbach, C. B. Cairns, J. Busby-Whitehead, and J. S. Kizer, "Predicting hospital admission and returns to the emergency department for elderly patients," *Academic Emergency Medicine*, vol. 17, no. 3, pp. 252–259, 2010.

[51] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: `http://yann.lecun.com/exdb/mnist/`.

[52] T. Rotter, L. Kinsman, E. L. James, A. Machotta, H. Gothe, J. Willis, P. Snow, and J. Kugler, "Clinical pathways: Effects on professional practice, patient outcomes, length of stay and hospital costs," *Cochrane Database of Systematic Reviews*, no. 3, 2010.

[53] I. Beardsell and S. Robinson, "Can emergency department nurses performing triage predict the need for admission?" *Emergency Medicine Journal*, vol. 28, no. 11, pp. 959–962, 2011.

[54] J. Gittins, K. Glazebrook, and R. Weber, *Multi-armed bandit allocation indices*. John Wiley & Sons, 2011.

[55] A. Guttmann, M. J. Schull, M. J. Vermeulen, and T. A. Stukel, "Association between waiting times and short term mortality and hospital admission after departure from emergency department: Population based cohort study from ontario, canada," *British Medical Journal*, vol. 342, 2011.

[56] M. Marlais, J. Evans, and E. Abrahamson, "Clinical predictors of admission in infants with acute bronchiolitis," *Archives of Disease in Childhood*, vol. 96, no. 7, pp. 648–652, 2011.

[57] S. Oredsson, H. Jonsson, J. Rognes, L. Lind, K. E. Göransson, A. Ehrenberg, K. Asplund, M. Castrén, and N. Farrohknia, "A systematic review of triage-related interventions to improve patient flow in emergency departments," *Scandinavian Journal of Trauma, Resuscitation and Emergency Medicine*, vol. 19, no. 1, p. 43, 2011.

[58] N. Parsons, M. Odumenya, A. Edwards, F. Lecky, and G. Pattison, "Modelling the effects of the weather on admissions to uk trauma units: A cross-sectional study," *Emergency Medicine Journal*, vol. 28, no. 10, pp. 851–855, 2011.

[59] Y. Sun, B. H. Heng, S. Y. Tay, and E. Seow, "Predicting hospital admissions at emergency department triage using routine administrative data," *Academic Emergency Medicine*, vol. 18, no. 8, pp. 844–850, 2011.

[60] S. Tanuja, D. U. Acharya, and K. Shailesh, "Comparison of different data mining techniques to predict hospital length of stay," *Journal of Pharmaceutical and Biomedical Sciences*, vol. 7, no. 7, 2011.

[61] E. L. Allgower and K. Georg, *Numerical continuation methods: an introduction*. Springer Science & Business Media, 2012, vol. 13.

[62] J. Boyle, M. Jessup, J. Crilly, D. Green, J. Lind, M. Wallis, P. Miller, and G. Fitzgerald, "Predicting emergency department admissions," *Emergency Medicine Journal*, vol. 29, no. 5, pp. 358–365, 2012.

[63] O. Karan, C. Bayraktar, H. Gümüşkaya, and B. Karlık, "Diagnosing diabetes using neural networks on small mobile devices," *Expert Systems with Applications*, vol. 39, no. 1, pp. 54–60, 2012.

[64] M. Kelly, L. Sharp, F. Dwane, T. Kelleher, and H. Comber, "Factors predicting hospital length-of-stay and readmission after colorectal resection: A population-based study of elective and emergency admissions," *BioMed Central Health Services Research*, vol. 12, no. 1, p. 77, 2012.

[65] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT Press, 2012.

[66] N. Mustafee, T. Lyons, P. Rees, L. Davies, M. Ramsey, and M. D. Williams, "Planning of bed capacities in specialized and integrated care units: Incorporating bed blockers in a simulation of surgical throughput," in *Proceedings of the 2012 Winter Simulation Conference (WSC)*, IEEE, 2012, pp. 1–12.

[67] M. E. H. Ong, C. H. L. Ng, K. Goh, N. Liu, Z. X. Koh, N. Shahidah, T. T. Zhang, S. Fook-Chong, and Z. Lin, "Prediction of cardiac arrest in critically ill patients presenting to the emergency department using a machine learning

score incorporating heart rate variability compared with the modified early warning score," *Critical Care*, vol. 16, no. 3, R108, 2012.

[68] G. A. Seber and A. J. Lee, *Linear regression analysis*. John Wiley & Sons, 2012, vol. 329.

[69] M. Wiering and M. Van Otterlo, "Reinforcement learning," *Adaptation, Learning, and Optimization*, vol. 12, p. 3, 2012.

[70] A. Arisha and W. Abo-Hamad, "Towards operations excellence: Optimising staff scheduling for new emergency department," in *Proceedings of the 20th International Annual EurOMA Conference-" Operations Management at the Heart of the Recovery*, vol. 9, 2013, p. 12.

[71] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[72] M. Blyth, "The austerity delusion: Why a bad idea won over the west," *Foreign Aff.*, vol. 92, p. 41, 2013.

[73] S. Dimakou, "Waiting time distributions and national targets for elective surgery in uk: Theoretical modelling and duration analysis," Ph.D. dissertation, City University London, 2013.

[74] R. Francis, *Report of the Mid Staffordshire NHS Foundation Trust public inquiry: executive summary*. The Stationery Office, 2013, vol. 947.

[75] R. Hall, "Patient flow," *AMC*, vol. 10, p. 12, 2013.

[76] A. Hosseinzadeh, M. T. Izadi, A. Verma, D. Precup, and D. L. Buckeridge, "Assessing the predictability of hospital readmission using machine learning.," in *Innovative Applications of Artifical Intelligence*, 2013.

[77] J. L. Krichmar and F. Röhrbein, "Value and reward based learning in neurorobots," *Frontiers in Neurorobotics*, vol. 7, p. 13, 2013.

[78] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International Conference on Machine Learning*, 2013, pp. 1139–1147.

[79] C. Combes, F. Kadri, and S. Chaabane, "Predicting hospital length of stay using regression models: Application to emergency department," 2014.

[80] V. Gravey, "Austerity at eu level?" *Political Perspectives*, vol. 8, no. 2, p. 4, 2014.

[81] A. L. Leppin, M. R. Gionfriddo, M. Kessler, J. P. Brito, F. S. Mair, K. Gallacher, Z. Wang, P. J. Erwin, T. Sylvester, K. Boehmer, *et al.*, "Preventing 30-day hospital readmissions: A systematic review and meta-analysis of randomized trials," *Journal of the American Medical Association Internal Medicine*, vol. 174, no. 7, pp. 1095–1107, 2014.

[82] Z. Liang, G. Zhang, J. X. Huang, and Q. V. Hu, "Deep learning for healthcare decision making with emrs," in *2014 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, IEEE, 2014, pp. 556–559.

[83] P. C. Pendharkar and H. Khurana, "Machine learning techniques for predicting hospital length of stay in pennsylvania federal and specialty hospitals.," *International Journal of Computer Science & Applications*, vol. 11, no. 3, 2014.

[84] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[85] W. Zaremba and I. Sutskever, "Learning to execute," *arXiv preprint arXiv:1410.4615*, 2014.

[86] H. Zhai, P. Brady, Q. Li, T. Lingren, Y. Ni, D. S. Wheeler, and I. Solti, "Developing and evaluating a machine learning based algorithm to predict the need of pediatric intensive care unit transfer for newly hospitalized children," *Resuscitation*, vol. 85, no. 8, pp. 1065–1071, 2014.

[87] A. Azari, V. P. Janeja, and S. Levin, "Imbalanced learning to predict long stay emergency department patients," in *2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, IEEE, 2015, pp. 807–814.

[88] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," *arXiv preprint arXiv:1506.03099*, 2015.

[89] D. A. Clifton, K. Niehaus, P. Charlton, and G. Colopy, "Health informatics via machine learning for the clinical management of patients," *Yearbook of Medical Informatics*, vol. 10, no. 1, p. 38, 2015.

[90] N. England, "The nhs constitution for england," *London: Department of Health*, 2015.

[91] M. Gül and A. F. Güneri, "Forecasting patient length of stay in an emergency department by artificial neural networks," *Journal of Aeronautics and Space Technologies*, vol. 8, no. 2, pp. 43–48, 2015.

[92] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[93] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[94] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[95] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[96] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh, "Action-conditional video prediction using deep networks in atari games," in *Advances in Neural Information Processing Systems*, 2015, pp. 2863–2871.

[97] J. P. Papay and M. A. Kraft, "Productivity returns to experience in the teacher labor market: Methodological challenges and new evidence on long-term career improvement," *Journal of Public Economics*, vol. 130, pp. 105–119, 2015.

[98] A. Pentina, V. Sharmanska, and C. H. Lampert, "Curriculum learning of multiple tasks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5492–5500.

[99] R. Stoean, C. Stoean, A. Sandita, D. Ciobanu, and C. Mesina, "Ensemble of classifiers for length of stay prediction in colorectal cancer," in *International Work-Conference on Artificial Neural Networks*, Springer, 2015, pp. 444–457.

[100] C. Sullivan, A. Staib, R. Eley, A. Scanlon, J. Flores, and I. Scott, "National emergency access targets metrics of the emergency department–inpatient interface: Measures of patient flow and mortality for emergency admissions to hospital," *Australian Health Review*, vol. 39, no. 5, pp. 533–538, 2015.

[101] G. A. Abel, M. E. Barclay, and R. A. Payne, "Adjusted indices of multiple deprivation to enable comparisons within and between constituent countries of the uk including an illustration using mortality rates," *British Medical Journal Open*, vol. 6, no. 11, 2016.

[102] S. Barnes, E. Hamrock, M. Toerper, S. Siddiqui, and S. Levin, "Real-time prediction of inpatient length of stay for discharge prioritization," *Journal of the American Medical Informatics Association*, vol. 23, no. e1, e2–e10, 2016.

[103] D. Campbell, "Nhs trust bosses slam£ 600m hospital fines over patient targets," *The Guardian(March 29), https://www. theguardian. com/society/2016/mar/29/nhs-bosses-slam-600m-hospital-fines-over-patient-targets*, 2016.

[104] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

[105] M. Elbattah and O. Molloy, "Coupling simulation with machine learning: A hybrid approach for elderly discharge planning," in *Proceedings of the 2016 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, 2016, pp. 47–56.

[106] ——, "Using machine learning to predict length of stay and discharge destination for hip-fracture patients," in *Proceedings of SAI Intelligent Systems Conference*, Springer, 2016, pp. 207–217.

[107] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-Wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark, "Mimic-iii, a freely accessible critical care database," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016.

[108] S. Karakusevic, "Understanding patient flow in hospitals," *Nuffield Trust, London*, 2016.

[109] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," in *Advances in Neural Information Processing Systems*, 2016, pp. 3675–3683.

[110] S. Molaei, F. K. Korley, S. R. Soroushmehr, H. Falk, H. Sair, K. Ward, and K. Najarian, "A machine learning based approach for identifying traumatic brain injury patients for whom a head ct scan can be avoided," in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, 2016, pp. 2258–2261.

[111] W. H. Organization, *World health statistics 2016: monitoring health for the SDGs sustainable development goals*. World Health Organization, 2016.

[112] D. Vieira and J. Hollmén, "Resource frequency prediction in healthcare: Machine learning approach," in *2016 IEEE 29th International Symposium on Computer-Based Medical Systems (CBMS)*, IEEE, 2016, pp. 88–93.

[113] J. Yoon, A. Alaa, S. Hu, and M. Schaar, "Forecasticu: A prognostic decision support system for timely prediction of intensive care unit admission," in *International Conference on Machine Learning*, 2016, pp. 1680–1689.

[114] A. R. Al Taleb, M. Hoque, A. Hasanat, and M. B. Khan, "Application of data mining techniques to predict length of stay of stroke patients," in *2017 International Conference on Informatics, Health & Technology (ICIHT)*, IEEE, 2017, pp. 1–5.

[115] S. Appalaraju and V. Chaoji, "Image similarity using deep cnn and curriculum learning," *arXiv preprint arXiv:1709.08761*, 2017.

[116] T. Desautels, R. Das, J. Calvert, M. Trivedi, C. Summers, D. J. Wales, and A. Ercole, "Prediction of early unplanned intensive care unit readmission in a uk tertiary care hospital: A cross-sectional machine learning approach," *British Medical Journal Open*, vol. 7, no. 9, 2017.

[117] A. Graves, M. G. Bellemare, J. Menick, R. Munos, and K. Kavukcuoglu, "Automated curriculum learning for neural networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, JMLR. org, 2017, pp. 1311–1320.

[118] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[119] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "Deepfm: A factorization-machine based neural network for ctr prediction," *arXiv preprint arXiv:1703.04247*, 2017.

[120] C. Ham, *Next steps on the nhs five year forward view*, 2017.

[121] L. Hiam, D. Dorling, D. Harrison, and M. McKee, "What caused the spike in mortality in england and wales in january 2015?" *Journal of the Royal Society of Medicine*, vol. 110, no. 4, pp. 131–137, 2017.

[122] S. Horng, D. A. Sontag, Y. Halpern, Y. Jernite, N. I. Shapiro, and L. A. Nathanson, "Creating an automated trigger for sepsis clinical decision support at emergency department triage using machine learning," *Public Library of Science One*, vol. 12, no. 4, e0174708, 2017.

[123] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[124] S. Kelly and M. I. Heywood, "Multi-task learning in atari video games with emergent tangled program graphs," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ACM, 2017, pp. 195–202.

[125] W. Liu, B. Dai, A. Humayun, C. Tay, C. Yu, L. B. Smith, J. M. Rehg, and L. Song, "Iterative machine teaching," *arXiv preprint arXiv:1705.10470*, 2017.

[126] Y. Liu, Y. Xie, and A. Srivastava, "Neural trojans," in *2017 IEEE International Conference on Computer Design (ICCD)*, IEEE, 2017, pp. 45–48.

[127] F. Ma, R. Chitta, J. Zhou, Q. You, T. Sun, and J. Gao, "Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2017, pp. 1903–1911.

[128] T. Matiisen, A. Oliver, T. Cohen, and J. Schulman, "Teacher-student curriculum learning," *arXiv preprint arXiv:1707.00183*, 2017.

[129] A. McCoy and R. Das, "Reducing patient mortality, length of stay and readmissions through machine learning-based sepsis prediction in the emergency department, intensive care unit and hospital floor units," *British Medical Journal Open Quality*, vol. 6, no. 2, 2017.

[130] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, PMLR, 2017, pp. 1273–1282.

[131] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, "Deep learning for healthcare: review, opportunities and challenges," *Briefings in Bioinformatics*, vol. 19, no. 6, pp. 1236–1246, May 2017.

[132] N. Offord, P. Harriman, and T. Downes, "Discharge to assess: Transforming the discharge process of frail older patients," *Future Hospital Journal*, vol. 4, no. 1, p. 30, 2017.

[133] W. Samek, T. Wiegand, and K.-R. Müller, "Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models," *arXiv preprint arXiv:1708.08296*, 2017.

[134] K. Shameer, K. W. Johnson, A. Yahi, R. Miotto, L. Li, D. Ricks, J. Jebakaran, P. Kovatch, P. P. Sengupta, S. Gelijns, *et al.*, "Predictive modeling of hospital readmission rates using electronic medical record-wide machine learning: A case-study using mount sinai heart failure cohort," in *Pacific Symposium on Biocomputing 2017*, World Scientific, 2017, pp. 276–287.

[135] A. Staib, C. Sullivan, J. B. Prins, A. Burton-Jones, G. Fitzgerald, and I. Scott, "Uniting emergency and inpatient clinicians across the ed–inpatient interface: The last frontier?" *Emergency Medicine Australasia*, vol. 29, no. 6, pp. 740–745, 2017.

[136] L. Turgeman, J. H. May, and R. Sciulli, "Insights from a machine learning model for predicting the hospital length of stay (los) at the time of admission," *Expert Systems with Applications*, vol. 78, pp. 376–385, 2017.

[137] A. Van Den Oord, O. Vinyals, *et al.*, "Neural discrete representation learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 6306–6315.

[138] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *Proceedings of the Ad Knowledge Discovery and Data Mining '17*, ACM, 2017, p. 12.

[139] B. Wellner, J. Grand, E. Canzone, M. Coarr, P. W. Brady, J. Simmons, E. Kirkendall, N. Dean, M. Kleinman, and P. Sylvester, "Predicting unplanned transfers to the intensive care unit: A machine learning approach leveraging diverse clinical elements," *Journal of Medical Internet Research Medical Informatics*, vol. 5, no. 4, e45, 2017.

[140] Y. Wu and Y. Tian, "Training agent for first-person shooter game with actor-critic curriculum learning," in *International Conference on Learning Representations*, 2017.

[141] E. Xia, J. Mei, G. Xie, X. Li, Z. Li, and M. Xu, "Learning doctors' medicine prescription pattern for chronic disease treatment by mining electronic health records: A multi-task learning approach," in *American Medical Informatics Association Annual Symposium Proceedings*, American Medical Informatics Association, vol. 2017, 2017, p. 1828.

[142] H. Xu, W. Wu, S. Nemati, and H. Zha, "Patient flow prediction via discriminative learning of mutually-correcting processes," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 1, pp. 157–171, 2017.

[143] X. Zhang, J. Kim, R. E. Patzer, S. R. Pitts, A. Patzer, and J. D. Schrager, "Prediction of emergency department hospital admission based on natural language processing and neural networks," *Methods of Information in Medicine*, vol. 56, no. 5, pp. 377–89, 2017.

[144] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Model poisoning attacks in federated learning," in *Workshop on Secure Machine Learning 32nd Conference Neural Information Processing Systems (NeurIPS)*, 2018.

[145] Y. Fan, F. Tian, T. Qin, X.-Y. Li, and T.-Y. Liu, "Learning to teach," *arXiv preprint arXiv:1805.03643*, 2018.

[146] B. Graham, R. Bond, M. Quinn, and M. Mulvenna, "Using data mining to predict hospital admissions from the emergency department," *IEEE Access*, vol. 6, pp. 10 458–10 469, 2018.

[147] W. S. Hong, A. D. Haimovich, and R. A. Taylor, "Predicting hospital admission at emergency department triage using machine learning," *Public Library of Science One*, vol. 13, no. 7, e0201016, 2018.

[148] G. Huang, S. Liu, L. Van der Maaten, and K. Q. Weinberger, "Condensenet: An efficient densenet using learned group convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2752–2761.

[149] Z. Huang and W. Dong, "Adversarial mace prediction after acute coronary syndrome using electronic health records," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 5, pp. 2117–2126, 2018.

[150] S. L. James, D. Abate, K. H. Abate, S. M. Abay, C. Abbafati, N. Abbasi, H. Abbastabar, F. Abd-Allah, J. Abdela, A. Abdelalim, *et al.*, "Global, regional, and national incidence, prevalence, and years lived with disability for 354 diseases and injuries for 195 countries and territories, 1990–2017: A systematic analysis for the global burden of disease study 2017," *The Lancet*, vol. 392, no. 10159, pp. 1789–1858, 2018.

[151] S. Kalagara, A. E. Eltorai, W. M. Durand, J. M. DePasse, and A. H. Daniels, "Machine learning modeling for predicting hospital readmission following lumbar laminectomy," *Journal of Neurosurgery: Spine*, vol. 30, no. 3, pp. 344–352, 2018.

[152] C. Launay, A. Kabeshova, A. Lanoé, J. Chabot, E. Levinoff, and O. Beauchet, "Age effect on the prediction of risk of prolonged length hospital stay in older

patients visiting the emergency department: Results from a large prospective geriatric cohort study," *BioMed Central Geriatrics*, vol. 18, no. 1, p. 127, 2018.

[153] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *International Symposium on Research in Attacks, Intrusions, and Defenses*, Springer, 2018, pp. 273–294.

[154] J. A. Lucke, J. de Gelder, F. Clarijs, C. Heringhaus, A. J. de Craen, A. J. Fogteloo, G. J. Blauw, B. de Groot, and S. P. Mooijaart, "Early prediction of hospital admission for emergency department patients: A comparison between patients younger or older than 70 years," *Emergency Medicine Journal*, vol. 35, no. 1, pp. 18–27, 2018.

[155] Q. Mao, M. Jay, J. L. Hoffman, J. Calvert, C. Barton, D. Shimabukuro, L. Shieh, U. Chettipally, G. Fletcher, Y. Kerem, *et al.*, "Multicentre validation of a sepsis prediction algorithm using only vital sign data in the emergency department, general ward and icu," *British Medical Journal Open*, vol. 8, no. 1, 2018.

[156] T. H. McCoy, A. M. Pellegrini, and R. H. Perlis, "Assessment of time-series machine learning methods for forecasting hospital discharge volume," *Journal of the American Medical Association Network Open*, vol. 1, no. 7, e184087–e184087, 2018.

[157] S. J. Patel, D. B. Chamberlain, and J. M. Chamberlain, "A machine learning approach to predicting need for hospitalization for pediatric asthma exacerbation at the time of emergency department triage," *Academic Emergency Medicine*, vol. 25, no. 12, pp. 1463–1470, 2018.

[158] T. J. Pollard, A. E. Johnson, J. D. Raffa, L. A. Celi, R. G. Mark, and O. Badawi, "The eicu collaborative research database, a freely available multi-center database for critical care research," *Scientific data*, vol. 5, no. 1, pp. 1–13, 2018.

[159] J. C. Rojas, K. A. Carey, D. P. Edelson, L. R. Venable, M. D. Howell, and M. M. Churpek, "Predicting intensive care unit readmission with machine learning using electronic health record data," *Annals of the American Thoracic Society*, vol. 15, no. 7, pp. 846–853, 2018.

[160] J. Rubin, C. Potes, M. Xu-Wilson, J. Dong, A. Rahman, H. Nguyen, and D. Moromisato, "An ensemble boosting model for predicting transfer to the pediatric intensive care unit," *International Journal of Medical Informatics*, vol. 112, pp. 15–20, 2018.

[161] K. Shailaja, B. Seetharamulu, and M. Jabbar, "Machine learning in healthcare: A review," in *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, IEEE, 2018, pp. 910–914.

[162] S. Srinivas and A. R. Ravindran, "Optimizing outpatient appointment system using machine learning algorithms and scheduling rules: A prescriptive analytics framework," *Expert Systems with Applications*, vol. 102, pp. 245–261, 2018.

[163] R. A. Taylor, C. L. Moore, K.-H. Cheung, and C. Brandt, "Predicting urinary tract infections in the emergency department with machine learning," *Public Library of Science One*, vol. 13, no. 3, e0194085, 2018.

[164] S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

[165] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečn, S. Mazzocchi, H. B. McMahan, *et al.*, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.

[166] T. A. Daghistani, R. Elshawi, S. Sakr, A. M. Ahmed, A. Al-Thwayee, and M. H. Al-Mallah, "Predictors of in-hospital length of stay among cardiac patients: A machine learning approach," *International Journal of Cardiology*, vol. 288, pp. 140–147, 2019.

[167] R. J. Delahanty, J. Alvarez, L. M. Flynn, R. L. Sherwin, and S. S. Jones, "Development and evaluation of a machine learning model for the early identification of patients at risk for sepsis," *Annals of Emergency Medicine*, vol. 73, no. 4, pp. 334–344, 2019.

[168] G. Hacohen and D. Weinshall, "On the power of curriculum learning in training deep networks," in *International Conference on Machine Learning*, PMLR, 2019, pp. 2535–2544.

[169] Y. Huang, Y. Cheng, A. Bapna, O. Firat, D. Chen, M. Chen, H. Lee, J. Ngiam, Q. V. Le, Y. Wu, *et al.*, "Gpipe: Efficient training of giant neural networks using pipeline parallelism," in *Advances in Neural Information Processing Systems*, 2019, pp. 103–112.

[170] F. K. Khattak, C. Pou-Prom, R. Wu, and F. Rudzicz, "Predicting icu transfers using text messages between nurses and doctors," in *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, 2019, pp. 89–94.

[171] N. Khosravan, H. Celik, B. Turkbey, E. C. Jones, B. Wood, and U. Bagci, "A collaborative computer aided diagnosis (c-cad) system with eye-tracking, sparse attentional model, and deep learning," *Medical Image Analysis*, vol. 51, pp. 101–115, 2019.

[172] J. Krämer, J. Schreyögg, and R. Busse, "Classification of hospital admissions into emergency and elective care: A machine learning approach," *Health Care Management Science*, vol. 22, no. 1, pp. 85–105, 2019.

[173] J. Lee, N. Ahmidi, R. Srinivasan, D. Alejo, J. Dinatale, S. Schena, G. Whitman, M. Sussman, and I. Shpitser, "P5706 finding predictors and causes of cardiac surgery icu readmission using machine learning and causal inference," *European Heart Journal*, vol. 40, no. Supplement_1, ehz746–0647, 2019.

[174] Y.-W. Lin, Y. Zhou, F. Faghri, M. J. Shaw, and R. H. Campbell, "Analysis and prediction of unplanned intensive care unit readmission using recurrent neural networks with long short-term memory," *Public Library of Science One*, vol. 14, no. 7, e0218942, 2019.

[175] G. Luo, B. L. Stone, F. L. Nkoy, S. He, and M. D. Johnson, "Predicting appropriate hospital admission of emergency department patients with bronchiolitis: Secondary analysis," *Journal of Medical Internet Research Medical Informatics*, vol. 7, no. 1, e12591, 2019.

[176] X. Min, B. Yu, and F. Wang, "Predictive modeling of the hospital readmission risk from patients' claims data using machine learning: A case study on copd," *Scientific Reports*, vol. 9, no. 1, pp. 1–10, 2019.

[177] W. E. Muhlestein, D. S. Akagi, J. M. Davies, and L. B. Chambless, "Predicting inpatient length of stay after brain tumor surgery: Developing machine learning ensembles to improve predictive performance," *Neurosurgery*, vol. 85, no. 3, pp. 384–393, 2019.

[178] A. Nelson, D. Herron, G. Rees, and P. Nachev, "Predicting scheduled hospital attendance with artificial intelligence," *Nature Partner Journals Digital Medicine*, vol. 2, no. 1, pp. 1–7, 2019.

[179] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[180] Y. Raita, T. Goto, M. K. Faridi, D. F. Brown, C. A. Camargo, and K. Hasegawa, "Emergency department triage prediction of clinical outcomes using machine learning models," *Critical Care*, vol. 23, no. 1, p. 64, 2019.

[181] P. N. Ramkumar, S. M. Navarro, H. S. Haeberle, J. M. Karnuta, M. A. Mont, J. P. Iannotti, B. M. Patterson, and V. E. Krebs, "Development and validation of a machine learning algorithm after primary total hip arthroplasty: Applications

to length of stay and payment models," *The Journal of Arthroplasty*, vol. 34, no. 4, pp. 632–637, 2019.

[182] H. Rosemarin, A. Rosenfeld, and S. Kraus, "Emergency department online patient-caregiver scheduling," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 695–701.

[183] G. Sariyer, C. Ö. Taşar, and G. E. Cepe, "Use of data mining techniques to classify length of stay of emergency department patients," *Bio-Algorithms and Med-Systems*, vol. 15, no. 1, 2019.

[184] P. Shah, F. Kendall, S. Khozin, R. Goosen, J. Hu, J. Laramie, M. Ringel, and N. Schork, "Artificial intelligence and machine learning in clinical development: A translational perspective," *Nature Partner Journals Digital Medicine*, vol. 2, no. 1, pp. 1–5, 2019.

[185] W. Song, C. Shi, Z. Xiao, Z. Duan, Y. Xu, M. Zhang, and J. Tang, "Autoint: Automatic feature interaction learning via self-attentive neural networks," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 1161–1170.

[186] N. W. Sterling, R. E. Patzer, M. Di, and J. D. Schrager, "Prediction of emergency department patient disposition based on natural language processing of triage notes," *International Journal of Medical Informatics*, vol. 129, pp. 184–188, 2019.

[187] F. P. Such, A. Rawal, J. Lehman, K. O. Stanley, and J. Clune, "Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data," *arXiv preprint arXiv:1912.07768*, 2019.

[188] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *2019 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2019, pp. 707–723.

[189] A. Williams, A.-M. Mekhail, J. Williams, J. McCord, and V. Buchan, "Effective resource management using machine learning in medicine: An applied example," *BMJ Simulation and Technology Enhanced Learning*, vol. 5, no. 2, 2019.

[190] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "Hybridalpha: An efficient approach for privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 13–23.

[191] N. Yousefi, F. Hasankhani, and M. Kiani, "Appointment scheduling model in healthcare using clustering algorithms," *arXiv preprint arXiv:1905.03083*, 2019.

[192] A. Artetxe, M. Graña, A. Beristain, and S. Rıos, "Balanced training of a hybrid ensemble method for imbalanced datasets: A case of emergency department readmission prediction," *Neural Computing and Applications*, vol. 32, no. 10, pp. 5735–5744, 2020.

[193] S. Bacchi, S. Gluck, Y. Tan, I. Chim, J. Cheng, T. Gilbert, D. K. Menon, J. Jannes, T. Kleinig, and S. Koblar, "Prediction of general medical admission length of stay with natural language processing and deep learning: A pilot study," *Internal and Emergency Medicine*, pp. 1–7, 2020.

[194] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2020, pp. 2938–2948.

[195] C. Baker, "Nhs key statistics," *House of Commons Library*, vol. 7281, 2020.

[196] R. El-Bouri, D. Eyre, P. Watkinson, T. Zhu, and D. Clifton, "Student-teacher curriculum learning via reinforcement learning: Predicting hospital inpatient admission location," *Proceedings of Machine Learning Research*, vol. 119, 2020.

[197] D. Castiñeira, K. R. Schlosser, A. Geva, A. R. Rahmani, G. Fiore, B. K. Walsh, C. D. Smallwood, J. H. Arnold, and M. Santillana, "Adding continuous vital sign information to static clinical data improves the prediction of length of stay

after intubation: A data-driven machine learning approach," *Respiratory Care*, vol. 65, no. 9, pp. 1367–1377, 2020.

[198] I. Y. Chen, S. Joshi, M. Ghassemi, and R. Ranganath, "Probabilistic machine learning for healthcare," *arXiv preprint arXiv:2009.11087*, 2020.

[199] A. Cooper, A. Carson-Stevens, T. Hughes, and A. Edwards, "Is streaming patients in emergency departments to primary care services effective and safe?" *British Medical Journal*, vol. 368, 2020.

[200] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *29th Security Symposium (Security 20)*, 2020, pp. 1605–1622.

[201] M. Fernandes, R. Mendes, S. M. Vieira, F. Leite, C. Palos, A. Johnson, S. Finkelstein, S. Horng, and L. A. Celi, "Predicting intensive care unit admission among patients presenting to the emergency department using machine learning and natural language processing," *Public Library of Science One*, vol. 15, no. 3, e0229331, 2020.

[202] S. N. Finkelstein, "Predicting intensive care unit admission among patients presenting to the emergency department using machine learning and natural language processing," *Public Library of Science (PLoS)*, 2020.

[203] B. Hyslop, "Not safe for discharge? words, values, and person-centred care," *Age and Ageing*, vol. 49, no. 3, pp. 334–336, 2020.

[204] J. Kirchebner, M. P. Günther, M. Sonnweber, A. King, and S. Lau, "Factors and predictors of length of stay in offenders diagnosed with schizophrenia-a machine-learning-based approach," *BioMed Central Psychiatry*, vol. 20, pp. 1–12, 2020.

[205] C. S. Lee and A. Y. Lee, "Clinical applications of continual learning machine learning," *The Lancet Digital Health*, vol. 2, no. 6, e279–e281, 2020.

[206] S.-Y. Lee, R. B. Chinnam, E. Dalkiran, S. Krupp, and M. Nauss, "Prediction of emergency department patient disposition decision for proactive resource allocation for admission," *Health Care Management Science*, vol. 23, no. 3, pp. 339–359, 2020.

[207] S. Lee and Y. H. Lee, "Improving emergency department efficiency by patient scheduling using deep reinforcement learning," in *Healthcare*, Multidisciplinary Digital Publishing Institute, vol. 8, 2020, p. 77.

[208] T. Mahmud, M. A. Rahman, and S. A. Fattah, "Covxnet: A multi-dilation convolutional neural network for automatic covid-19 and other pneumonia detection from chest x-ray images with transferable multi-receptive feature optimization," *Computers in Biology and Medicine*, vol. 122, p. 103 869, 2020.

[209] D. A. Martinez, S. R. Levin, E. Y. Klein, C. R. Parikh, S. Menez, R. A. Taylor, and J. S. Hinson, "Early prediction of acute kidney injury in the emergency department with machine-learning methods applied to electronic health record data," *Annals of Emergency Medicine*, 2020.

[210] F. Mowbray, M. Zargoush, A. Jones, K. de Wit, and A. Costa, "Predicting hospital admission for older emergency department patients: Insights from machine learning," *International Journal of Medical Informatics*, p. 104 163, 2020.

[211] M. A. Rahman, B. Honan, T. Glanville, P. Hough, and K. Walker, "Using data mining to predict emergency department length of stay greater than 4 hours: Derivation and single-site validation of a decision tree algorithm," *Emergency Medicine Australasia*, vol. 32, no. 3, pp. 416–421, 2020.

[212] S. Ramgopal, C. M. Horvat, N. Yanamala, and E. R. Alpern, "Machine learning to predict serious bacterial infections in young febrile infants," *Pediatrics*, vol. 146, no. 3, 2020.

[213] B. P. Roquette, H. Nagano, E. C. Marujo, and A. C. Maiorano, "Prediction of admission in pediatric emergency department with deep neural networks and triage textual data," *Neural Networks*, 2020.

[214] S. Sheikhalishahi, V. Balaraman, and V. Osmani, "Benchmarking machine learning models on multi-centre eicu critical care dataset," *Public Library of Science One*, vol. 15, no. 7, e0235424, 2020.

[215] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *European Symposium on Research in Computer Security*, Springer, 2020, pp. 480–501.

[216] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," *arXiv preprint arXiv:2002.06440*, 2020.

[217] H. Yuan and T. Ma, "Federated accelerated stochastic gradient descent," *arXiv preprint arXiv:2006.08950*, 2020.

[218] T. Zahavy, Z. Xu, V. Veeriah, M. Hessel, J. Oh, H. P. van Hasselt, D. Silver, and S. Singh, "A self-tuning actor-critic algorithm," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[219] A. Pryce, M. Unwin, L. Kinsman, and D. McCann, "Delayed flow is a risk to patient safety: A mixed method analysis of emergency department patient flow," *International Emergency Nursing*, vol. 54, p. 100 956, 2021.