Developing Deep Learning Continuous Risk Models for Early Adverse Event Prediction in Electronic Health Records: an AKI Case Study

Nenad Tomašev (nenadt@deepmind.com)
DeepMind, London, UK
Xavier Glorot
DeepMind, London, UK
Jack W. Rae
DeepMind, London, UK
Michal Zielinski
DeepMind, London, UK
Harry Askham
DeepMind, London, UK
Andre Saraiva
DeepMind, London, UK
Anne Mottram
DeepMind, London, UK
Clemens Meyer
DeepMind, London, UK
Suman Ravuri
DeepMind, London, UK
Ivan Protsyuk
DeepMind, London, UK
Alistair Connell
DeepMind, London, UK
Cían O. Hughes
DeepMind, London, UK
Alan Karthikesalingam
DeepMind, London, UK
Julien Cornebise
DeepMind, London, UK
Hugh Montgomery
Institute for Human Health and Performance, University College London, London, UK
Geraint Rees

Institute of Cognitive Neuroscience, University College London, London, UK
Chris Laing
University College London Hospitals, London, UK
Clifton R. Baker
Department of Veterans Affairs, USA
Kelly Peterson
VA Salt Lake City Healthcare System, USA
Ruth Reeves
Department of Veterans Affairs, USA
Demis Hassabis
DeepMind, London, UK
Dominic King
DeepMind, London, UK
Mustafa Suleyman
DeepMind, London, UK
Trevor Back
DeepMind, London, UK
Christopher Nielson
Department of Veterans Affairs, USA
Joseph R. Ledsam (jledsam@google.com)
DeepMind, London, UK
Shakir Mohamed
DeepMind, London, UK

Method Article

Keywords: machine learning, deep learning, artificial intelligence, electronic health records, risk prediction, acute kidney injury

Posted Date: July 31st, 2019

DOI: https://doi.org/10.21203/rs.2.10083/v1

Abstract

Early detection of patient deterioration is key to unlocking the potential for targeted preventative care and improving patient outcomes. This protocol describes a workflow for developing deep learning continuous risk models for early prediction of future acute adverse events from electronic health records (EHR), taking the prediction of the risk of future acute kidney injury (AKI) as an exemplar. The protocol consists of 34 steps grouped into the following stages: formal problem definition, data processing, model architecture selection, risk calibration and uncertainty, and evaluating model generalisability. For the protocol to be applicable to modelling the future risk of a particular condition, the problem formulation should be clinically and physiologically plausible and there needs to be sufficient associated predictive signal in routinely collected EHR data. Prospective validation is key in evaluating whether retrospective models developed by following the proposed protocol are clinically applicable and useful.

Introduction

Identifying and treating patient deterioration is an important part of delivering effective hospital care. Delays in detection can lead to both short- and long-term complications, extended length of stay, increased costs and poor patient outcomes: an estimated 11% of in-hospital deaths result from a failure to promptly recognise patient deterioration[1].

Machine learning models aiming to provide predictive insights into likely future patient deterioration might help predict adverse patient events. Several such ML models of patient deterioration have been proposed recently[2-18]. Despite increasing interest, few solutions have been validated prospectively and adopted in routine clinical care. There are several practical challenges when designing ML models from retrospective EHR data and a failure to address these challenges can impair the clinical applicability of the model.

Systems for early detection or prediction of patient deterioration need to: (i) deliver actionable insights on preventable conditions; (ii) be personalised for specific patients; (iii) offer sufficient contextual information to inform clinical decision-making; and (iv) be generally applicable across patient populations[19]. What makes addressing these points especially difficult is the nature of EHR data: they are sparse, high-dimensional, asynchronous, and often noisy due to missing and incorrect entries. Our proposed protocol aims to address these challenges and provide a robust way of developing ML models on retrospective datasets for sequentially predicting early patient deterioration from incoming evidence in EHR (Figure 1). The models should be able to utilise the entirety of patient history along with the observations made at present time, to provide predictions early enough to give clinicians the necessary amount of time to assess the patients and act on the predicted risk of future adverse events.

Development of the protocol

We demonstrate the efficacy of our protocol using the example of developing models for early prediction of Acute Kidney Injury (AKI)[20], a common condition affecting up to 1 in 5 hospital inpatients[21,22]. It is believed that a significant proportion of cases might be preventable with early treatment[23], although the inability of existing rule-based AKI detection systems to detect early physiological change associated with renal injury renders this hypothesis hard to test in practice. In particular, the commonly used 'Kidney Disease: Improving Global Outcomes' (KDIGO) criteria[24] define AKI according to an observed absolute or relative increases in serum creatinine concentration. However, significant increases in serum creatinine are sometimes only apparent 48h after substantial damage to the kidneys has already taken place. KDIGO-based alerts therefore rarely provide the opportunity to intervene early enough to limit damage and existing prospective trials of KDIGO-based alerting systems show little practical benefit in improving patient outcomes[25]. These studies highlight the need for providing earlier alerts of deteriorating kidney function to open new opportunities for providing early preventative care.

The continuous model of future AKI risk developed using this protocol could potentially be a part of the solution, a conclusion based on retrospective evaluation of its predictive performance[20]. The model was able to correctly predict future KDIGO AKI up to 48h ahead of time, a window sufficient for clinical intervention. The model correctly predicted 55.8% of all inpatient AKI events ahead of time, and 90.2% of cases where subsequent regular administration of dialysis was scheduled within 90 days of the initial onset. This performance was obtained at an operating point where the model would raise two false positive alerts for each true positive prediction. Further analysis showed this to be a conservative estimate of the false positive rate; true positive predictions made earlier than 48h as well as *trailing* predictions of AKI in cases where it was in the process of resolving were both formally counted as false positives. Only 6% of all false positive alerts could not be attributed to patients at an increased risk of AKI due to either known comorbidities or events observed in the admission. Difficulties in assessing the percentage of false positive alerts that would be distracting and which would not bring any clinical insights highlight the necessity of prospective evaluation. Clinical trials are necessary in order to reliably evaluate the clinical utility of retrospectively developed ML predictive risk models.

The proposed protocol involves five model development stages: (a) formal problem definition, (b) data processing, (c) model architecture selection, (d) risk calibration and uncertainty, plus (e) evaluating model generalisability. (a) Formal problem definition involves assessing a clinical use case and providing a technical specification on how to extract the relevant ground truth information on the primary and secondary prediction targets. (b) Data preprocessing is required to transform EHR data into a sequential representation that is amenable to training deep learning models. (c) Model architecture selection involves the steps required to implement, select and optimise the predictive risk model on data representations derived from the EHR in the previous stage. (d) The final model architecture is then calibrated and uncertainty estimates are associated with each prediction. (e) Finally, additional evaluation is done on model generalisability to gather evidence for clinical applicability and model readiness for subsequent prospective evaluation. We describe our procedure in 34 steps. The final prospective evaluation step described in Step 34 was not included in the manuscript[20], but is recognised

as a necessary step in model validation. We include it here for completeness, to explicitly point out that retrospective validation is not sufficient to establish clinical utility.

Applications of the method

The proposed protocol can potentially be applied to developing ML models for continuously predicting risk of various future adverse events from EHRs. Most of the protocol is generic and target-agnostic. The parts of the existing protocol that are specific to its application to AKI prediction[20] are limited to the integration of domain knowledge in the technical specification of the ground truth definition, the inclusion/exclusion criteria for patients and individual steps, and the time windows across which the predictions are made. For these steps, we provide guidelines on how to adapt the protocol to additional future applications.

Examples of potential further applications of the protocol include sepsis[7,10], cardiopulmonary arrest, unplanned Intensive Care Unit (ICU) admission and mortality.

For this protocol to be applicable to a particular clinical use case, the use case needs to meet several basic conditions: (i) the condition needs to be acute i.e. occurring at a defined time; (ii) ground truth should either be possible to computationally derive from EHR data or be reliably provided by clinical experts for a sufficiently large retrospective dataset; (iii) there needs to be sufficient predictive signal in routinely collected EHR data to train machine learning (ML) models that can reach clinically applicable levels of retrospectively evaluated performance; (iv) predictive features should be encoded in structured fields of EHR, in contrast to the unstructured content of clinical notes; (v) the clinically actionable time window over which the risk prediction is made needs to be plausible with respect to the granularity of entries in EHR and the delays with which the data are being recorded.

Finally, continuous risk models can easily be applied to the more restricted case of associating a general risk of future deterioration at admission time. Instead of using fixed future time windows, using the duration of the admission as the future time frame across which the risk is to be provided is a simple extension. Such a general risk score can be useful for triaging and resource allocation, whereas a continuously updated sequential risk of future deterioration across short future time windows presents a more directly actionable use case, enabling clinicians to react to concrete developments.

Comparison with other methods

A number of recent studies have proposed using machine learning to build predictive models from EHR data[2-18]. Existing studies employ many of the steps outlined in our proposed protocol, with some differences. Direct detailed comparison of formal model development protocols is non-trivial, given that these tend not to be formally reported at the level of detail that is required. Detailed specifications of model development protocols play an important role in reproducibility of the reported results.

None of the individual steps contained in the proposed protocol are novel in and of themselves. It is the combination of all these best practices that enables the development of effective predictive models of future adverse events from the EHR.

In practice, protocols can be compared based on the effectiveness of the resulting ML risk models. Given that we have demonstrated the effectiveness of the proposed protocol in the context of predicting the risk of future AKI, here we compare our protocol to those that were previously used to develop AKI risk models. Most prior work on AKI risk modelling with ML involved protocols that were limited to only produce models that provide predictions at a single point in time[26-28], instead of a personalised, continuously updating risk score. Previously developed continuous machine learning models of AKI risk have either not demonstrated a clinically applicable level of predictive performance[29] or have focused on predictions across too short a time horizon, leaving little time for clinical assessment and intervention[30]. The application of our proposed protocol resulted in state-of-the-art retrospective performance for continuously predicting the risk of future AKI[20], at a level that is potentially clinically applicable, pending future prospective validation.

Experimental design

Clinical design

The aspects of the protocol that relate to clinical decisions, and the modelling and the experimental design of the clinical use case, may vary depending on the research question being studied; these aspects may need to be adjusted accordingly. During step 1 these differences should be identified and subsequent steps should be adjusted accordingly. Steps 2 and 4, referring to the derivation of ground truth labels and dataset specification respectively may differ considerably. For conditions such as sepsis there may be no accepted ground truth, and a pragmatic composite measure may require a consensus definition from expert panels and international best-practice. Other conditions, such as pulmonary embolism, may be associated with an appropriate gold standard to confirm diagnosis - in this case radiological evidence - that appears in EHR data after the condition has already been identified. Another example might be prediction of mortality, where certain patient groups must be identified and excluded to avoid potentially biasing the results: palliative care patients are a notable example in this case. In each of these cases the experimental decisions made will differ in clinically important ways. In all cases the exact data specifications may differ, and efforts to identify features for use in baseline models will focus on parts of the data identified as most likely to be predictive by clinical experts.

Taking the specific example of sepsis, the following illustrates how specific steps in the proposed protocol can be adjusted to this particular clinical predictive task:

1. Adjustments to Step 2: As sepsis definitions have changed over time[31] and EHR may not always encode the entirety of the information required to accurately identify sepsis in each case, pragmatic rule-based definitions present a potentially viable alternative. One example of this is (i) recorded evidence of

presumed serious infection (blood culture obtained and ≥ 4 days of antibiotics administered starting within ±2 days of the blood culture) and (ii) evidence of acute organ dysfunction within ±2 days of the blood culture (vasopressor initiation, initiation of mechanical ventilation, doubling in serum creatinine level or decrease by 50% of estimated glomerular filtration rate relative to baseline (excluding patients with ICD-9-CM code for end-stage kidney disease [585.6]), total bilirubin level 2.0 mg/dL and doubling from baseline, platelet count <100 cells/μL and 50% decline from baseline, serum lactate 2.0 mmol/L)[32].

2. Adjustments to Step 4: Due to its role in establishing a ground truth, particular care will be needed to review accuracy in how microbiology data are recorded.

3. Adjustments to Step 5: Vital signs and lab tests more relevant to sepsis. Examples include white blood cell count, C-Reactive Protein and Erythrocyte Sedimentation Rate.

4. Adjustments to Step 8: Higher granularity of sequential data is likely to be required, corresponding to shorter atomic time windows.

5. Adjustments to Step 11: A different set of manually-engineered features would be required. Examples include clinical codes specific to infection and sepsis, and laboratory tests specific to potential causes of infection.

6. Adjustments to Step 14: Shorter future prediction time windows within which to predict the onset of sepsis would likely be required[10].

7. Adjustments to Step 30: Examples of clinically relevant subgroups include immunosuppressed patients, hospital acquired infections and specific causative organisms.

Expertise needed to implement the protocol

Development of ML solutions in healthcare requires interdisciplinary collaboration between experts in ML, clinical medicine, clinical outcomes evaluation and patients themselves, all of whom should be involved in the implementation, execution and evaluation of the proposed protocol. Some steps in the protocol require significant technical expertise in deep learning and/or big data analytics, and should be executed by researchers with ML expertise and software engineering skills. Other steps require deep clinical knowledge and should be executed by clinicians or clinical academics. Prospective evaluation will require experts in trial design and in quantitative and qualitative analysis. Patient involvement in co-creation helps identify relevant clinical use cases, contributes to design and evaluation, and provides insights into security and privacy concerns.

Limitations

This protocol does not support using unstructured free text contained in clinical notes. The protocol could be extended to support additional inputs including features derived from clinical notes, though the natural language processing (NLP) component required for this extension would likely need to be pre-trained and

fixed, instead of being trained end-to-end along with the risk model. The reason for the exclusion of the NLP component in the initial protocol lies mainly in the fact that the research dataset available for model development in the manuscript[20] did not include clinical notes. Compiling large datasets of de-identified clinical notes is in general not an easy task, given that fully automating the anonymisation of such large datasets poses significant privacy challenges.

The proposed protocol does not currently support developing risk models that aim to infer explicit causal relationships in the data and at the moment only allows for modelling the observed correlations. We acknowledge that causal modelling from electronic health records is an important research direction towards developing physiologically plausible models that are robust and easy to generalise across clinical use cases[15,33].

The requirements that the clinical use cases need to satisfy for the protocol to be applicable are outlined in the "Applications of the method" Section. We would like to point out requirements ii) and iii) in particular as they relate to the existence of sufficient signal in the structured entries in EHR, for the use case in question. The implementation of the protocol needs to be done in close collaboration between ML researchers and clinical experts, who can provide the necessary clinical domain knowledge at various critical stages.

Developing machine learning models is an iterative process and a full assessment of whether the signal in the data is going to be sufficient for a high level of predictive performance is difficult to complete prior to having done initial investigative work. Despite there being commonalities to EHR datasets, each dataset will likely still pose some unique challenges in terms of data quality that are not always known beforehand. In some cases, the initial stages of the protocol will need to be re-executed in order to provide additional information regarding the likelihood of success for a particular use case. Despite this, work spent at this stage may be transferable to new research questions as most parts of the technical implementation of the protocol can be shared between different use cases.

In terms of additional limitations to the clinical use case, the proposed protocol may not be well-suited for recommendations in primary care or for predictions of disease progression in chronic conditions. Adjustments to make the protocol applicable to these additional types of predictive problems may be possible, but these extensions are yet to be considered.

The application of the protocol requires the availability of a sufficiently large deidentified EHR research dataset. The size of the dataset is critical not only to meet the requirements of training deep learning models, but also to capture the full extent of the variation in clinical practice. Large datasets representative of diverse patient demographics and disease phenotypes are required to train ML predictive adverse event risk models that can potentially be clinically applicable. The dataset used in the initial application of the protocol[20] comprised several years of longitudinal data on 703,782 adult patients across 172 inpatient and 1,062 outpatient sites at the US Department of Veterans Affairs (VA)[34].

Data size requirements will depend on the difficulty of the clinical use case, as well as the complexity of ML solutions being used.

Equipment

To execute the proposed protocol, researchers need access to the data, access to computational resources, and access to software packages that enable an efficient implementation of the technical software solutions that correspond to the individual steps of the protocol.

The protocol can be implemented in many different ways, since there are many software libraries that support large data processing and development of ML models and deep learning models in particular. Here we list one set of implementation options, corresponding to the existing implementation of the protocol in the manuscript[20].

Software

• Data processing framework Apache Beam (https://beam.apache.org/)

• Plotting library Matplotlib[35] (https://matplotlib.org/)

• Scientific computing library Numpy[36] (https://www.numpy.org/)

• Scientific computing library Scipy[37] (https://www.scipy.org/)

• Plotting library Seaborn (https://seaborn.pydata.org/)

• Machine learning library Scikit-learn[38] (https://scikit-learn.org/)

• Machine learning library Sonnet[39] (https://github.com/deepmind/sonnet/)

• Machine learning framework TensorFlow[40] (https://github.com/tensorflow/tensorflow/)

• Machine learning library Xgboost[41] (https://github.com/dmlc/xgboost/)

Procedure

The proposed protocol consists of the following five stages, each comprising a number of steps: (a) formal problem definition (5 steps), (b) data processing (10 steps), (c) model architecture selection (10 steps), (d) risk calibration and uncertainty (3 steps), and (e) model generalisability evaluation (6 steps). In total we describe 34 steps with each step itself comprising a number of sub-steps.

Formal problem definition

1. Evaluate the feasibility of a clinical use case: Identify a clinical use case involving adverse event prediction where sufficient clinical data are available. Formulate the research question through patient engagement, structured interviews with clinical teams and consultation of clinical guidelines and literature evidence. Map real life patient pathways associated with the clinical problem and evaluate whether the proposed research question will impact clinical practice as intended. If there is consensus between multiple clinical and patient experts on the feasibility of the use case, proceed.

2. Obtain or derive ground truth labels. Identify an appropriate ground truth 'label' in the data for the model to predict that is as close to a gold standard diagnosis as possible. Internationally accepted proxy measures - such as "Kidney Disease: Improving Global Outcomes (KDIGO)" criteria[24] for AKI - may be used where obtaining gold standard labels is not feasible.

3. Inclusion and exclusion criteria: Identify types of patients that should be excluded from research data, based on the particular clinical use case. Remove all patient records that match the full patient exclusion criteria. For the remaining patient records, define exclusion criteria on a level of individual time steps and intervals that are to be excluded either entirely or only from evaluation, while permitted in model training. For example, when predicting future AKI, models should not be evaluated on a step-by-step basis while within an AKI episode, but there is still benefit in including these segments in model training since the models can learn about patterns of kidney injury progression and when it is likely to resolve or deteriorate further. Time intervals during which the patients are receiving renal replacement therapy (RRT) should be excluded both from training and testing splits. Surrogate entries are to be excluded in all cases (see Step 8). To exclude these identified time intervals, compute binary overlay *masks* associated with each step in each patient, where the value of 1 marks the particular time step as being included and the value of 0 marks the step as being excluded for purposes of loss and metrics computation. Note that, importantly, these time steps are not programmatically removed from the data, allowing the models to sequentially iterate over them and form memory representations of past clinical events correspond to these intervals. Similarly, these entries are factored in the historical feature representations in data preparation.

4. Assess dataset quality: Produce a formal specification of each data type present in the research dataset. Conduct structured interviews with experts with knowledge of the data to list known issues. Compute distributions of recorded blood test values in the EHR; compare the computed distributions to known physiological ranges. Validate admission records based on the recorded length of stay; identify admissions recorded as being within other admissions. Compile a random sample of the data for a closer clinical assessment of label quality; involve several clinical experts and have each expert evaluate each labelled example. Maintain a spreadsheet or database of all label assessments. Compute the variance in label assessments between different experts. Compute the percentage of ground truth labels produced by the primary labelling process marked as incorrect by being outside of the variance of the experts. Conduct meetings to reach clinical consensus on whether the computed percentage exceeds tolerable levels of labelling error. If the error level is deemed to be acceptable, proceed.

5. Select auxiliary prediction targets: Identify blood tests and vitals directly related to the clinical use case. Exclude rare types of measurements. Expose the values of these entries as auxiliary prediction targets, adding interpretability and helping to regularise developed ML models[9].

Data processing

6. Create data splits: Randomly assign each patient ID to one of the following data splits: *training*, *validation*, *calibration* and *test*. The minimum size of each split needs to be sufficient to derive valid statistical conclusions about the models and the data and should be based on an appropriate power calculation. In large datasets, assigning 80% of the data to the *training* split, 5% of the data to the *validation* split, 5% of the data to the *calibration* split and 10% of the data to the *test* split is a reasonable choice[20].

7. Feature filtering: Conduct interviews with clinical experts and the data experts that provided the research dataset to identify EHR entries that are not likely to be of value for the considered clinical use case. Compile a set of feature types that are not thought to be informative or are too specific to individual sites, if the dataset is compiled from multiple sites. Implement a script that removes the identified entries from the research dataset. Execute the script and produce a data version with those features removed.

8. Produce a discrete sequential EHR data representation: First, define the length of the atomic discrete time window. This can be defined as 6h[20] or 4h or any time window consistent with the granularity of the available data and the clinical use case. Select a time window that is a divisor of 24h. The inconsistencies in sequentiality of EHR entries limit the utility of very short atomic time windows. For each patient in all data splits, represent each day as a sequence of unordered sets of events that take place within equal-width time windows as defined above. For each day, provide an additional set to represent all the entries for which the date is known, but the timestamp is not present in the EHR; refer to this as the *surrogate bucket*. Sort the event sets chronologically and append the surrogate bucket at the end of each daily sequence. In cases when there are no EHR entries corresponding to a particular time window of a particular day for a particular patient record, insert an empty set. Compute the corresponding time for each event set and maintain a record of which time window corresponds to inpatient versus outpatient clinics. Concatenate all of the daily sequences into a full longitudinal representation for each patient in chronological order.

9. Shift EHR entries that were timestamped as preceding the events they encode: Based on an understanding of processes behind the EHR that were used to compile the research dataset, define a list of potentially useful entries that may have been entered out of order, prior to the time when they actually occurred. In the manuscript[20] this was the case with some of the diagnoses entries, which were timestamped at the beginning of admission even though the actual diagnosis might have been made at a slightly different time. To avoid leaking information from the future while retaining as much information as possible, move the entries in the sequential representation to a time window corresponding to the end of the current episode.

10. Compute aggregate historical features: Generate a set of historical time windows over which to compute historical aggregate features. In practice, 48 hours can be used for shorter history, and longer historical trends can be captured by considering 6 months, 1 year or 5 years prior. Define a set of statistical functions to use for feature aggregation of numerical values. We recommend using count, mean, median, standard deviation, minimum value, maximum value, difference between the last observed value and the minimum/maximum, and average difference between subsequent steps. For each patient, for each time step, list all of the prior entries that fall within the specified list of historical time windows. Compute feature-specific historical lists for each feature index for that step. Apply each of the selected statistical functions for feature aggregation to each of the numerical feature lists for each of the time steps for each patient. For non-numerical features, record a binary flag for whether they were or were not present in the interval. Record the computed values as separate historical features associated with each steps, assigning them a unique feature index. Perform this for each of the historical windows under consideration.

11. Compute manually-engineered features: Identify a set of manually-engineered features that may hold predictive value, based on a thorough literature review as well as domain knowledge provided by clinical experts. Examples include quantity ratios (e.g. ratio of blood urea nitrogen to serum creatinine), group features that correspond to sets of related EHR entries (e.g. a combined group feature derived from ICD-9 codes 250.42 and 250.4 as well as blood levels of haemoglobin A1c can help identify diabetic patients) and interaction terms. Implement functions that can be used to compute these features when applied to each step in the sequence. For each patient and each time step, compute the specified manually-engineered features.

12. Vectorise the event sequence: For each numerical entry, define the following features: the actual numerical value if available, a binary indicator of whether the feature was present in a particular time window, and a categorical variable corresponding to whether the quantity is considered to be normal, low or high, based on physiological ranges. For all numerical, binary and categorical features, reserve a unique index. For numerical and binary entries, for each patient and each step provide the *(index, value)* pair. For categorical features, for each patient and each step compute the one-hot encoding[42], representing each possible value as a separate binary feature. Convert all values to a floating point format, suitable for model development.

13. Normalise the numerical features: To improve convergence speed[43], normalise the input data features to be within the unit range 0-1. First, for every input feature compute the 1st and 99th percentile values on the training set. Using percentiles is preferable to using the minimum and maximum values, as these are more prone to data entry errors, resulting in physiologically implausible minimum/maximum entries in EHR data. For each value of each feature in each step for each patient, clip the value according to these percentiles as *value*=clip(*value*), where clip(*value*)=min(max(*value*, $percentile_1$), $percentile_{99}$). Proceed to normalise each of the clipped values by *value*=normalise(*value*), where normalise(*value*) = (*value* - $percentile_1$) / ($percentile_{99}$ - $percentile_1$).

14. Compute model targets: Define a list of future time windows for which to predict the expected primary and auxiliary target values, based on plausible clinical assumptions about the earliest possible time at which a condition might be predictable in certain clinical cases. For example, for AKI prediction[20], we considered future predictions up to 72 hours ahead of time, though we present the main results of future AKI prediction up to 48 hours ahead of AKI onset, based on considerations around model performance and the predictability of AKI. In that case, the future prediction time window list was defined as: 6, 12, 18, 24, 30, 36, 42, and 48 hours. For each patient and each step and for each future time window and each of the prediction tasks, compute the corresponding observed future label to be used for model development and evaluation. For binary targets the future label is to be 1 if the condition had occurred within the window and 0 if the condition did not occur within the window, for the given step. For numerical targets, define the statistics of interest and compute them across the future time windows for each of the numerical targets at each step.

15. Implement clinically relevant performance metrics: Define a set of relevant metrics both for the primary use case as well as the auxiliary prediction targets. For each, define separately a) model development metrics to be used for model architecture selection and b) final model evaluation metrics. For binary (non-numeric) targets, use area under the precision-recall curve (PR AUC) and the area under the receiver operating characteristic curve (ROC AUC) in model development. For final model evaluation, compute for each future prediction window the sensitivity for multiple levels of precision that correspond to different model operating points[44]. For numerical targets, compute the mean squared error (MSE) during model development. For final model evaluation, compute the percentage of correctly predicted substantial increases or decreases in value. Implement the relevant binary masks for step sub-selection according to the relevant stepwise inclusion and exclusion criteria, as per Step 3.

Model architecture selection

16. Compute standard statistical baseline models: Identify a set of statistical techniques commonly used in previous publications and protocols for a given clinical targets. These can include prediction models (e.g. Logistic Regression, Gradient Boosted Trees) and data processing techniques (e.g. Principal Component Analysis, Multiple Imputation). Estimate the parameters for those models and compute predictions for a given target. Estimate the performance for the models and use them for comparison during step 21.

17. Embed the inputs into a continuous feature space: For each timestep, transform the high-dimensional and sparse input features into a lower-dimensional continuous representation (i.e. embedding), that will be later used as an input to the sequential architecture. First, compute separate embeddings for separate types of input features: discrete features, continuous features, sequential and historical aggregations. Next, combine the resulting embedding vectors by concatenating them. Use a deep multilayer perceptron with residual connections and rectified-linear (ReLU) activations to capture the non-linear representation. Use L_1 regularisation on the embedding parameters to prevent overfitting and to ensure that the model focuses on the most salient features. Implement an optional reconstruction path with a reconstruction

loss at each step and a framework that supports the use of autoencoders (AEs) and variational autoencoders (VAEs).

18. Implement a deep architecture on top of input embeddings: Implement a recurrent and convolutional deep learning framework that sequentially takes the input embeddings as its input and provides a sequential output for each of the primary and auxiliary prediction targets, for each of the future prediction time windows. Make the frameworks configurable with respect to recurrent cell types and their parameters, as well as different types of convolutional kernels and architectures. Implement a batching and queuing process that provides the input embeddings sequentially to deep learning models.

19. Set up the model optimiser: Using the output modelled sequence and the ground truth sequence, compute a scalar loss value for each time step. Next, compute scalar losses for each auxiliary task in the same fashion. The loss function is dependent on the type of modelled target. In Tomašev et al[20] cross-entropy loss function (Bernoulli log-likelihood) was used for a binary adverse event classification, and L1/L2 losses for the auxiliary laboratory test regression. Optionally, re-weight the loss to account for skewed target distribution. Define a composite loss as a weighted sum of primary and auxiliary losses. Add the regularisation L1/L2 loss dependent on a subset of model weights that used to compute input embeddings. Initialise all the weights according to Xavier initialisation[45]. Use the computed loss alongside a mini-batch Adam[46] optimiser to iteratively adapt the weights of the neural network architecture, based on the computed gradients of the loss. Train using exponential learning rate decay on the training data split until convergence.

20. Test the validity of the implementation: Implement unit tests for all standalone components in the software for model implementation, as well as all components in data processing scripts. Resolve any outstanding issues and repeat the process until full code coverage is reached and no known issues are left unfixed. Implement integration and regression tests for the distributed data processing pipeline, as well as the model experimental framework. Resolve any issues that appear at the integration testing stage that were not previously identified in the unit testing stage. Repeat the process until no outstanding issues are left and the code is considered reliable.

21. Run an iterative sequence of hyperparameter sweeps: Decide on a desired level of predictive model performance that needs to be reached in order to imply potential clinical applicability. Define a set of potentially promising parameter values for each of the parameters in the model that is set to be configurable. Refer to this set as the set of hyperparameters in model development (see Table 7 for a list of hyperparameters tested in the manuscript[20]). Repeatedly perform hyperparameter sweeps aiming to establish good hyperparameter combinations, until a desired level of performance is reached or the maximum amount of time or resources is consumed. If the target performance is not reached, revisit and expand earlier steps on data processing and model implementation. In each hyperparameter sweep, formulate a hypothesis for which model architectural changes are likely to lead to performance improvements based on expert ML knowledge. In each sweep, generate a set of random combinations of hyperparameters corresponding to the current working hypothesis. For each of the hyperparameter

combinations, train a model on the training set to obtain a set of models. Evaluate all trained models on the validation set by computing the clinically relevant model development metrics. Select models based primarily on the clinically relevant metrics for the primary prediction task. Whilst using PR AUC and ROC AUC to select candidate hyperparameter configurations, prefer configurations that improve on both ROC AUC and PR AUC performance. When comparing configurations where configuration A achieves a higher PR AUC and configuration B a higher ROC AUC, prefer configuration A. Select one or more of the most promising hyperparameter configurations to move into the next sweep when exploring additional hypotheses. Repeating this iterative process over time should result in improving the model performance based on a refinement of design choices. Select the best performing configuration at the end of this process as the final model architecture at this stage.

22. Perform an ablation study: Take the final model architecture from the previous step and define a set of components to systematically remove to assess their contribution to the overall performance, i.e. to ablate. These will be components of the model that can either be simplified or removed. For example, this can include the number of stacked layers in the deep model architecture, additional feature input types like the historical aggregate features, regularisation approaches, auxiliary prediction tasks, layer width etc. For each ablation component, train a predictive model with the components removed or simplified on the training set. Next, evaluate each of the ablated models on the validation set, by computing the model development metrics. To test for statistical significance, train a collection of baseline and ablation models, each with sampled initial parameters, and calculate confidence intervals on the average performance. If any of the ablated models perform at least as well as the more complex final model at the end of the previous step, redefine the final model to be the best performing of the ablated models; reducing the model complexity. Repeat this process until the model can no longer be simplified without loss of performance.

23. Compute feature saliency: Perform occlusion analysis by evaluating the change in model risk when each feature is individually occluded, i.e. is removed or set as missing. The occluded value should equate to the value used during training when a feature is not present; for sparse feature sets this is simply zero. For non-sparse data with no missing values, the occluded value could be a mean-value for the given feature, or sampled from the marginal distribution of values for this feature[47]. By monitoring the change in risk for each feature present versus not present, sampled over many time steps and patients, we can compute the average effect on the risk per feature. This can be used to infer the saliency of features, and the direction of effect. Unlike metrics which determine saliency based upon the gradient of the risk with respect to each input[48], occlusion analysis is robust to different input features having different scales.

24. Failure case analysis: Compute patient-level and admission-level metrics for all patients and admissions in the validation set. Compile a set of representative success and failure cases of predictions made by the model, as well as a set of best success and worst failure cases made by the model. Assess the failure cases by having them reviewed by a set of clinical experts. Examine feature saliency at the point in time when incorrect predictions were made and investigate the root cause of the issues. Maintain a spreadsheet or a database of these assessments and at the end of the evaluation, establish if there

exist common causes of model failures. Assemble ML experts and clinical experts to propose potential solutions to reduce the number of model failures. If potential solutions are identified and judged as worth investigating, identify previous steps in the protocol that need adjusting and return to those steps. Repeat the entire protocol from those steps onwards, with the proposed adjustments. When reaching this step again, reassess and decide on whether to proceed.

25. Define the final model architecture: Define the resulting model architecture as final and do not revisit any of the previous steps at this point. Use the fixed set of parameters corresponding to this model to compute the predictions for all time steps in all patients for each data split.

Risk calibration and uncertainty

26. Train an ensemble of models for prediction uncertainty: To quantify the uncertainty of model predictions train an ensemble of multiple models with a fixed set of hyperparameters (corresponding to the final model) but different initial seeds, similar to Defauw et al[49]. To get the uncertainty ranges, trim distribution tails depending on the desired level of confidence.

27. Compute the uncertainty estimates: To gauge uncertainty on a trained model's performance, calculate confidence intervals of performance metrics using bootstrapping. First, sample the entire validation and test dataset with replacement (e.g. for 95% confidence intervals, take 200 samples). Resample entire patient history to conform with bootstrapping assumption of resampling independent events. Next, compute the pivot bootstrap estimator[50] using resampled values.

28. Re-calibrate the risk model: Use the previously computed calibration set to align the predicted values with the underlying probability of the adverse event occurring at a given time step. Fit an isotonic regression[51] model on the model predictions against the target variable on the calibration set. Assess the quality of the calibration by comparing uncalibrated predictions to recalibrated ones using Brier score[52] and reliability plots[53].

Model generalisability evaluation

29. Establish a risk score threshold for positive alerts: Performance metrics are dependent on the choice of an operating point. Evaluate precision and specificity for each time step and compute sensitivity for each individual AKI episode for each possible operating point on the validation set. Use that to examine the trade-off between increasing precision and decreasing sensitivity. Choose a specific operating point for a deployment strategy.

30. Analyse model performance across sub-populations: To identify types of patients or events where the model is applicable, define *a priori* a set of clinical subpopulations relevant to the modelled target. These can include demographic (age, gender, ethnicity, geography) and medical characteristics. In the manuscript[20] these included patients with chronic kidney injury (CKD), diabetes, or admissions resulting

in ICU transfer. Next, identify patients, admissions or individual time steps that fall into each type. Finally, compute the performance on each subpopulation.

31. Quantify the expected daily alert rate: Align all the predictions for all of the patient sequences from the test set in time. For each day in the longitudinal test set, compute the percentage of inpatients on that day where the model produced a true positive alert, the percentage of cases where the model produced a false positive alert without having provided a true positive alert, and the percentage of cases where the model did not produce any alerts. Compute the mean daily alert rate across all days in the longitudinal set for the total percentage of alerts as well as the true and false positive alerts. Report this metric to inform of the likely resource burden in future prospective evaluation.

32. Evaluate model generalisability on future unseen data (Figure 2): Choose a point in time t_P such that approximately 80% of data entries occur prior to t_P and approximately 20% occur after t_P. Create the same data splits as described in step 6. Train a model using the final architecture determined in step 25 using only data entries that occurred prior to t_P from the *training* split. Generate model predictions for the *test* split. Generate 95% confidence intervals of PR AUC for predictions made prior to t_P and for predictions made subsequent to t_P. Compare confidence intervals to determine if model performance on future unseen data is comparable to performance on historic unseen data.

33. Evaluate model generalisability in simulated cross-site deployments (Figure 3): Choose a split in hospital sites such that approximately 80% of patient admissions occur at sites in group *A* and approximately 20% occur at sites in group *B*. Create the same data splits as described in step 6. Train a model using the final architecture determined in step 25 using the *training* split, excluding data entries from admissions at sites in group *B*. Run inference to generate model predictions for the *test* split. Generate 95% confidence intervals of PR AUC for predictions made during admissions at sites in group *A* and for admissions at sites in group *B*. Compare confidence intervals to determine if model performance for unseen sites is comparable to performance for sites used during training.

34. Prospective evaluation: Prospectively evaluate the performance of the model in a real-world clinical environment. Initially, observational studies should seek to define (i) the feasibility of ingestion and processing of data, and the formulation and the delivery of predictions to clinicians in real time, (ii) how performance of the model is impacted by real-time deployment in specific clinical settings, (iii) the possible clinical and operational impacts of algorithm predictions on care delivered to patients, and (iv) when and how model predictions might best be presented to clinicians. Robust interventional studies with rigorous statistical methodology and analysis should then define the clinical, operational and economic impacts of deployment. Whilst detailing the optimal design of such evaluations falls outside the scope of this protocol, their conduct will be essential in demonstrating safety prior to widespread clinical use[54].

Troubleshooting

We identify three types of issues that may arise at any stage during the implementation and execution of the protocol: data issues, software implementation issues, problem definition issues.

Data issues: EHRs contain records of routinely collected data, and are therefore prone to data entry errors and inconsistencies. Additional errors could potentially be introduced in the process of exporting and automatically de-identifying the data prior to making it available for research. These errors can adversely affect the performance of developed predictive ML models. Ideally, all data issues would be uncovered early while performing the thorough quality assessment outlined in Step 4. Practically, it is not always possible to compile an exhaustive list of such issues beforehand, given the complexity of EHR data. There may be unanticipated issues that manifest themselves only upon a partial execution of the protocol. We would advise the researchers working on developing predicting models from EHR to remain vigilant and keep checking the integrity of the data throughout the model development, as well as maintain an open communication channel to the data partners and fully utilise the available clinical expertise.

Software implementation issues: The proposed protocol involves software engineering work to develop the software components used throughout its application, including the data processing framework, quality assurance (QA) scripts, input embedding modules, deep recurrent and convolutional model architectures, baselines, experimental framework, evaluation modules and performance metrics. Most steps in the protocol involve software engineering. Implementation errors in any of the steps could, if undetected and thus unresolved, invalidate the results. The entirety of the software implementation therefore needs to be thoroughly tested according to known best practices (see Step 20). Automated testing is complemented by careful case analysis (Step 24), which is critical in uncovering unanticipated implementation issues that may not have been explicitly tested. It is worth noting that implementation issues can manifest themselves in different ways and that these errors do not always lead to lower estimated model performance metrics. For example, errors in data processing could lead to data being sequentially presented to the models out of order, artificially enhancing the perceived predictive performance. Similarly, an incorrect implementation of the inclusion and exclusion criteria may manifest as an increase in the computed model performance, if there is an erroneous inclusion of cases where predictions are trivial or an erroneous exclusion of cases where predictions are very difficult. It is important to perform an in-depth analysis of every result throughout the execution of the protocol and to ensure that the software implementation is correct and yields reliable scientific conclusions that can be generalised and replicated.

Problem definition issues: Formalising the clinical use case (Step 1, Step 3) in ways that enable the software implementation of clinical target computations can involve choices that impact the efficacy of models developed by executing the proposed protocol. For example, consider the use case of predicting AKI and implementing clinical targets based on KDIGO[20]. First, KDIGO has been designed primarily for detection of AKI, and may not necessarily provide optimal tracking of its progression. This is due to the adjustments to the creatinine baseline that result from elevated measurements during an onset of AKI. Simply discounting any creatinine measurements for the purpose of baseline computation after an AKI

onset does not resolve the issue, in cases where there is lasting kidney damage that does not fully resolve after an acute injury – at least not without adding more rules to handle the edge cases. Second, KDIGO levels of AKI could be computed either directly from asynchronous measurements of creatinine at the points at which they become available or from imputed values of expected serum creatinine at each time step derived from subsequent creatinine measurements by applying either a polynomial interpolation technique or a Gaussian process model. There are pros and cons to each of the design choices and it is important to be aware of these limitations when working with a particular ground-truth definition. If in the course of executing the proposed protocol the limitations of the problem definition are seen as outweighing the benefits, the problem definition may need to be revisited, and subsequent steps in the protocol repeated.

Time Taken

The running of this protocol requires a multidisciplinary team, access to a research dataset and sufficient computational resources. The timings given are an approximation and will vary in practice.

Steps 1-5: ~2-4 weeks (depending on the complexity of the clinical use case this may be considerably longer)

Steps 6-13: ~4h

Steps 15-20: ~4-8 weeks, given access to sufficient engineering resources.

Step 21: ~2-3 days for one hyperparameter sweep, repeated multiple times until optimal performance is reached.

Step 22: ~2-3 days for each ablation, all of which can be executed simultaneously.

Step 23: ~2-3 days

Step 24:~1-2 weeks

Step 25: ~2-3 days

Step 26: ~2-3 days

Step 27: ~1-2 days

Step 28: ~3-4 hours

Step 29: ~3-4 hours

Step 30: ~3-4 hours

Step 31: ~1-2 hours

Step 32: ~2-3 days

Step 33: ~2-3 days

Step 34: The timing of a prospective study can vary considerably depending on the exact study design used.

Anticipated Results

Here we present the results obtained by applying the proposed protocol to the clinical use case of early AKI risk prediction[20]. The application of the protocol resulted in continuous risk models of future AKI capable of identifying 55.8% of all AKI cases up to 48h early, allowing for two false positives for one true positive. In terms of AKI requiring subsequent administration of dialysis within 90 days of the initial onset, the model was able to correctly predict 90.2% of such cases ahead of time. The model was additionally able to correctly identify substantial future increases in seven auxiliary biochemical tests in 88.5% of cases.

Here we focus on experiments performed during the model development process to highlight the comparisons that informed our choice of the final model presented in the manuscript[20]. These include the embedding and model comparisons, ablations, hyperparameter selection and model generalisability experiments.

Input embeddings

Finding good input feature embeddings (Step 17) was a key step in model development. Figure 4 gives a comparison of several input embedding architectures that we have explored in our experiments, when used along with a UGRNN[55] recurrent AKI risk model. A layer in a shallow or a deep embedding model is defined as follows: Let the input at time t be given as a collection of values $x_1, x_2, ..., x_{nt}$ corresponding to features $i_1, i_2, ..., i_{nt}$. An embedding layer is then defined as a mapping $f(sum(x_1\theta_{i1}, x_2\theta_{i2}, ..., x_{nt}\theta_{i\,nt}))$ where f is a configurable non-linearity and $\theta \in R^{n \times d}$ is the associated trainable weight matrix. A shallow architecture would consist of just a single layer, whereas a deeper architecture would incorporate several stacked layers. Residual connections[56] have shown to be helpful for optimising deep networks in other applications and we include them in our comparisons. Additionally, we evaluated using a reconstruction loss along in an autoencoder (AE) or variational autoencoder (VAE) architecture[57], to establish whether it can help with model generalisation by reducing overfitting[58,59]. The comparisons in Figure 4 revealed a significant benefit in using residual connections and two hidden layers, versus one. There were no observed benefits to using AEs and VAEs. The best embedding architecture outperformed the others in both PR AUC and ROC AUC. There were no significant benefits to very deep embedding architectures.

Model comparison

A comparison of different deep recurrent and convolutional architectures (Step 16) when used for predicting the risk of AKI within 48h is given in Figure 5, and has led to the following conclusions:

Recurrent neural networks (SRU[60,61], NTM[62], LSTM[63], MANN[64], DNC[65], UGRNN[55], GRU, Intersection RNN[55], RMC[66]) achieve the highest performance for both PR AUC and ROC AUC, with minimal differences between different cell types. They require the least amount of feature engineering as they achieve the highest performance using only the sequential information with the last 48h of patient history and a competitive performance with sequential information only, without additional historical inputs.

Feed-forward models (deep MLP, shallow MLP, Logistic Regression, Random Forest, Gradient Boosted Trees) achieve a significantly lower performance than the recurrent models, and require a more extensive set of inputs due to their inability to automatically keep track of patient history. The best results with the feed-forward models were achieved when using either 6 months or 5 years of historical information.

Giving more weight to positive examples during training to account for class imbalance was beneficial for training Gradient Boosted Trees (GBTs), but not for other models.

Tree-based methods were trained on one third of the data due to their inability to fit the entire dataset in memory. Our experiments suggest that this did not have a substantial adverse effect on their performance, as a further reduction by 40% would have lead to a decrease in ROC AUC and PR AUC of 0.2% and 0.8% respectively.

Ablation analysis

To simplify the model architecture and justify the final model that resulted from repeated hyperparameter tuning, we performed a series of ablations (Step 22) where we removed certain parts of the model and evaluated the model performance without those components (Figure 6). In all cases we saw a non-trivial reduction in performance when each of these components was removed. The removal of the auxiliary prediction loss and removal of regularisation resulted in some of the largest drops in model performance. We also compared models trained on only the sequential information to models augmented with historical features over short-term (last 48 hours) and long-term (last 6 months) time frames. The results are presented in Figure 7. The RNN model is able to aggregate information across time and there is a smaller difference in performance than for the logistic regression baseline which benefits heavily from hand-crafted historical features.

Generalisability on unseen future data

In a prospective deployment, the model would not have access to future data and would be given access to the entirety of historical data. It is therefore important for the trained risk models to display generalisability across time, on future unseen data. Prior to a prospective study, model generalisability can be evaluated retrospectively as described in Step 32. Figure 8 demonstrates that in the case of the

developed AKI risk models, there was no substantial difference in performance when evaluated on future previously unseen data after the cutoff point t_P.

Despite potential differences to an actual live deployment, for example integration challenges that cannot be simulated here, the results do suggest that an actual prospective deployment of our model may be feasible and opens up an exciting opportunity for future prospective studies that would help determine whether early predictive AKI alerts can help improve clinical outcomes.

Generalisability across different sites

Different hospitals may implement different clinical pathways and processes based on the available resources and the patient population and local demographics. If a model is to be deployed to a previously unseen hospital site, a level of robustness to this local variation is required. Step 33 in the protocol corresponds to this aspect of generalisability testing. To test the robustness of the proposed model we have introduced a data split across sites so that 80% of the data was in group *A* and 20% in group *B*. No site from group *B* was present in group *A* and vice versa. The data was split into *training*, *validation*, *calibration* and *test* in the same way as in the other experiments. Training data from site group *A* was used for model development. The models were evaluated on test set patients in both site groups. The results from the simulated cross-site deployment (Figure 9) suggest an absence of a substantial difference in performance across the key metrics. Comparable performance on patients not previously seen, and at sites not previously trained on suggests that the AKI risk models developed by using the proposed protocol can generalise well across the population of the hospital system that provided the data for the study.

Hyperparameter selection

The presented results are achieved by considering the following set of developed and evaluated model architecture choices and hyperparameter combinations outlined in Figure 10. Applying the protocol to building AKI risk models on comparably large datasets of similar patient demographics should yield comparable results in terms of the expected model performance when predicting the risk of future AKI. As discussed in ”Applications of the method” in the introduction, with minor adjustments the protocol could potentially be used to develop continuous risk models for other adverse conditions in inpatient care. Future studies will need to confirm this assumption and a prospective validation of the demonstrated retrospective model performance will be key to assessing the clinical utility of the developed continuous risk models.

References

[1] R. Thomson, D. Luettel, F. Healey, and S. Scobie, “Safer care for the acutely ill patient: Learning from serious incidents,” *National Patient Safety Agency*, 2007.

[2] K. E. Henry, D. N. Hager, P. J. Pronovost, and S. Saria, “A targeted real-time early warning score (trewscore) for septic shock,” *Science Translational Medicine*, vol. 7, no. 299, pp. 299ra122–299ra122, 2015.

[3] A. Rajkomar, E. Oren, K. Chen, A. M. Dai, N. Hajaj, M. Hardt, P. J. Liu, X. Liu, J. Marcus, M. Sun, P. Sundberg, H. Yee, K. Zhang, Y. Zhang, G. Flores, G. E. Duggan, J. Irvine, Q. Le, K. Litsch, A. Mossin, J. Tansuwan, D. Wang, J. Wexler, J. Wilson, D. Ludwig, S. L. Volchenboum, K. Chou, M. Pearson, S. Madabushi, N. H. Shah, A. J. Butte, M. Howell, C. Cui, G. Corrado, and J. Dean, “Scalable and accurate deep learning with electronic health records,” *NPJ Digital Medicine*, vol. 1, no. 1, 2018.

[4] J. L. Koyner, R. Adhikari, D. P. Edelson, and M. M. Churpek, “Development of a multicenter ward based AKI prediction model,” *Clinical Journal of the American Society of Nephrology*, pp. 1935–1943, 2016.

[5] P. Cheng, L. R. Waitman, Y. Hu, and M. Liu, “Predicting inpatient acute kidney injury over different time horizons: How early and accurate?,” in *AMIA Annual Symposium Proceedings*, vol. 2017, p. 565, American Medical Informatics Association, 2017.

[6] J. L. Koyner, K. A. Carey, D. P. Edelson, and M. M. Churpek, “The development of a machine learning inpatient acute kidney injury prediction model,” *Critical Care Medicine*, vol. 46, no. 7, pp. 1070–1077, 2018.

[7] M. Komorowski, L. A. Celi, O. Badawi, A. Gordon, and A. Faisal, “The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care,” *Nature Medicine*, vol. 24, pp. 1716–1720, 2018.

[8] A. Avati, K. Jung, S. Harman, L. Downing, A. Y. Ng, and N. H. Shah, “Improving palliative care with deep learning,” *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 311–316, 2017.

[9] B. Lim and M. van der Schaar, “Disease-Atlas: Navigating disease trajectories with deep learning,” *Proceedings of Machine Learning Research*, vol. 85, 2018.

[10] J. Futoma, S. Hariharan, and K. A. Heller, “Learning to detect sepsis with a multitask gaussian process RNN classifier,” in *Proceedings of the International Conference on Machine Learning* (D. Precup and Y. W. Teh, eds.), pp. 1174–1182, 2017.

[11] P. Nguyen, T. Tran, N. Wickramasinghe, and S. Venkatesh, “Deepr: A convolutional net for medical records,” *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 1, pp. 22–30, 2017.

[12] R. Miotto, L. Li, B. Kidd, and J. T. Dudley, “Deep Patient: An unsupervised representation to predict the future of patients from the electronic health records,” *Scientific Reports*, vol. 6, no. 26094, 2016.

[13] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzel, “Learning to diagnose with LSTM recurrent neural networks,” *International Conference on Learning Representations*, 2016.

[14] P. Z. J. H. Yu Cheng, Fei Wang, “Risk prediction with electronic health records a deep learning approach,” in *Proceedings of the SIAM International Conference on Data Mining*, 850pp. 432–440, 2016.

[15] H. Soleimani, A. Subbaswamy, and S. Saria, “Treatment-response models for counter-factual reasoning with continuous-time, continuous-valued interventions,” *arXiv Preprint arXiv:1704.02038*, 2017.

[16] A. M. Alaa, J. Yoon, S. Hu, and M. van der Schaar, “Personalized risk scoring for critical care patients using mixtures of gaussian process experts,” *arXiv Preprint arXiv:1605.00959*, 2016.

[17] A. Perotte, N. Elhadad, J. S. Hirsch, R. Ranganath, and D. Blei, “Risk prediction for chronic kidney disease progression using heterogeneous electronic health record data and time series analysis,” *Journal of the American Medical Informatics Association*, vol. 22, no. 4, pp. 872–880, 2015.

[18] A. Bihorac, T. Ozrazgat-Baslanti, A. Ebadi, A. Motaei, M. Madkour, P. M. Pardalos, G. Lipori, W. R. Hogan, P. A. Efron, F. Moore, et al., “MySurgeryRisk: Development and validation of a machine-learning risk algorithm for major complications and death after surgery,” *Annals of Surgery*, 2018.

[19] A. E. W. Johnson, M. M. Ghassemi, S. Nemati, K. E. Niehaus, D. A. Clifton, and G. D. Clifford, “Machine learning and decision support in critical care,” *Proceedings of the IEEE*, vol. 104, no. 2, pp. 444–466, 2016.

[20] N. Tomasev, X. Glorot, J. W. Rae, M. Zielinski, H. Askham, A. Saraiva, A. Mottram, C. Meyer, S. Ravuri, I. Protsyuk, A. Connell, C. O. Hughes, A. Karthikesalingam, J. Cornebise, H. Montgomery, G. Rees, C. Laing, C. R. Baker, K. Peterson, R. Reeves, D. Hassabis, D. King, M. Suleyman, T. Back, C. Nielson, J. R. Ledsam, and S. Mohamed, “A clinically applicable approach to the continuous prediction of future acute kidney injury,” *Nature*, 2019.

[21] H. E. Wang, P. Muntner, G. M. Chertow, and D. G. Warnock, “Acute kidney injury and mortality in hospitalized patients,” *American Journal of Nephrology*, vol. 35, pp. 349–355, 2012.

[22] M. Kerr, M. Bedford, B. Matthews, and D. O’Donoghue, “The economic impact of acute kidney injury in England,” *Nephrology Dialysis Transplantation*, vol. 29, no. 7, pp. 1362–1368, 2014.

[23] A. MacLeod, “NCEPOD report on acute kidney injury—must do better,” *The Lancet*, vol. 374, no. 9699, pp. 1405–1406, 2009.

[24] A. Khwaja, “KDIGO clinical practice guidelines for acute kidney injury,” *Nephron Clinical Practice*, vol. 120, no. 4, pp. c179–c184, 2012.

[25] F. P. Wilson, M. G. S. Shashaty, J. M. Testani, I. Aqeel, Y. Borovskiy, S. S. Ellenberg, H. I. Feldman, H. E. Fernandez, Y. Gitelman, J. Lin, D. Negoianu, C. R. Parikh, P. P. Reese, R. Urbani, and B. D. Fuchs, “Automated, electronic alerts for acute kidney injury: a single-blind, parallel-group, randomised controlled trial,” *The Lancet*, vol. 385, pp. 1966–1974, 2015.

[26] S. J. Weisenthal, C. M. Quill, S. A. Farooq, H. A. Kautz, and M. S. Zand, “Predicting acute kidney injury at hospital re-entry using high dimensional electronic health record data,” *arXiv Preprint arXiv:1807.09865*, 2018.

[27] R. M. Cronin, J. P. VanHouten, E. D. Siew, S. K. Eden, S. D. Fihn, C. D. Nielson, J. F. Peterson, C. R. Baker, T. A. Ikizler, T. Speroff, and M. E. Matheny, “National Veterans Health Administration inpatient risk stratification models for hospital acquired acute kidney injury,” *Journal of the American Medical Informatics Association*, vol. 22, no. 5, pp. 1054–1071, 2015.

[28] S. J. Weisenthal, H. Liao, P. Ng, and M. S. Zand, “Sum of previous inpatient serum creatinine measurements predicts acute kidney injury in rehospitalized patients,” *arXiv Preprint arXiv:1712.01880*, 2017.

[29] H. Mohamadlou, A. Lynn-Palevsky, C. Barton, U. Chettipally, L. Shieh, J. Calvert, N. R. Saber, and R. Das, “Prediction of acute kidney injury with a machine learning algorithm using electronic health record data,” *Canadian Journal of Kidney Health And Disease*, vol. 5, 2018.

[30] Z. Pan, H. Du, K. Yuan Ngiam, F. Wang, P. Shum, and M. Feng, “A self-correcting deep learning approach to predict acute conditions in critical care,” *arXiv Preprint arXiv:1901.04364*, 2019.

[31] M. Singer, C. S. Deutschman, C. W. Seymour, M. Shankar-Hari, D. Annane, M. Bauer, R. Bellomo, G. R. Bernard, J.-D. Chiche, C. M. Coopersmith,et al., “The third international consensus definitions for sepsis and septic shock (sepsis-3),” *Jama*, vol. 315, no. 8, pp. 801–810, 2016.

[32] C. Rhee, R. Dantes, L. Epstein, D. J. Murphy, C. W. Seymour, T. J. Iwashyna, S. S. Kadri, D. C. Angus, R. L. Danner, A. E. Fiore,et al., “Incidence and trends of sepsis in us hospitals using clinical vs claims data, 2009-2014,” *Jama*, vol. 318, no. 13, pp. 1241–1249, 2017.

[33] P. Yadav, L. Pruinelli, A. Hoff, M. Steinbach, B. L. Westra, V. Kumar, and G. J. Simon, “Causal inference in observational data,” *CoRR*, vol. abs/1611.04660, 2016.

[34] Department of Veterans Affairs, “Veterans Health Administration: Providing health care for Veterans.”https://www.va.gov/health/, 2018 (accessed November 9, 2018).

[35] J. D. Hunter, “Matplotlib: A 2d graphics environment,”

Computing in Science & Engineering, vol. 9, no. 3, pp. 90–95, 2007.

[36] T. Oliphant, “NumPy: A guide to NumPy.”

http://www.numpy.org/, 2019 (accessed June 10, 2019).

[37] E. Jones, T. Oliphant, P. Peterson, et al., “SciPy: Open source scientific tools for Python.” http://www.scipy.org/, 2019 (accessed June 10, 2019).

[38] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M.Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[39] M. Reynolds, G. Barth-Maron, F. Besse, D. de Las Casas, A. Fidjeland, T. Green, A. Puigdomènech, S. Racanière, J. Rae, and F. Viola, "Open sourcing Sonnet - a new library for constructing neural networks." https://deepmind.com/blog/open-sourcing-sonnet/, 2017.

[40] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Van-houcke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," *arXiv Preprint arXiv:1603.04467,*

2015. Software available from tensorflow.org.

[41] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and DataMining, KDD '16*, (New York, NY, USA), pp. 785–794, ACM, 2016.

[42] D. B. Suits, "Use of dummy variables in regression equations," *Journal of the American Statistical Association*, vol. 52, no. 280, pp. 548–551, 1957.

[43] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop*, (London, UK), pp. 9–50, Springer-Verlag, 1998.

[44] S. Romero-Brufau, J. M. Huddleston, G. J. Escobar, and M. Liebow, "Why the c-statistic is not informative to evaluate early warning scores and what metrics to use," *Critical Care*, vol. 19, no. 1, p. 285, 2015.

[45] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feed forward neural networks," in *International Conference on Artificial Intelligence and Statistics* (Y. W. Teh and M. Titterington, eds.), vol. 9, pp. 249–256, 2010.

[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 2015.

[47] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *European Conference on Computer Vision*, 2014.

[48] J. M. Steppe and K. W. Bauer Jr, “Feature saliency measures,” *Computers & Mathematics With Applications*, vol. 33, no. 8, pp. 109–126, 1997.

[49] J. D. Fauw, J. R. Ledsam, B. Romera-Paredes, S. Nikolov, N. Tomasev, S. Blackwell, H. Askham, X. Glorot, B. O’Donoghue, D. Visentin, G. van den Driessche, B. Lakshminarayanan, C. Meyer, F. Mackinder, S. Bouton, K. W. Ayoub, R. Chopra, D. King, A. Karthikesalingam, C. O. Hughes, R. A. Raine, J. C. Hughes, D. A. Sim, C. A. Egan, A. Tufail, H. Montgomery, D. Hassabis, G. Rees, T. Back, P. T. Khaw, M. Suleyman, J. Cornebise, P. A. Keane, and O. Ronneberger, “Clinically applicable deep learning for diagnosis and referral in retinal disease,” *Nature Medicine*, vol. 24, pp. 1342–1350, 2018.

[50] B. Efron and R. J. Tibshirani, An introduction to the bootstrap. *CRC press*, 1994.

[51] B. Zadrozny and C. Elkan, “Transforming classifier scores into accurate multiclass probability estimates,” in *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 694–699, ACM, 2002.

[52] G. W. Brier, “Verification of forecasts expressed in terms of probability,” *Monthly Weather Review*, vol. 78, no. 1, pp. 1–3, 1950.

[53] A. Niculescu-Mizil and R. Caruana, “Predicting good probabilities with supervised learning,” in *Proceedings of the International Conference on Machine Learning* (L. D. Raedt and S. Wrobel, eds.), pp. 625–632, ACM, 2005.

[54] E. J. Topol, “High-performance medicine: the convergence of human and artificial intelligence,” *Nat. Med.*, vol. 25, pp. 44–56, Jan 2019.

[55] J. Collins, J. Sohl-Dickstein, and D. Sussillo, “Capacity and learnability in recurrent neural networks,” *International Conference on Learning Representations*, 2017.

[56] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[57] D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” *International Conference on Learning Representations*, 2014.

[58] B. Shickel, P. J. Tighe, A. Bihorac, and P. Rashidi, “Deep ehr: A survey of recent advances in deep learning techniques for electronic health record (EHR) analysis,” *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 5, pp. 1589–1604, 2018.

[59] M. Z. Nezhad, D. Zhu, N. Sadati, and K. Yang, “A predictive approach using deep feature learning for electronic medical records: A comparative study,” *arXiv Preprint arXiv:1801.02961*, 2018.

[60] J. Bradbury, S. Merity, C. Xiong, and R. Socher, “Quasi-recurrent neural networks,” *International Conference on Learning Representations*, 2017.

[61] T. Lei and Y. Zhang, “Training RNNs as fast as CNNs,” *arXiv Preprint arXiv:1709.02755*, 2017.

[62] A. Graves, G. Wayne, and I. Danihelka, “Neural turing machines,” *arXiv Preprint arXiv:1410.5401*, 2014.

[63] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[64] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, “Meta-learning with memory-augmented neural networks,” in *Proceedings of the International Conference on Machine Learning* (M. F. Balcan and K. Q. Weinberger, eds.), pp. 1842–1850, 2016.

[65] A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwi´nska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou,et al., “Hybrid computing using a neural network with dynamic external memory,” *Nature*, vol. 538, no. 7626, pp. 471–476, 2016.

[66] A. Santoro, R. Faulkner, D. Raposo, J. Rae, M. Chrzanowski, T. Weber, D. Wierstra, O. Vinyals, R. Pascanu, and T. Lillicrap, “Relational recurrent neural networks,” *arXiv Preprint arXiv:1806.01822*, 2018.

[67] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics* (G. Gordon, D. Dunson, and M. Dudík, eds.), vol. 15, pp. 315–323, PMLR, 2011.

[68] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” in *Proceedings of the International Conference on Machine Learning* (S. Dasgupta and D. McAllester, eds.), vol. 30, p. 3, 2013.

[69] P. Ramachandran, B. Zoph, and Q. V. Le, “Searching for activation functions,” *International Conference on Learning Representations*, 2018.

[70] D. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and accurate deep network learning by exponential linear units (ELUs),” *International Conference on Learning Representations*, 2016.

[71] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, “Self-normalizing neural networks,” in *Advances in Neural Information Processing Systems* (I. Guyon, U. Luxburg,1000S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, pp. 971–980, 2017.

[72] M. Basirat and P. M. Roth, “The quest for the golden activation function,” *arXiv Preprint arXiv:1808.00783*, 2018.

Acknowledgements

We thank the Veterans and their families under the care of the US Department of Veterans Affairs. We would also like to thank A. Graves, O. Vinyals, K. Kavukcuoglu, S. Chiappa, T. Lillicrap, R. Raine, P. Keane,

A. Schlosberg, O. Ronneberger, J. De Fauw, K. Ruark, M. Jones,J. Quinn, D. Chou, C. Meaden, G. Screen, W. West, R. West, P. Sundberg and the Google AI team, J. Besley, M. Bawn, K. Ayoub and R. Ahmed. Finally, we thank the many VA physicians, administrators and researchers who worked on the data collection, and the rest of the DeepMind team for their support, ideas and encouragement.

G.R. & H.M. were supported by University College London and the National Institute for Health Research (NIHR) University College London Hospitals Biomedical Research Centre. The views expressed are those of these author(s) and not necessarily those of the NHS, the NIHR or the Department of Health.

Figures

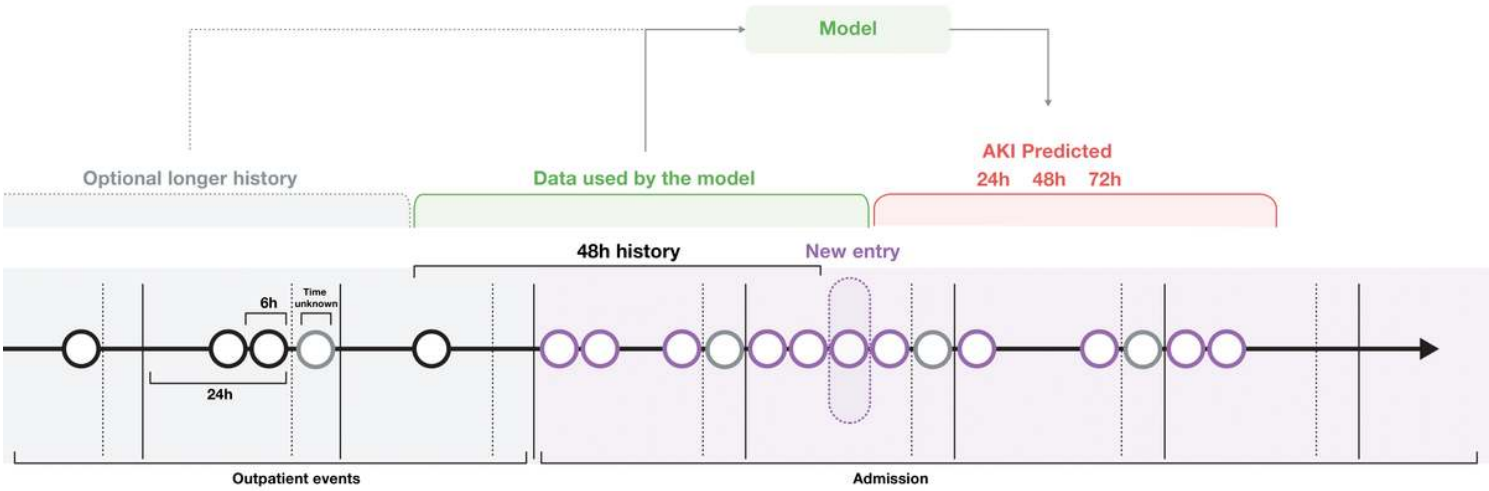

Figure 1

Sequential risk prediction from EHR data: a case of AKI. A sequential framework for making predictions from EHR data in fixed, discrete steps, each shown as circles and corresponding to 6 hours each. In each 24 hour period, events without a recorded timestamp within EHR entries were included in a fifth, surrogate block. The models compute the risk of future adverse events.

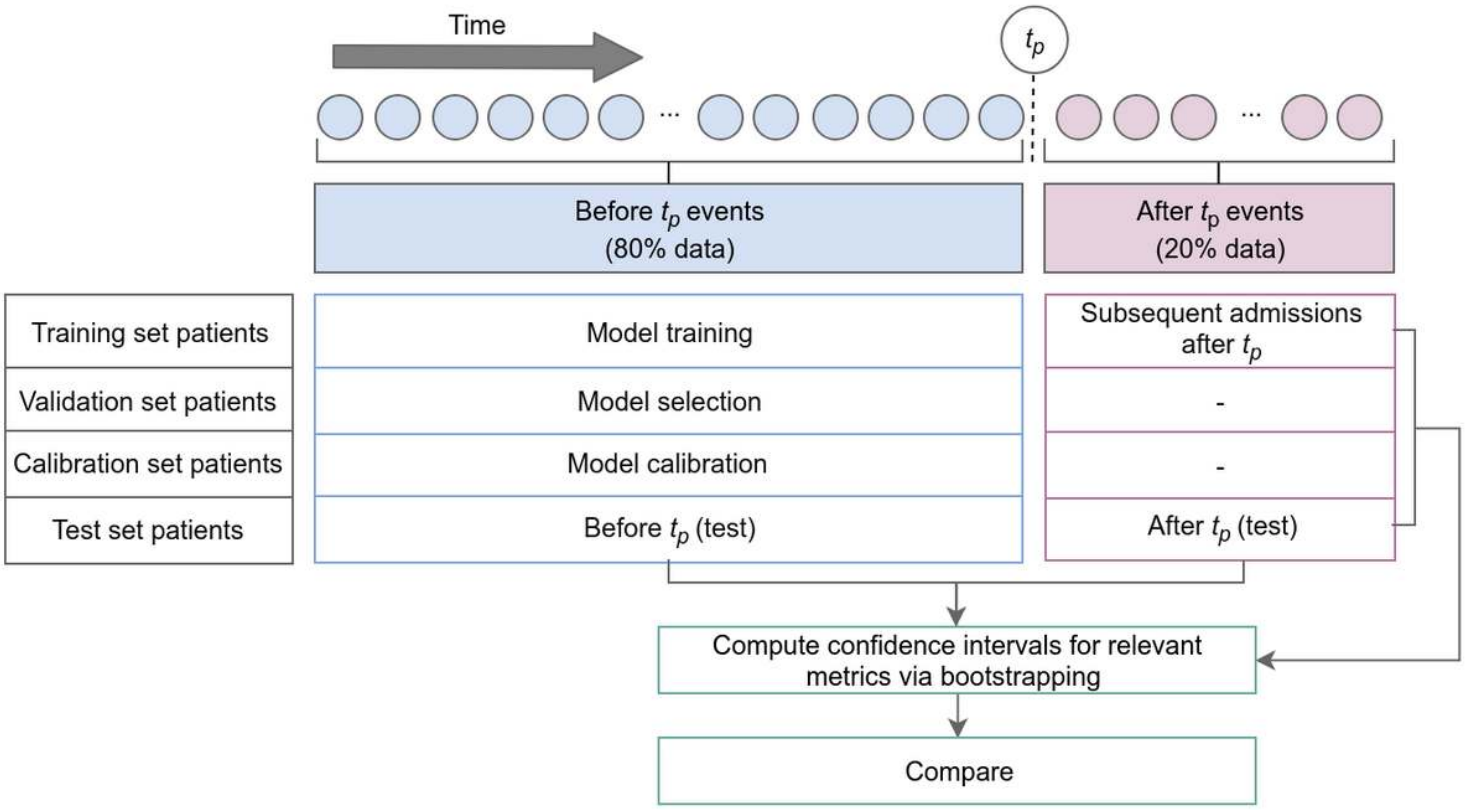

Figure 2

Diagram of Step 32: evaluating model generalisability on future unseen data. Confidence intervals for performance metrics on the test split prior to the time point tP are compared to those on the test split after tP to determine if performance is preserved on future unseen data. Confidence intervals for performance metrics on the train split after tP are compared to those on the test split after tP to determine if there is a benefit from having prior historical data on patients present during model training.

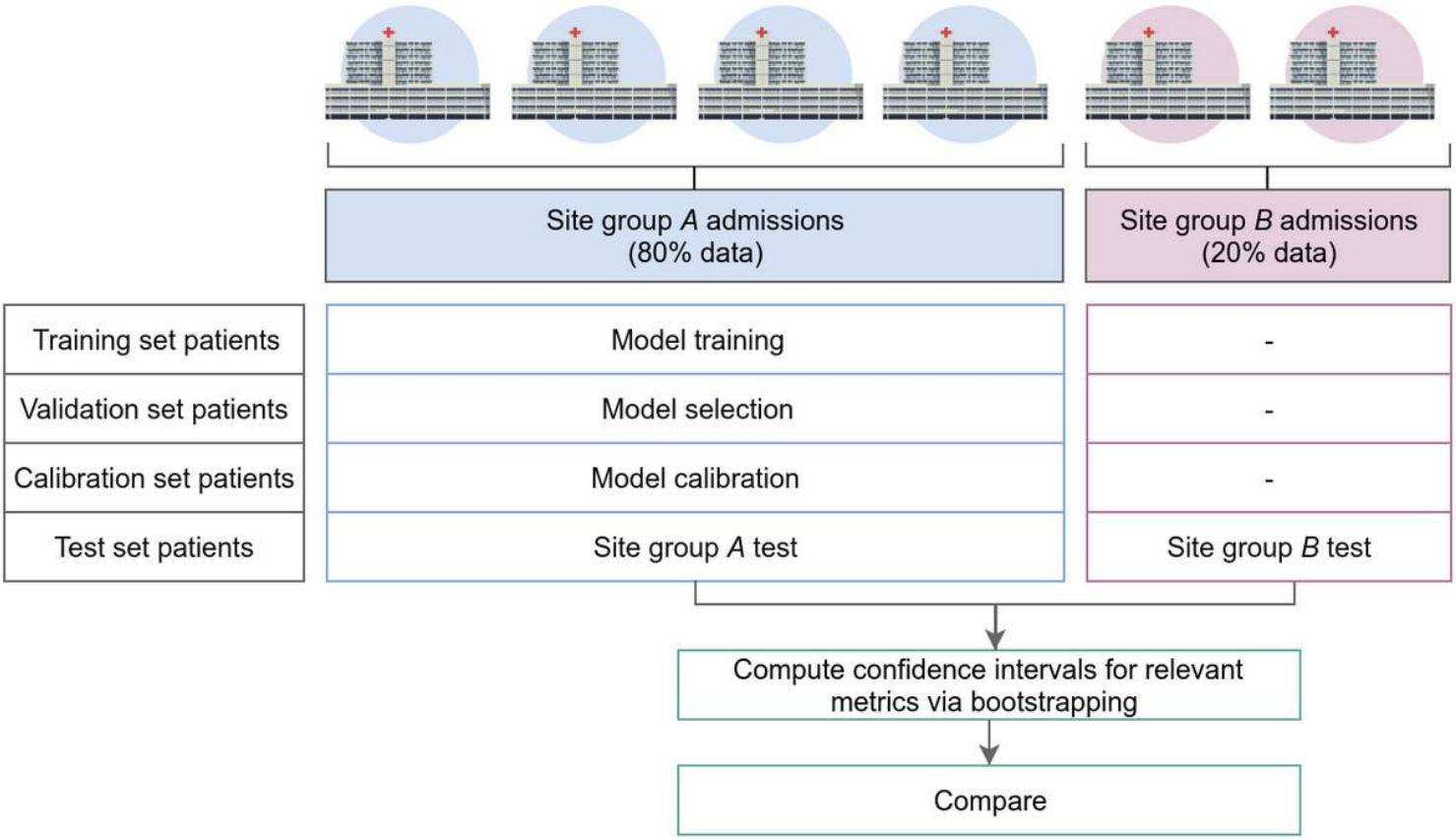

Figure 3

Diagram of Step 33: evaluating model generalisability in simulated cross-site deployments. Confidence intervals for performance metrics on test split predictions made during admissions at sites in group A are compared to those for test split predictions made during admissions at sites in group B to determine if performance is preserved at sites that were unseen during training.

Embedding Architecture	ROC AUC (%) [95% CI]	PR AUC (%) [95% CI]
1 Layer	90.5 [90.3, 90.8]	23.4 [22.3, 24.5]
2 Layers	90.9 [90.7, 91.1]	25.5 [24.4, 26.5]
2 Layers + Residuals	**91.8 [91.6, 92.0]**	**28.8 [27.7, 29.9]**
2 Layers + Residuals Variational Autoencoder	90.4 [90.1, 90.6]	25.2 [24.0, 26.4]
2 Layers + Residuals Autoencoder	91.1 [90.9, 91.4]	26.4 [25.3, 27.4]

Figure 4

Comparison of feature embedding architectures on a UGRNN cell. Using Mann–Whitney U test on 200 samples per model residual architecture significantly outperforms other models predicting the risk of AKI within 48h in terms of PR AUC (p-value of <0.001).

AKI task	Model	PR AUC (%) [95% CI]	ROC AUC (%) [95% CI]
Any AKI up to 48 hours early	SRU	29.7 [28.5, 30.8]	92.1 [91.9, 92.3]
	Intersection RNN	29.6 [28.5, 30.7]	91.9 [91.7, 92.1]
	NTM	29.0 [27.6, 30.0]	91.9 [91.5, 91.9]
	MANN	28.9 [27.8, 30.0]	92.0 [91.8, 92.2]
	LSTM	28.8 [27.7, 30.0]	92.1 [91.8, 92.2]
	UGRNN	28.3 [27.2, 29.5]	91.9 [91.7, 92.1]
	GRU	27.8 [26.7, 28.8]	92.0 [91.8, 92.2]
	RMC	26.2 [25.0, 27.3]	91.3 [91.1, 91.5]
	DNC	26.5 [25.4, 27.4]	91.9 [91.7, 92.1]
	Deep MLP	25.1 [23.9, 26.1]	90.3 [90.0, 90.6]
	CNN	23.8 [22.8, 24.8]	90.1 [89.9, 90.4]
	Shallow MLP	22.3 [21.1, 23.2]	89.9 [89.6, 90.1]
	Gradient Boosted Trees*	22.0 [21.0, 22.9]	88.9 [88.6, 89.2]
	Random Forest*	19.8 [18.8, 20.9]	87.1 [86.7, 87.4]
	Logistic Regression*	17.3 [16.2, 18.2]	86.3 [86.0, 86.7]
AKI stages 2 and 3 up to 48 hours early	Intersection RNN	37.8 [35.7, 40.0]	95.7 [95.5, 96.0]
	UGRNN	37.3 [35.1, 39.2]	95.6 [95.3, 95.9]
	LSTM	37.1 [35.4, 39.1]	95.5 [95.2, 95.8]
	NTM	36.9 [35.1, 39.0]	95.5 [95.2, 95.7]
	GRU	36.2 [34.2, 38.1]	95.5 [95.2, 95.8]
	MANN	36.2 [34.6, 38.1]	95.4 [95.1, 95.7]
	DNC	35.7 [33.6, 37.5]	95.5 [95.2, 95.8]
	Deep MLP	32.2 [30.2, 33.9]	94.9 [94.5, 95.2]
	SRU	29.0 [27.1, 30.6]	94.7 [94.4, 95.0]
	CNN	27.2 [25.3, 28.9]	94.3 [93.9, 94.6]
	Shallow MLP	25.3 [23.9, 26.8]	93.7 [93.4, 94.1]
	Gradient Boosted Trees	25.1 [23.3, 26.8]	92.5 [92.2, 92.9]
	Random Forest	25.1 [22.9, 26.6]	91.1 [90.6, 91.5]
	RMC	21.9 [20.5, 23.2]	91.1 [90.6, 91.6]
	Logistic Regression	16.7 [15.2, 18.1]	87.0 [86.3, 87.6]
AKI stage 3 up to 48 hours early	NTM	48.7 [46.4, 51.1]	98.0 [97.8, 98.2]
	MANN	47.9 [45.8, 50.0]	98.0 [97.7, 98.1]
	Intersection RNN	47.8 [45.3, 50.2]	98.0 [97.8, 98.2]
	GRU	47.5 [45.6, 49.9]	98.0 [97.8, 98.2]
	UGRNN	47.1 [45.1, 49.1]	98.1 [97.9, 98.2]
	LSTM	46.8 [44.7, 49.3]	98.0 [97.8, 98.2]
	SRU	46.6 [44.4, 48.9]	98.0 [97.8, 98.2]
	DNC	45.0 [42.0, 47.5]	97.8 [97.6, 98.0]
	Deep MLP	40.9 [38.8, 42.9]	97.5 [97.3, 97.8]
	CNN	38.8 [36.8, 41.0]	97.3 [97.1, 97.5]
	Random Forest	34.6 [31.9, 37.2]	95.5 [95.2, 95.9]
	Gradient Boosted Trees	32.9 [30.9, 35.0]	96.2 [95.9, 96.5]
	Shallow MLP	32.7 [30.8, 34.6]	96.7 [96.4, 96.9]
	RMC	24.7 [22.2, 26.4]	93.8 [93.3, 94.3]
	Logistic Regression	24.5 [23.1, 25.9]	93.0 [92.5, 93.6]

Figure 5

Comparison of different predictive models and RNN cells. SRU significantly outperforms the Logistic Regression, Gradient Boosted Trees and Random Forest baselines in terms of PR AUC for the main task of predicting any AKI up to 48 hours ahead of time; using two-sided Mann–Whitney U test on 200 samples per model SRU is significantly better with a p-value of <0.001.

PR AUC	SRU	MLP
Full model	29.7 ± 1.2	25.1 ± 1.1
Shallow model	23.1 ± 0.7	22.9 ± 0.1
Without regularisation	22.5 ± 1.3	23.3 ± 0.1
Without auxiliary regression	26.6 ± 1.4	24.3 ± 0.1
Without numerical features	20.6 ± 0.6	16.7 ± 0.5
Without presence features	22.4 ± 0.9	18.6 ± 0.2

Figure 6

AKI predictive model performance with ablations. Performance is expressed in PR AUC. We compare the performance for a recurrent model (SRU) and feed-forward model (MLP) on predicting any AKI within 48 hours. 95% confidence intervals are calculated from an un-paired z-test, with 50 models trained from random initialisation per configuration.

PR AUC [95% CI]	Intersection RNN	Logistic Regression
Sequential information only	28.5 [27.3, 29.4]	14.7 [13.9, 15.4]
Sequential + historical aggregations	28.7 [27.5, 29.7]	17.3 [16.3, 18.1]

Figure 7

AKI predictive model performance with and without historical features. PR AUC performance for models using sequential and short-term information and optionally being augmented with long-term history aggregation.

	Patient cohorts			
Metric [95% CI]	Before t_P (test)	New admissions after t_P (test)	Subsequent admissions after t_P	All patients after t_P
Sensitivity (AKI episode)	55.09 [54.01, 56.06]	59 [57.11, 60.71]	59.04 [58.38, 59.63]	58.97 [58.33, 59.52]
ROC AUC	92.25 [92.01, 92.42]	90.19 [89.76, 90.77]	89.98 [89.83, 90.17]	89.98 [89.81, 90.14]
PR AUC	29.97 [28.61, 31.15]	30.75 [28.65, 32.81]	31.54 [30.87, 32.30]	31.28 [30.44, 32.02]
Sensitivity (step)	34.26 [33.17, 35.28]	36.87 [35.2, 38.85]	37.23 [36.67, 37.88]	37.08 [36.40, 37.65]
Specificity (step)	98.55 [98.50, 98.60]	97.66 [97.54, 97.76]	97.63 [97.58, 97.68]	97.64 [97.59, 97.68]
Precision	32.51 [31.44, 33.21]	32.66 [31.2, 34.03]	32.97 [32.52, 33.47]	32.84 [32.28, 33.33]

Figure 8

Generalisability to future data. Performance of the AKI risk model when trained before the time point tP and tested after tP, both on the entirety of the future patient population as well as subgroups of patients for which the model has or has not seen historical information during training. The model maintains a comparable level of performance on unseen future data, with a higher level of sensitivity of 59% for a time window of 48 hours ahead of time and a precision of two false positives per step for each true positive. Note that this experiment is not a replacement for a prospective evaluation of the model.

Metric [95% CI]	Site group *A*	Site group *B*
Sensitivity (AKI episode)	55.6% [54.5, 56.6]	54.6% [52.8, 56.3]
ROC AUC	91.8% [91.6, 92.1]	91.3% [90.8, 91.7]
PR AUC	30.0% [28.6, 31.2]	30.6% [28.3, 32.7]
Sensitivity (step)	34.3% [33.1, 35.2]	34.7% [32.6, 36.2]
Specificity (step)	98.5% [98.4, 98.5]	98.3% [98.2, 98.4]

Figure 9

Cross-site generalisability. Comparison of model performance when applied to data from previously unseen hospital sites. Data was split across sites so that 80% of the data was in group A and 20% in group B. No site from group B was present in group A and vice versa. The data was split into training, validation, calibration and test in the same way as in the other experiments. The table reports model performance when trained on site group A when evaluating on the test set within site group A versus the test set within site group B for predicting all AKI severities up to 48 hours ahead of time. Most of the key metrics were comparable across site groups. Note that the model would still need to be retrained to generalise outside of the VA population to a different demographic and different set of clinical pathways and hospital processes elsewhere.

Hyperparameter	Values considered
RNN cell type	LSTM, GRU, UGRNN, SRU, Intersection RNN, MANN, NTM, DNC, RMC
RNN cell size	100, 150, 200, 250, 300, 400, 500
RNN num. layers	1, 2, 3
Embedding num. layers	1, 2, 3
Embedding dim. per feature type	200, 250, 300, 400, 500
Embedding combination	concatenate, sum
Embedding architecture type	MLP, AE, VAE
Embedding reconstruction loss weight	1e-2, 1e-3, 1e-4
Embedding reconstruction sampling ratio	1, 2, 5, 10
Optimise directly for PR AUC	on, off
Highway connections	on, off
Residual embedding connections	on, off
Input dropout	0, 0.1, 0.2, 0.3
Output dropout	0, 0.1, 0.2, 0.3
Embedding dropout	0, 0.1, 0.2, 0.3
Variational dropout	0, 0.1, 0.2, 0.3
Input regularisation type	None, L1, L2
Input regularisation term weight	1e-3, 1e-4, 1e-5
BPTT Window	32, 64, 128, 256, 512
Embedding activation functions	Tanh, ReLU [61], Leaky ReLu [62], Swish [63], ELU [64], SELU [65], ELiSH [66], Hard ELiSH [66], Sigmoid, Hard Sigmoid
Auxiliary task loss weight	0., 0.1, 0.5, 1, 5, 10
Learning rate	1e-2, 1e-3, 1e-4, 1e-5
Learning rate decay scheduling	on, off
Learning rate decay num. steps	6000, 8000, 12000, 15000, 20000
Learning rate decay base	0.7, 0.8, 0.85, 0.9, 0.95
Batch size	32, 64, 128, 256, 512
NTM/DNC memory capacity	64, 128, 256
NTM/DNC memory word size	16, 32, 64
NTM/DNC memory num. reads	6, 10
NTM/DNC memory num. writes	1, 2, 3

Figure 10

Model architecture choices and hyperparameter combinations evaluated in the experiments for the entire duration of AKI risk model development.