Benchmarking Deep Learning Architectures for Predicting Readmission to the ICU and Describing Patients-at-Risk

Sukrit Ganesh
sukritg2@illinois.edu
UIUC Department of Computer Science

Azaan Barlas
abarlas2@illinois.edu
UIUC Department of Computer Science

ABSTRACT

Recent advancements in artificial intelligence hardware and software have enabled machine learning models to perform complex predictions. The field of healthcare, currently a multi-trillion dollar industry, has seen massive growth in the use of artificial intelligence, as models can save both money and lives by predicting patient outcomes, performing diagnoses, and even discovering new drugs. One of the most useful applications of deep learning in healthcare is predicting patient readmission, as doing so can allow for proactive measures taken by a healthcare facility to maximize the quality of treatment and minimize costs. Various techniques exist for predicting patient readmission, each with its unique benefits and drawbacks. We propose running benchmark tests on a wide variety of machine learning techniques to predict the ICU readmission likelihood for patients in order to determine the optimal techniques for various scenarios. We also propose tweaking the hyperparameters of these models in order to determine the effect of modifying said hyperparameters on the outcome of the model.

CCS CONCEPTS

• **Machine Learning in Healthcare**; • **Data Pre-Processing**; • **Recurrent Neural Networks**; • **Machine Learning Benchmarking**; • **Attention Neural Network**;

KEYWORDS

Machine Learning, Fairness, Bias

ACM Reference Format:
Sukrit Ganesh and Azaan Barlas. 2023. Benchmarking Deep Learning Architectures for Predicting Readmission to the ICU and Describing Patients-at-Risk. In *Proceedings of AI-Faire '22: Fake ACM Symposium on Applied Machine Learning (AI-Faire '22).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The paper we are working on is *Benchmarking deep learning architectures for predicting readmission to the ICU and describing patients-at-risk* [2]. The paper utilizes a variety of neural network architectures to predict the probability of patient readmission to an ICU using prior patient medical data. As described in the abstract, predicting readmission is one of the most useful tools hospitals can use to reduce costs, improve convenience for patients, and even save lives. Unnecessary admissions, or failure to readmit patients who have relapsed or fallen seriously ill, can strain hospital resources, result in extra resources for delayed treatment, or even cost lives. Furthermore, predicting patient readmission can allow doctors to craft tailored treatment plans for patients and allocate future hospital resources accordingly. It is important to examine the different ways to predict readmission and understand how changing the different parameters relating to prediction can affect the accuracy of the process.[3]

Several deep learning architectures that used methods such as attention mechanisms, recurrent layers, neural ordinary differential equations (ODEs), and medical concept embeddings with time-aware attention were used in this paper and trained using the MIMIC-III dataset associated with 45,298 ICU stays for 33,150 patients. The best-performing model for predicting readmissions to the ICU was an RNN-ODE hybrid model (Precision: 0.331, AUROC: 0.739, F1-Score: 0.372). In general, for problems that involve EMR data, using attention mechanisms or recurrent layers in isolation would always provide subpar results. Methods have been proposed to counteract this such as appending time-dependent data and using ordinary differential equations.

The utility of predicting patient readmissions into the ICU is an extremely vital task in the world of healthcare as 10% of critically ill patients are readmitted after being discharged, and around 30% of hospital costs go into readmissions to the ICU. Furthermore, around 1% of the entire gross national product of the United States is related to ICU admissions, largely owing to the very high cost of providing ICU treatment and the nature of the medical problems that necessitate an ICU visit. This paper aims to predict a patient's risk of readmission within 30 days of initial discharge from the ICU, while also evaluating the feasibility of using different machine-learning models involving attention and ODEs to gain a better understanding of intensive-care patients with an increased risk of readmission.

2 SCOPE OF REPRODUCABILITY

As it stands, the paper is easily reproducible. The codebase to process the MIMIC-III dataset and train and test the models are publicly available.[1] Although the MIMIC-III dataset requires completing a data handling and privacy course, it is available to anyone who successfully completes that course. The code repository, found on GitHub, contains the pre-trained models, as well as code to process the dataset, train the models, and evaluate them. Some additional processing and modification may be required, as the code is from 2019, but overall it is in relatively good condition. Reproducing the pipeline will require tweaking the code to match modern standards and running the entire pipeline (described in more detail in the next section) to generate training data, trained models, and inference

AI-Faire '22, Dec 10–Dec 20, 2022, Urbana, IL
2023. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

results.

Some challenges exist, however, in reproducing the code. First of all, deep learning models are very computationally intensive, and it may take an inordinately long time to run these models on an average CPU. We propose using a GPU in order to speed up training. The GPU can yield a performance speedup of several orders of magnitude when using CUDA (a library that enables mathematical computations to be performed on an NVidia Graphics Card). If CUDA turns out to be infeasible, then we can reduce the size of the training data as well. Although this may yield inferior results, the performance tradeoff should be negligible as adding more data has diminishing returns. Utilizing a cloud computing process such as Amazon AWS is preferred due to the adequate computing power provided, though such solutions may cost money. We may also try utilizing the tensor processing unit (TPU) on Jupyter Colab in order to speed up training. All of these options for speeding up training have tradeoffs and benefits, but they are all viable options not just for this paper but for machine learning in general. Inference should not be an issue, as models do not contain significant quantities of parameters, and inputs are relatively straightforward, consisting of medical records ranging several weeks. Secondly, the code may be outdated, utilizing old packages, and the training data may need to be massaged to fit the dataset. Once again, this challenge should be straightforward to overcome, as MIMIC-III is a commonly used dataset, and the machine learning models used for benchmarking use relatively generic mechanisms.

Potential ablations, described in detail in the next section, can also be implemented. These ablations will include additional models, as well as modifications on the training data, in order to gather more benchmarking data. We may contact the original authors of the code for assistance in running and augmenting the model, as we may encounter roadblocks along the way. The first of these potential ablations is to use additional machine learning models for benchmarking. The second potential ablation is tweaking the hyperparameters such as the learning rate and optimizer. The third potential ablation is modifying the training data to use different windows of historical medical data. For instance, one pass may only take ICU admission data from the last 30 days, while another pass will use the entire ICU admission history of the patient.

The code will be written entirely in Python. The MIMIC-III dataset will be downloaded from MIT PhysioNet, after completing the required data handling courses.

3 METHODOLOGY

3.1 Model Description

The original paper uses many different models, but mainly a combination of these main models:

(1) Recurrent Neural Networks - RNNs can use information from previous inputs in their computation, making them suitable for tasks where the output depends on the previous inputs. RNNs have a loop that allows information to persist, and the output of the loop is used as input to the next iteration.
(2) Ordinary Differential Equations (Neural Networks) - ODEs are used to model complex systems and dynamic processes. Instead of computing a fixed function, an ODE learns a differential equation that describes the behavior of a system. This allows for more accurate and flexible modeling of dynamic systems than traditional neural networks.
(3) Attention Mechanisms - Attention is used in neural networks to selectively focus on different parts of the input sequence, allowing the network to process information more efficiently. Attention is often used in sequence-to-sequence models, where the network needs to process variable-length inputs and outputs, such as machine translation or image captioning. Instead of processing the entire input sequence at once, the network uses attention to focus on the most relevant parts of the input at each step.
(4) Time Decay - Older samples in a sequence tend to have a smaller impact on a prediction. In the case of predicting ICU readmission, older ICU visits have less influence on readmission probabilities than more recent visits. Applying time decay mechanisms (can use ODEs or simple mathematical functions) can give more influence to more recent events in an input sequence, as internal memory states decay over time.
(5) Medical Concept Embedding - MCE is a technique used to embed complex medical codes to be fed into an RNN or other machine learning model. They are based on the continuous bag-of-words model.
(6) Logistic Regression - LR is a binary classification method. It can be modeled as a function that can take in any number of inputs and constrain the output to be between 0 and 1. We can consider logistic regression to be a one-layer neural network. We can use a binary classification method because our dependent variable is just a readmission score derived from a yes/no result from our models, on the likelihood of readmission.

Below is a list of the specific model configurations used for benchmarking:

- Vanilla Recurrent Neural Network (RNN)
- RNN with ODE
- RNN with ODE time decay
- RNN with ODE time decay and Attention
- RNN with ODE and Attention
- RNN with exponential time decay
- RNN with exponential time decay and Attention
- RNN with concatenated time differences between observations
- RNN with concatenated time differences between observations and Attention
- Neural ODE with Attention
- Attention with concatenated time differences between observations (no RNN)
- RNN with Medical Concept Embeddings
- RNN with Medical Concept Embeddings and Attention

- Medical Concept Embeddings and Attention (no RNN)

Our goal was to retrain the top models in the original paper as well as our own Vanilla RNN model and an RNN Logistic Regression model. These models were entirely trained using the MIMIC-III dataset and the existing infrastructure provided by the original authors of the paper. We hypothesized that we would receive close to or better results than the original paper, yet we did not. We hypothesized that the custom models will yield similar performance to the best-benchmarked models. This ties into the law of diminishing returns, whereby a basic neural network yields a relatively high base accuracy, and additional features yield marginal but significant improvements. We also hypothesized that with sufficient training and parameters, even a fully-connected layer can yield higher performance than any of the more complicated models, although this will require very long training times. Although more complex models are more efficient, brute-force solutions that involve large models with many parameters will win out. We believed there are diminishing marginal gains in accuracy as models become more complex and elaborate, or are trained for longer periods of time; they may even overfit the training data. The results of our experiments showed differently as seen in the Results section of our paper.

3.2 Data Descriptions

The MIMIC-III dataset contains longitudinal electronic medical records for numerous sample patients. Two distinct inputs will be fed into the model: static and timestamped codes. Static data refers to variables describing a specific patient, such as age, sex, insurance type, marital status, etc. Timestamped codes, on the other hand, refer to medical data such as diagnoses, prescriptions, and procedures. These medical entities have unique codes. The timestamped codes provide information about a patient's medical history and will be used to predict readmission risk to the ICU.

The following features are contained within the MIMIC-III dataset:

(1) Clinical Notes
(2) Demographic Information
(3) Admission and discharge information
(4) ICD-9 codes
(5) Lab Results
(6) Medications
(7) Vital Signs
(8) Procedures Performed
(9) Imaging data

The main features that are used in our model after data preprocessing are shown as below:

(1) Patient data: This is a group of columns, which are demographic data such as age, sex, and other things like this.
(2) Readmission rate: This column contains the dependent variable that we are trying to determine.
(3) Diagnosis and procedures: This column stores information about the diagnoses and procedures a patient goes through. This data is stored in the form of timestamped codes, where the individual diagnoses and procedures for a patient are associated with a timestamp.
(4) Charts and prescriptions: This stores information about prescriptions and charts related to a patient's prescriptions. This data is also stored in timestamped code format.
(5) Times: This column is related to the times that a patient is diagnosed, and prescribed, as well as when a procedure is performed on the patient.
(6) IDs: These are IDs for the training, validation, and testing data for each patient's event IDs.

This project will only use ICU admission data from the MIMIC-III dataset; any timestamped codes which take place outside an ICU visit will not be used to compute readmission risk. The dataset maintains patient anonymity and abides by all medical privacy laws.

3.3 Implementation

For our research question, we followed a specific approach:

(1) Data Preprocessing: We will use the existing preprocessing infrastructure in order to process the MIMIC-III dataset and generate the required compressed, processed data for input to the models. This step will load the compressed data and save it in array format for use by later stages of the pipeline.
(2) Model Selection, Parameter Modifications, and Additional Data Inclusion: We will compare the performance of several deep learning models, including recurrent neural networks (RNNs), transformers, and graph neural networks (GNNs), and other regression models against the models used in the original paper. We will also run the model on data containing different lengths of observation periods. We will include both the original models as well as our custom models.
(3) Model Training: We will train the selected models on the preprocessed EMR data. We will use a training/validation/test split to evaluate the performance of the models. This step will remain largely unchanged, except we will use our additional datasets and models as part of the benchmarking. We will also train the model for varying epochs in order to judge the relationship between training epochs and model performance.
(4) Model Evaluation: We will evaluate the performance of the selected models and compare them to the models used in the original paper. We will use metrics such as Precision, AUROC, and F1-Score. Variables modified will include the number of epochs, the length of the observation period, and the specific model used.
(5) Results Interpretation: We will compare our methods, results, and idea with the findings in the original paper and generate useful visualizations. If our strategies appear to be competitive with those used in the paper, we will discuss the possible implications of utilizing our methods.

For our ablations, we aimed to implement three distinct models: a vanilla RNN network, an RNN network with logistic regression,

and a Bayesian RNN. The first model is a simple, compact model that we expect to be quick to train but will yield the worst results. We also predict that RNN with logistic regression will yield marginally better performance owing to the additional layer, while the Bayesian RNN will yield the best performance.

For the loss functions, we used "BCEWithLogitsLoss" from Pytorch, which is a standard binary classification function loss. This is a cross-entropy loss that gets the probability of the predicted value, by taking a sigmoid of the predicted values. We also used the positive weight argument in order to get the proportion of positively predicted labels.

The optimizer we used was a standard Adam algorithm, which is a gradient descent optimizer based on a learning rate given. It gets the derivative of the loss in every iteration and updates the model's parameters to minimize the loss.

Overall, our specific approach will allow us to compare the performance of additional deep learning models and datasets against the models and singular MIMIC-III dataset used in the original paper and determine their suitability for predicting ICU readmissions. We will employ significant automation to run the entire pipeline and generate visualizations.

3.4 Computational Requirements

Initially, we faced a significant challenge in obtaining suitable computational resources. We had to wait for a strong AWS GPU to get approved, which took a long time. After approval, we used the g4dn.8xlarge GPU instance on AWS Sagemaker to speed up the computations. The GPU instance provided the necessary computational power to run our models efficiently, allowing us to train our models in a timely manner.

The training process of our models was computationally intensive. To minimize the computational burden, we opted to use fewer samples for our testing, however, training still took around 3-4 hours for each model and we did not have time to train our Bayesian model to an accuracy of our liking.

Another significant computational challenge was the processing of the MIMIC-III dataset. The dataset is large, weighing around 49 GB, and processing all the CSV files took a considerable amount of time. Our computation time was further increased by the need to clean and pre-process the data before training. Overall, the computational requirements for this study were significant, and we relied heavily on cloud computing resources to enable us to carry out our research efficiently.

4 RESULTS

The results we obtained show the performance of different neural network models on a task, where the evaluation metrics include Average Precision, AUROC, F1 score, PPV (true positive rate), NPV (false positive rate), sensitivity, and specificity. The

	Average Precision	AUROC	F_1-Score	Sensitivity	Specificity
ODE + RNN + Attention	0.314 [0.306,0.321]	0.739 [0.736,0.741]	0.376 [0.371,0.381]	0.685 [0.666,0.704]	0.677 [0.658,0.696]
ODE + RNN	0.331 [0.323,0.339]	0.739 [0.737,0.742]	0.372 [0.367,0.377]	0.672 [0.659,0.686]	0.697 [0.683,0.711]
RNN (ODE time decay) + Attention	0.316 [0.307,0.324]	0.743 [0.741,0.746]	0.375 [0.370,0.379]	0.648 [0.641,0.656]	0.733 [0.726,0.739]
RNN (ODE time decay)	0.300 [0.293,0.308]	0.741 [0.738,0.744]	0.372 [0.367,0.376]	0.710 [0.698,0.722]	0.667 [0.655,0.679]
RNN (exp time decay) + Attention	0.320 [0.312,0.328]	0.748 [0.745,0.751]	0.377 [0.372,0.382]	0.704 [0.692,0.715]	0.680 [0.668,0.692]
RNN (exp time decay)	0.304 [0.297,0.311]	0.735 [0.732,0.738]	0.368 [0.363,0.373]	0.707 [0.700,0.714]	0.670 [0.663,0.676]
RNN (concatenated Δtime) + Attention	0.312 [0.303,0.320]	0.741 [0.739,0.744]	0.368 [0.363,0.372]	0.687 [0.680,0.695]	0.688 [0.681,0.696]
RNN (concatenated Δtime)	0.311 [0.303,0.320]	0.739 [0.737,0.742]	0.364 [0.359,0.369]	0.698 [0.692,0.704]	0.688 [0.684,0.693]
ODE + Attention	0.294 [0.285,0.302]	0.717 [0.714,0.720]	0.333 [0.328,0.339]	0.776 [0.768,0.784]	0.554 [0.548,0.560]
Attention (concatenated time)	0.286 [0.277,0.295]	0.711 [0.709,0.714]	0.330 [0.325,0.334]	0.700 [0.686,0.714]	0.614 [0.601,0.628]
MCE + RNN + Attention	0.317 [0.308,0.325]	0.736 [0.734,0.739]	0.373 [0.369,0.378]	0.630 [0.622,0.638]	0.744 [0.738,0.749]
MCE + RNN	0.298 [0.291,0.306]	0.727 [0.724,0.730]	0.361 [0.357,0.366]	0.654 [0.645,0.663]	0.706 [0.697,0.715]
MCE + Attention	0.269 [0.261,0.278]	0.689 [0.686,0.692]	0.312 [0.308,0.316]	0.686 [0.676,0.695]	0.616 [0.607,0.625]
Logistic Regression	0.257 [0.248,0.266]	0.659 [0.656,0.663]	0.296 [0.291,0.300]	0.606 [0.597,0.615]	0.647 [0.639,0.655]

Figure 1: The original results from the paper.

models are variants of bi-RNN (bidirectional recurrent neural networks), Vanilla_RNNs (simple recurrent neural networks), and RNN_logistic_regression, with some including attention mechanisms and ODE (ordinary differential equation) solvers. The yellow rows in the table refer to the two extra models we ran - a vanilla RNN and an RNN with logistic regression.

The precision and AUROC scores for the existing models were very similar between the paper's statistics and our replication study. For instance, the RNN + ODE + Attention model had precision and AUROC of 0.314 and 0.739, respectively, in the original paper. When we tried training and evaluating the same model, we got precision and AUROC of 0.316 and 0.737, respectively. Other rows and columns have similar statistics.

As hypothesized, the extra models we ran had slightly worse results owing to their reduced complexity, albeit not by that much. The vanilla RNN, for instance, had a precision of 0.253 and an AUROC of 0.648. These figures are roughly 20% worse than the best models. These models took significantly less time for training, however.

We also attempted to run a Bayesian RNN, but unfortunately, we were unable to get it functional before the deadline.

Comparing the results, we see that the models with attention generally perform better than those without. The ODE-based models perform similarly to the bi-RNN models, but with higher time complexity. The best model is the ode_bi-RNN with attention, with a higher average precision and AUROC than the other models. On the other hand, the Vanilla_RNN and the RNN_logistic_regression models performed worse than the others, indicating the importance of more sophisticated architectures for this task.

In terms of evaluation metrics, PPV is high for many models, indicating a low false positive rate. However, some models have a sensitivity score below 0.7, suggesting that they are not able to detect a significant portion of the positive cases.

Model	Average Precision	AUROC	F1 Score	PPV	NPV	Sensitivity	Specificity
Vanilla RNN	0.253 [0.239,0.268]	0.648 [0.643,0.654]	0.299 [0.291,0.307]	0.982 [0.944,1.019]	0.883 [0.881,0.885]	0.527 [0.516,0.538]	0.716 [0.709,0.722]
RNN + Logistic Regression	0.257 [0.243,0.272]	0.658 [0.652,0.664]	0.305 [0.298,0.312]	0.982 [0.946,1.018]	0.883 [0.881,0.885]	0.524 [0.501,0.547]	0.73 [0.709,0.75]
RNN + Time Decay	0.304 [0.274,0.335]	0.73 [0.718,0.743]	0.368 [0.351,0.386]	0.975 [0.918,1.032]	0.88 [0.875,0.885]	0.698 [0.661,0.734]	0.672 [0.638,0.705]
RNN + ODE + Attention	0.316 [0.279,0.353]	0.737 [0.725,0.749]	0.378 [0.354,0.401]	1.0 [nan,nan]	0.881 [0.876,0.886]	0.707 [0.636,0.777]	0.652 [0.579,0.725]
RNN + ODE	0.335 [0.298,0.373]	0.737 [0.724,0.75]	0.374 [0.351,0.397]	1.0 [nan,nan]	0.882 [0.878,0.887]	0.647 [0.59,0.705]	0.722 [0.662,0.782]
RNN + ODE + Attention + Time Decay	0.32 [0.285,0.356]	0.74 [0.728,0.752]	0.375 [0.355,0.394]	0.96 [0.87,1.05]	0.881 [0.876,0.886]	0.63 [0.606,0.653]	0.746 [0.728,0.764]
RNN + Attention + Time Decay	0.324 [0.288,0.36]	0.745 [0.733,0.758]	0.378 [0.358,0.399]	1.0 [nan,nan]	0.882 [0.877,0.887]	0.701 [0.657,0.745]	0.675 [0.626,0.725]

Figure 2: The results we got from running our experiment. The yellow rows refer to our own neural networks; the other rows are the existing paper's neural networks that we re-trained.

5 DISCUSSION

The results of our experiments provide insights into the effectiveness of different neural network architectures for predicting mortality risk in ICU patients using the MIMIC-III dataset. Overall, our models demonstrated moderate to good performance on several evaluation metrics, including AUROC, F1, and sensitivity. However, there were some notable differences in performance between our models and those reported in previous studies.

One key finding was that our best-performing model, the ode-bi-RNN architecture, achieved an AUROC of 0.737, which is similar to or slightly better than the performance reported by other studies that used similar datasets and model architectures. This suggests that the "ode-birnn" is a promising approach for patient readmission risk prediction in ICU patients. Our results suggest that more complex models, such as the "ode-bi-RNN" architecture, may be necessary for accurate predictions. Overall, these results provide insights into the effectiveness of different neural network models for this task and highlight the importance of attention mechanisms and complex architectures for achieving good performance.

The most difficult part of the implementation was actually setting up the environment to run the code, as minor tweaks were necessary. Once everything was set up, however, we were easily able to train the existing models and run our own models as well. The authors did an excellent job modularizing and documenting the code.

Another thing to consider is the tradeoff between runtime and accuracy. Our RNN models took significantly less time for training and inference than the more complex models which utilized advanced techniques such as attention or ODEs. However, the reduction in AUROC and precision was not as significant, roughly 20%. This alludes to the law of diminishing returns, whereby the marginal increase in performance metrics reduces as the model becomes more and more complex. Granted, most healthcare companies will have access to supercomputers capable of processing vast quantities of data, but the tradeoff between speed and performance should be considered.

Future research should explore more complicated models and perhaps data preprocessing in order to gauge their impact on the performance metrics. Clearly, more complicated techniques are necessary to yield optimal results, and advanced architectures such as Graph Neural Networks (GNNs) should be considered. The models can be run with truncated input datasets (ex. a smaller window of past medical data). Medical data is inherently complex, and trying out different combinations of data preprocessing techniques, hyperparameters, and model architectures is essential to understanding how these black box models function.

6 CONCLUSION

Predicting patient readmission is one of the most important things a hospital can do to save money, time, and lives. Benchmarking a variety of datasets and models for patient readmission will help put the process into perspective and provide insight to healthcare professionals and researchers who may want to come up with efficient and effective algorithms to achieve the task, especially when working with limited computing power or poor datasets. Our project showed that predicting patient readmission with relatively high accuracy within 30 days is a viable task with consumer electronics and that even basic models yield satisfactory performance. Further exploration into the subject should use more complicated models, larger training datasets, and more powerful computers. Predicting patient readmission has the potential to save billions of dollars and, more importantly, save countless lives, and our project demonstrated that it is viable to use patients' medical records to this end. Combining healthcare with deep learning can potentially revolutionize the industry, and future research should focus extensively on predicting patient outcomes in order to save money and lives.

7 LINKS

We recommend downloading the video as the Google Drive player can be buggy. The github repository also contains the video.

- Github code: Click here.
- Presentation Slides: Click here.
- Presentation Video: Click here.

REFERENCES

[1] Roger Mark Alistair Johnson, Tom Pollard. Mimic-iii clinical database. 2016. URL https://physionet.org/content/mimiciii-demo/1.4/.
[2] Sebastiano Barbieri, James Kemp, Oscar Perez-Concha, Sradha Kotwal, Martin Gallagher, Angus Ritchie, and Louisa Jorm. Benchmarking deep learning architectures for predicting readmission to the icu and describing patients-at-risk. *Scientific reports*, 10(1):1111, 2020.
[3] Yaron Blinder. Predicting 30-day icu readmissions from the mimic-iii database. 2017.