Enhancing Diagnostic Classification in Electronic Health Records Using Graph-Guided Neural Networks A Study on the MIMIC-IV Dataset

Joel Fehr
CAI Centre for Artificial Intelligence
ZHAW Zurich University of Applied Sciences
Zurich, Switzerland
joel.fehr@bluewin.ch

***Abstract*—This work presents an implementation of the RAINDROP model to perform classification on 25 clinical care conditions of the MIMIC-IV dataset. The model uses electronic health record data in order to perform the classification task. Two models were trained and compared. The first was trained on a dataset containing multiple Intensive Care Unit (ICU) stays per patient and the second one on data only containing one ICU stay per patient. The results showed that the model was not able to match benchmark scores achieved by previous works, indicating that further research is necessary to proof the concept of the approach.**

***Index Terms*—Multivariate Irregular Time Series, Electronic Health Records, RAINDROP, MIMIC-IV, Phenotype Classification**

I. Introduction

Electronic health record (EHR) systems are widely adopted in hospitals across the United States and help to collect and store data during clinical routine practice [1]. These EHR systems help a lot in the collection of digital data and rise opportunities for machine learning researchers to solve various problems in health care [2]. Such collected EHR data is often delivered as multivariate time series data because the data is collected over the course of time and most of the time multiple variables are involved since multiple processes are monitored and administrated during a hospital stay. A problem that is often associated with multivariate time series are irregularities that are caused by missing observations [3]. Other issues with irregular times series are that samples may vary in their number of observations, the time between observations is not uniformly distributed across the data set and also not across each sample and not every observation may contain values for every feature [4]. The authors of RAINDROP [3] discuss the problem of modern techniques like RNNs [5], LSTMs [6], GRUs [7] and transformers [8] because they are restricted to regular sampling or that they assume aligned measurements across different modalities. The RAINDROP model tries to leverage recent advances in graph neural network in order to take advantage of relational structures among sensors. The authors justify this approach because a two-stage approach including the imputation of missing values to obtain regular time series data and a following optimization of a downstream model does not seem to be optimal [9],[10]. The goal is to circumvent the imputation stage and directly apply a model on irregular times series data. RAINDROP builds on the idea of learning latent graphs from multivariate irregular times series and to use neural message passing to model time-varying inter-sensor dependencies.

This work explores the challenges that are associated with multivariate irregular time series data. The data particularly comes from EHR data of the MIMIC-IV [1] dataset. The work implements the RAINDROP model in order to perform phenotype classification. For the classification task 25 clinical care conditions are used and 15 variables were extracted from the dataset using a pre-processing pipeline which was built by Hayat et al. [11].

The main experiment was the comparison of a model trained on all samples with one that was only trained on one sample coming from a distinct patient. The partial model was trained on data including at most one ICU stay per patient. The results showed that the full model outperformed the partial model by a small margin. However, overall, the achieved results did not match benchmark performance reported in previous works.

In conclusion, the work provides insights into the challenges of handling multivariate irregular time series data in healthcare settings and evaluated the RAINDROP model's performance on the MIMIC-IV EHR dataset. In order to achieve state-of-the-art performance and make a deployment into a practical setting possible further improvements are necessary.

II. Related Work

This work implements the RAINDROP model in order to overcome the challenges associated with multivariate irregular time series. RAINDROP leverages a graph neural network and neural message passing to model time-varying dependencies between sensors.

Health care datasets that are publicly available are important to advance research in the medical field. MIMIC-IV is a freely available dataset that is sourced from EHR data from real hospitals. In this work MIMIC-IV is used as clinical time series dataset. For the predecessor of MIMIC-IV [12] many papers exist that presented benchmark results or pre-processing pipelines for the dataset [2], [13]. Harutyunyan et al. [2] present benchmarks for four clinical prediction tasks, including

modeling risk of mortality, forecasting length of stay, detecting physiologic decline and phenotype classification. The work is built on MIMIC-III. Purushotham et al. [13] published another work that is built on MIMIC-III and present benchmark results on mortality prediction, length of stay prediction and ICD code group prediction. The authors propose deep learning and ensemble machine learning models. The newest version of MIMIC is less extensively studied but a few papers are available [11], [14], [15]. Mandyam et al. [14] present COP-E-CAT, an open-source processing and analysis software for MIMIC-IV. The proposed software [1] enables users to build datasets that can be further used for downstream tasks. Gupta et al. [15] present a pipeline to extract, clean and process the data of MIMIC-IV. The pipeline [2] covers readmission, length of stay, mortality and phenotype prediction tasks.

This work uses the data pre-processing pipeline that was built by the researchers that published MedFuse [11]. The pipeline [3] is specifically adapted for MIMIC-IV and is based on the pre-processing pipeline that was developed by Harutyunyan et al. [2].

III. Methodology

A. RAINDROP

RAINDROP takes samples as input and each sample contains multiple sensors which each consists of irregularly recorded observations. In the context of the dataset MIMIC-IV which is used in this work, samples correspond to patients and sensors correspond to the different variables that were recorded during the hospital stay at different time steps. The notation of RAINDROP lets $\mathcal{D} = \{(S_i, y_i)|i = 1, \ldots, N\}$ denote an irregular time series dataset with N samples. Each sample S_i is associated with a label $y_i \in \{1, \ldots, C\}$ which indicates which of the classes C are associated with the sample S_i. Each of the samples contains M sensors that are not uniformly measured. These sensors are denoted as u, v etc. Each sensor consists out of a sequence of time ordered observations. A sensor u of a sample S_i has a single observation denoted as a tuple $(t, x^t_{i,u})$, meaning that sensor u was recorded with a value $x^t_{i,u} \in \mathbb{R}$ at a timestamp $t \in \mathbb{R}^+$. Since the time series is irregularly sampled, time intervals between successive observations can vary across sensors. $\mathcal{T}_{i,u}$ denotes the set of timestamps of sensor u. Given a dataset D RAINDROP aims to learn a function $f : S_i \rightarrow z_i$ that is able to map S_i to a fixed-length representation z_i that is suitable for downstream tasks such as classification.

The aim of the RAINDROP model is to learn an embedding z_i with fixed dimensions for a sample S_i and to predict its associated labels $\hat{y}_i$. The model leverages a hierarchical architecture which is composed of three levels in order to generate a sample embedding. The three levels are based on the modelling of observations, sensors and whole samples.

In a first step RAINDROP constructs a graph for every sample where nodes correspond to sensors and edges indicate their relationships. $\mathcal{G}_i$ denotes a sensor graph for sample S_i and $e_{i,uv}$ is used for the associated edge weights of a directed graph from sensor u to sensor v. In the beginning all graphs are initialized as fully-connected graphs.

In a second step RAINDROP generates embeddings of individual observations. Let u be an active sensor with a recorded value at timestamp t and let v be an inactive sensor. An observation embedding denoted as $h^t_{i,u}$ is based on an observed value $x^t_{i,t}$ at a timestamp t and passes messages to neighboring sensors in order to generate observation embedding $h^t_{i,v}$. The embedding of the active sensor u is generated using a nonlinear transformation $h^t_{i,u} = \sigma(x^t_{i,u} R_u)$ where R_u is a trainable weight vector which is shared across samples. RAINDROP uses information of active sensors at timestamp t to estimate observation embeddings for non-active sensors that are neighbors of the active sensor u in the sensor dependency graph $\mathcal{G}_i$. For an edge between sensor u and sensor v with an edge weight $e_{i,uv}$ the inter-sensor attention weight $\alpha^t_{i,uv} \in [0, 1]$ is calculated first. It represents how important the sensor u is to the sensor v. The observation embedding of the inactive sensor v is then calculated by multiplication of the observation embedding $h^t_{i,u}$, two trainable weight matrices w_u and w_v, the inter-sensor attention weight $\alpha^t_{i,uv}$ and the edge weight $e_{i,uv}$. RAINDROP automatically updates the edge weights and prunes less important edges. The edge weights are updated based on the inter-sensor attention weights.

In a next step RAINDROP aggregates observation embeddings into sensor embeddings. Since observation embeddings at different timestamps have unequal importance to the sensor embedding, temporal attention weights are used. These temporal attention weights represent the importance of an observation embedding at timestamp t. A slight adaption from standard self-attention is used in order to calculate the temporal attention weights. A sensor embedding for sensor v in generated by the following steps: A concatenation of observation embedding $h^t_{i,v}$ with a time representation p^t_i. Then the concatenated embeddings for all $t \in \mathcal{T}_{i,v}$ are stacked into a matrix $H_{i,v}$. Then $\beta_{i,v}$ is calculated by

$$\beta_{i,v} = \text{softmax}(\frac{Q_{i,v} K^T_{i,v}}{\sqrt{d_k}} s)$$

, where $Q_{i,v}$ and $K_{i,v}$ are intermediate matrices that are derived from the stacked observation embeddings $H_{i,v}$. s is a trainable weight that results from the adaptation of the standard self-attention mechanism. Based on the calculated temporal attention weights the sensor embedding $z_{i,v}$ calculated by

$$z_{i,v} = \sum_{t \in \mathcal{T}_{i,v}} (\beta^t_{i,v} [h^t_{i,v} || p^t_i] W)$$

, where matrix W is a linear projector shared by all sensor and samples. $||$ denotes the concatenation of observation embedding with the time representation. All the attention weights such as $\alpha^t_{i,uv}$ and $\beta_{i,v}$ can be multi-head.

In a last step RAINDROP generates sample embeddings for a sample S_i. In order to generate sample embeddings, the sensor embeddings are aggregated across all sensors to obtain

[1] https://github.com/aishwarya-rm/cop-e-cat
[2] https://github.com/healthylaife/MIMIC-IV-Data-Pipeline
[3] https://github.com/nyuad-cai/MedFuse/tree/6f827589afd89562813cc5aa915762d054c29efc/mimic4extract

an embedding z_i. The aggregation is performed with a readout function g with $z_i = g(z_{i,v}|v = 1, 2, \ldots, M)$. The readout function g can be concatenation or averaging aggregation. The sample embedding z_i that is obtained from sample S_i can be used for further downstream tasks such as classification.

RAINDROP projects constant attributes that do not change over time (e.g. demographic information) to a vector a_i with a fully-connected layer and concatenates it with the sample embedding z_i. For a classification task, the concatenated vector $[z_i||a_i]$ is fed into a classifier that maps to the number of available classes $y_i \in \{1, \ldots, C\}$.

It is important to note that the experiments in this work are based on a multi-label setting, so a sample S_i can be associated with more than one label of $y_i \in \{1, \ldots, C\}$. For all the experiments the model was trained using a Binary Cross Entropy loss function combined with a sigmoid activation function at the final classification layer.

B. MIMIC-IV

MIMIC-IV is a freely accessible EHR dataset. The providers of the dataset extracted data of patients from hospital databases and created a master patient list. This list contained numbers that corresponded to all patients that were admitted to the ICU or the emergency department. The time of the admissions range from 2008 until 2019. The providers of MIMIC-IV denormalized tables, removed audit trails and built fewer tables in order to facilitate retroperspective data analysis. The providers did not perform data cleaning steps to make sure that the whole dataset is a good representation of a real-world dataset. For deidentifiying the dataset, the providers of MIMIC-IV removed patient identifiers and replaced them with random ciphers. Deidentification also included random shifting of date and time and it is important to note that temporal comparisons between different patients is not possible. The MIMIC-IV dataset follows a modular structure and consists of the *hosp* and *icu* module. The *hosp* module contains the data that was derived from the hospital wide EHR. The *icu* module on the other hand contains data that was sourced from clinical information systems. The version of MIMIC-IV that is used for this work is version 1.0. The work initially attempted to use the newest version 2.2 but it order to replicate the the pre-processing of Medfuse, version 1.0 was used. In this work we use the same terminology as Harutyunyan et al. [2]. *Patients* are called *subjects* and with each patient one or more *hospital admissions* are associated. Further can a patient have one or more *ICU stays* per admission and these are called *episodes*. Single measurements, observations or treatments are called *events*.

IV. Experiments

A. Phenotype classification

The conducted experiments in this work are based on a phenotype classification of 25 clinical care conditions. The 25 labels and their prevalence in the whole dataset after pre-processing can be found in Table I. The 25 labels are multi-label, therefore a patient in a given ICU stay can be associated with more than one label. The dataset in this work $\mathcal{D} = \{(S_i, y_i)|i = 1, \ldots, N\}$ consists of a sample S_i which contains the extracted EHR data for a patient and a given ICU admission and y_i corresponds to its associated labels. This work trains RAINDROP to predict labels y_i given a specific sample S_i.

TABLE I: Prevalence of diseases

Disease	Prevalence
Acute and unspecified renal failure	0.268
Acute cerebrovascular disease	0.055
Acute myocardial infarction	0.075
Cardiac dysrhythmias	0.325
Chronic kidney disease	0.207
Chronic obstructive pulmonary disease and bronchiectasis	0.143
Complications of surgical procedures or medical care	0.188
Conduction disorders	0.101
Congestive heart failure; nonhypertensive	0.254
Coronary atherosclerosis and other heart disease	0.313
Diabetes mellitus with complications	0.114
Diabetes mellitus without complication	0.172
Disorders of lipid metabolism	0.46
Essential hypertension	0.418
Fluid and electrolyte disorders	0.372
Gastrointestinal hemorrhage	0.070
Hypertension with complications and secondary hypertension	0.216
Other liver diseases	0.125
Other lower respiratory disease	0.095
Other upper respiratory disease	0.049
Pleurisy; pneumothorax; pulmonary collapse	0.067
Pneumonia (except that caused by tuberculosis or sexually transmitted disease)	0.126
Respiratory failure; insufficiency; arrest (adult)	0.160
Septicemia (except in labor)	0.157
Shock	0.122

B. Pre-processing of MIMIC-IV dataset

MIMIC-IV contains International Classification of Diseases (ICD) codes of version 9 and 10. The pre-processing pipeline of Hayat et al. [11] maps all ICD-10 to ICD-9 codes according to the guidelines of Centers for Medicare & Medicaid Services [4]. The pre-processing pipeline then maps the ICD-9 codes to Clinical Classifications Software (CSS) categories. This work attempted to use the same 17 clincial variables as Hayat et al. [11] and Harutyunyan et al. [2]. An overview about these 17 variables can be found in Table II and in Table III. Figure 2 and 3 show visualizations of the selected variables. The variables `Capillary refill rate` and `Glascow coma scale total` only contained `NaN` values after further inspection. The authors of Medfuse were contacted in order to obtain further information but without success. Therefore this work only makes use of the other remaining 15 variables. After the pre-processing and one-hot encoding of the categorical variables a total of 29 variables is

[4]Centers for Medicare & Medicaid Services, https://www.cms.gov/Medicare/Coding/ICD10/2018-ICD-10-CM-and-GEMs

obtained. As static variables which do not change over time `Gender`, `Ethnicity` and `Age` were used.

TABLE II: Summary Statistics continuous variables

Variable	Mean	STD	Range
Diastolic blood pressure	63.015	15.280	0.000-348.000
Fraction inspired oxygen	0.496	0.175	0.210-1.000
Glucose	148.726	66.890	33.000-1967.000
Heart Rate	85.352	18.271	0.000-295.000
Height	169.933	12.295	63.000-198.000
Mean blood pressure	78.814	16.383	14.000-330.000
Oxygen saturation	96.466	4.312	0.000-100.000
Respiratory rate	19.792	6.043	0.000-300.000
Systolic blood pressure	120.090	22.554	0.000-352.000
Temperature	36.939	0.690	26.100-43.056
Weight	85.138	24.553	0.000-249.793
pH	7.361	0.143	6.300-8.350

TABLE III: Summary Statistics categorical variables

Variable	Categories	Counts
Capillary refill rate	NaN	7'064'075
Glascow coma scale eye opening	None	124'605
	Spontaneously	694'624
	To Pain	57'292
	To Speech	201'250
	NaN	5'986'304
Glascow coma scale motor response	Abnormal Flexion	11'106
	Abnormal extension	5'518
	Flex-withdraws	66'964
	Localizes Pain	118'416
	No response	72'174
	Obeys Commands	798'946
	NaN	5'990'951
Glascow coma scale total	NaN	7'064'075
Glascow coma scale verbal response	Confused	125'651
	Inappropriate Words	11'378
	Incomprehensible sounds	24'122
	No Response	40'903
	No Response-ETT	335'284
	Oriented	538'433
	NaN	5'988'304

The pre-processing of the data was handled with several scripts which Hayat et al. [11] adapted for MIMIC-IV from the original scripts for MIMIC-III from Harutyunyan et al. [2]. A graphical overview of the following pre-processing pipeline and training procedure of RAINDROP can be found in Figure 1 which is based on a similiar figure created by Harutyunyan et al. [2]. Before pre-processing the critical care database contained $53'150$ patients, $69'211$ hospital admissions and $76'540$ ICU stays. In a first step relevant data is extracted from the raw tables using the script `extract_subjects_iv.py`. The script also excludes admissions that have multiple associated ICU stays or ICU transfers between different units. The resulting dataset after this first step consists of $47'046$ unique patients, $59'372$ hospital admissions and ICU stays. Further the resulting dataset contains $294'769'993$ events. In a next step the script `validate_events.py` script is applied. The script removes all events without an admission id (`hadm_id`) and events with and admissionn id that is not present in the table `stays.csv` which links ICU stay properties. The script also filters all events with an invalid ICU stay id (`stay_id`) but makes and attempt to recover these events based on the admission id. The next script in the pipeline `extract_episodes_from_subjects.py` generates a time series file for each of the remaining ICU stay episodes. The script makes sure that only the selected variables from Tables II and III are included. The script extracts the selected variables across multiple raw tables of the original dataset and values are cleaned and converted to a unified scale. In a next step the `split_train_and_test.py` is applied and partitions the data into a training- (80%) and a test-set (20%). The `create_phenotyping.py` script processes the data in order to build a dataset that is specific to the task. Further, the training-set is split into a training- (90%) and validation-set (10%). In order to arrive at our final dataset for this work the `length_dist.py` script gets the distribution of the length of the time series files. In a next step only files with a maximum length of 600 ($97.5th$ *percentile* of distribution) are generated. In a final step the `sanity_check.py` script scans across all the time series files and makes sure that improper files are removed. The green marked final dataset in Figure 1 is used for training and evaluating the RAINDROP model. The `get_info.py` script calculates the mean and standard deviations for the continuous variables in order to perform standardization on the variables when they are fed into the model. The script also removes outliers according to a list created by Harutyunyan et al. [2] which should provide clinically reasonable value ranges for the selected variables.

C. Experimental setup

The main experiment in this work is to compare the performance of a model that was trained on all samples with a model that was only trained on unique samples. For the second model the data was reduce to a dataset that only contained at maximum one ICU stay per unique patient. The particular ICU stay for a patient was drawn randomly if multiple existed. The dataset consists of $33'888$ samples for the training set, $3'729$ for the validation set and 9402 for the test set after the reduction. Both of the models were trained for 50 epochs with the parameters for RAINDROP that are visible in Figure 1 and the used optimizer was the ADAM [16] optimizer and the learning rate was set to 0.0001. As a loss function the built-in `pytorch` loss function `BCEWithLogitsLoss` was used. After training both of the models were evaluated on a test set containing mulitple ICU stays per patient and a test only containing one ICU stay per patient. Due to the high class imbalance that can be observed in Table I, for both models an attempt was made to oversample the minority classes in training batches in order to achieve balanced batches during the training. The motivation of this experiment was to

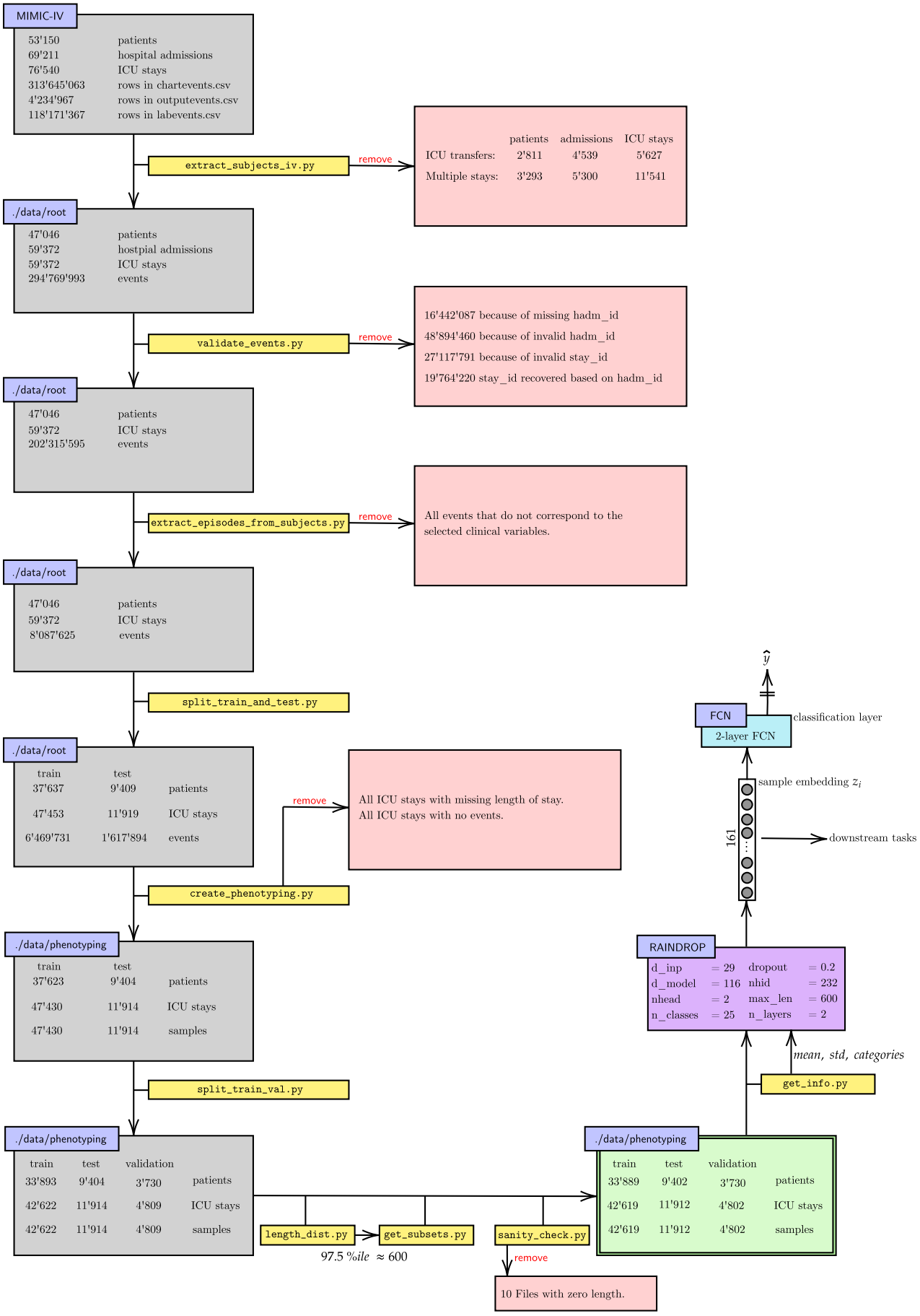


Fig. 1: Visualization of the pre-processing pipeline and model training

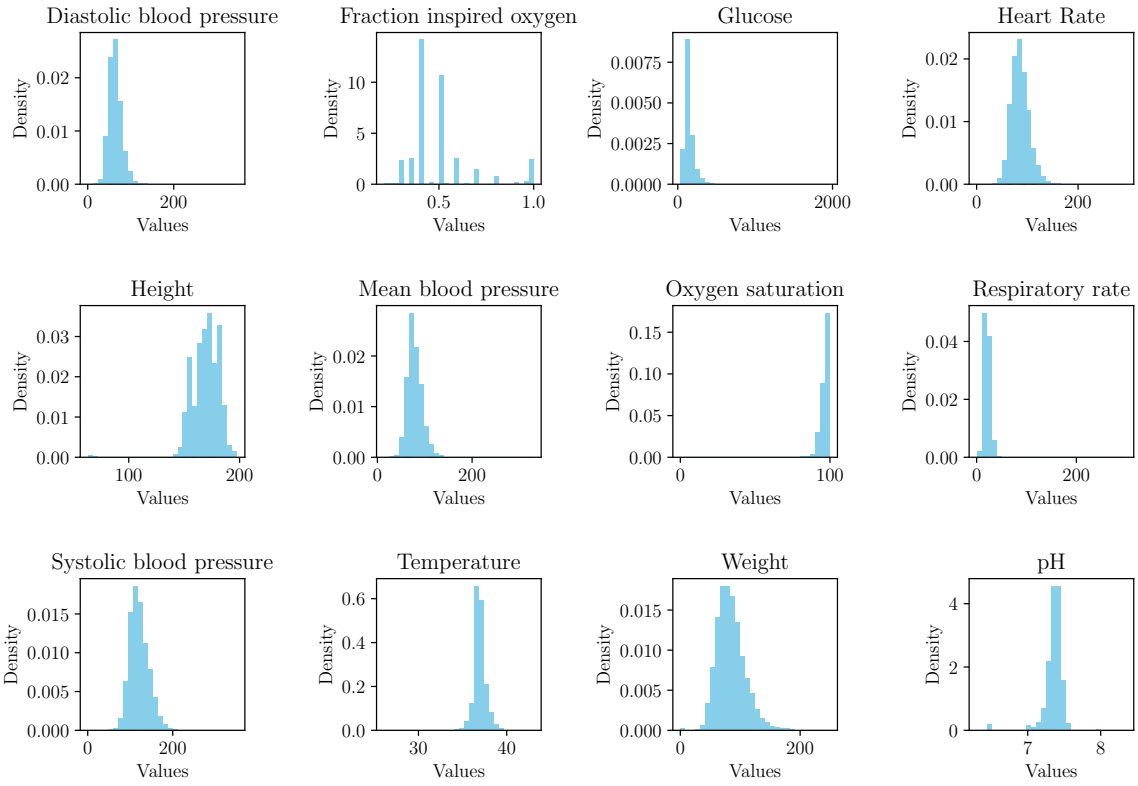


Fig. 2: Visualization of continuous variables

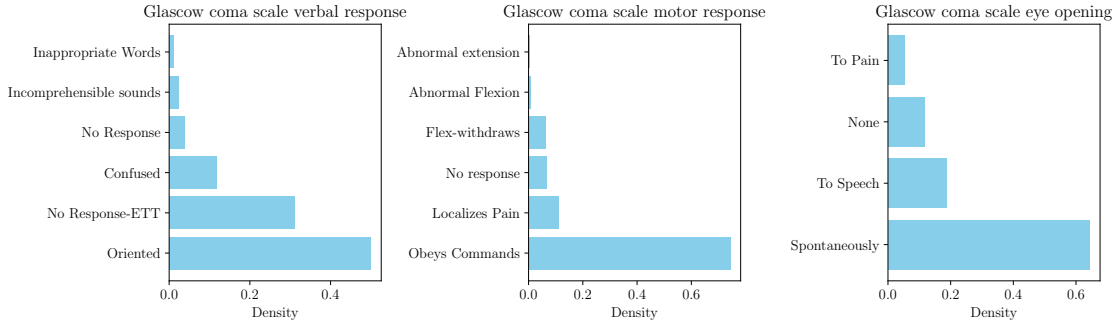


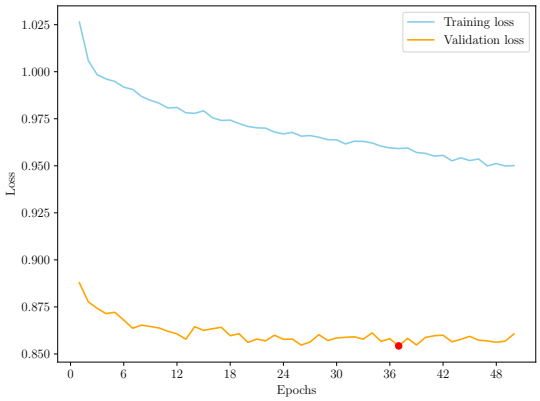
Fig. 3: Visualization of categorical variables

investigate if it has an effect on the models performance if it is trained on unique samples (*partial model*) or on samples that might be correlated due to multiple samples of one patient (*full model*). On Figure 1 one can see that distinct ICU stays are treated as samples and not individual patients.

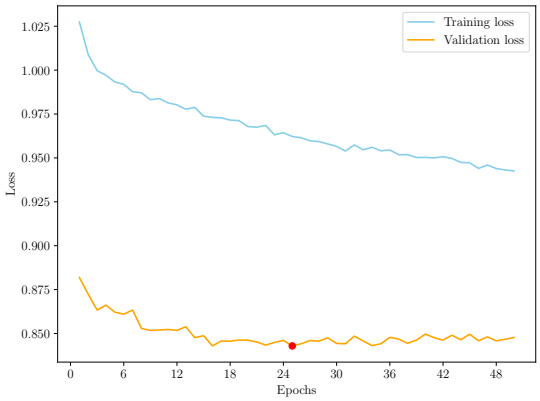
D. Experimental results

In Figures 4a and 4b the loss curves of both models were plotted in order to analyze the models' training behaviour. In the plots one can see the training and validation loss over epochs, illustrating the models' convergence and generalization capability. The red dot indicates the epoch where the best checkpoint of the model was saved according to the lowest validation loss.

Figure 4a shows the loss curves for the model that was trained on the full dataset. One can see that the training loss is continually decreasing and the model is able to learn based on the training data. In the beginning the validation loss is also decreasing but towards the end of training it start to plateau and even increase. Figure 4b displays the loss curves for the



(a) Loss curves for full model



(b) Loss curves for partial model

Fig. 4: Comparison of Loss curves

TABLE IV: Model evaluation on full test-set and test-set with unique patients only

Model	**Full test-set**			**Test-set with unique patients only**		
	macro AUROC	**micro AUROC**	**AUPRC**	**macro AUROC**	**micro AUROC**	**AUPRC**
Full	0.663	0.669	0.297	0.669	0.678	0.292
Partial	0.661	0.669	0.295	0.668	0.680	0.289

model that was only trained on a subset of the full dataset. The curves show a similar behaviour than for the full model. The validation loss is plateuing towards the end of training and might even start to increase.

Several attempts (learning rates from $1e{-}1$ to $1e{-}6$, weight decay from $1e{-}1$ to $1e{-}7$, weighted loss, different values for β_1, β_2 in ADAM) were made in order to try to improve the training behaviour of the model. Further other hyperparameters of RAINDROP were adjusted (`d_model`, `nhead`, `dropout`, `nhid`, `n_layers`) but without success and the best models are reported in this work.

Table IV shows the evaluation metrics for both models on the test set containing patients with multiple ICU stays and on the test set only containing unique patients.

For the evaluation AUROC and AURPRC were used. AUROC is a widely used metric to evaluate classification performance. It is well suited for imbalanced datasets. AURPRC is a performance metric that provides information about precision and recall trade-off, especially when the positive classes are rare.

For both models the macro and micro averaged AUROC and the macro averaged AUPRC were calculated and reported in Table IV. Besides of the micro averaged AUROC score on both datasets the full model outperforms the partial model by a small margin. It is not trivial to evaluate if the different dataset had an influence on the scores of the model. Given the small difference it is possible that the full model reaches slightly better scores due to the fact that it was trained on more samples.

In general the results are not satisfying and do not even

TABLE V: AUROC for each disease

Disease	**AUROC**
Acute and unspecified renal failure	0.683
Acute cerebrovascular disease	0.671
Acute myocardial infarction	0.665
Cardiac dysrhythmias	0.683
Chronic kidney disease	0.698
Chronic obstructive pulmonary disease and bronchiectasis	0.662
Complications of surgical procedures or medical care	0.679
Conduction disorders	0.671
Congestive heart failure; nonhypertensive	0.699
Coronary atherosclerosis and other heart disease	0.750
Diabetes mellitus with complications	0.602
Diabetes mellitus without complication	0.622
Disorders of lipid metabolism	0.697
Essential hypertension	0.651
Fluid and electrolyte disorders	0.655
Gastrointestinal hemorrhage	0.595
Hypertension with complications and secondary hypertension	0.707
Other liver diseases	0.652
Other lower respiratory disease	0.584
Other upper respiratory disease	0.602
Pleurisy; pneumothorax; pulmonary collapse	0.634
Pneumonia (except that caused by tuberculosis or sexually transmitted disease)	0.712
Respiratory failure; insufficiency; arrest (adult)	0.740
Septicemia (except in labor)	0.682
Shock	0.739

come close to other reported benchmark results [2] (MIMIC-III), [11].

Table V shows the AUROC scores of the model that was trained on the full dataset for each class. It is evaluated on the test set than contains multiple ICU stays per patient. The model differs in the AUROC scores for different diseases but there is no clear pattern that can be observed depending on the prevalence or the kind of disease.

E. Interpretation of lower dimensional embeddings

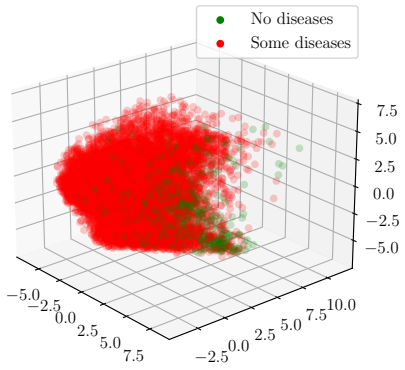


Fig. 5: Disease vs no Diseases

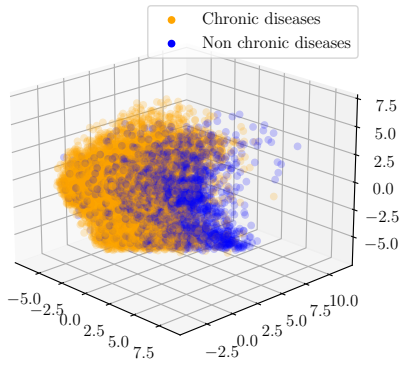


Fig. 6: Chronic vs non chronic diseases

The main principle of RAINDROP is to learn representative sample embeddings z_i. In order to analyze the embeddings and check if the meaning of them is interpretable, a Principal Component Analysis (PCA) to three dimensions was performed. Figure 5 displays the reduced embeddings and compares embeddings that correspond to samples with no diseases with samples that are associated with some diseases. Figure 6 compares embeddings of samples that are associated with chronic diseases (Chronic kidney disease, Chronic obstructive pulmonary disease, Coronary atherosclerosis and related, Diabetes mellitus without complication, Disorders of lipid metabolism, Essential hypertension and Hypertension with complications) with samples that are not associated with chronic diseases. In both scenarios one can see a grouping of the two classes but the clusters do not have clear boundaries. It seems like RAINDROP was able to learn sample embeddings based on some characteristics. It would be interesting to see if the representational capacity of the embeddings would have more meaning if the model learned better on the classification task.

V. Discussion

This work explored the RAINDROP model to tackle the difficulties of multivariate irregular time series data. The data used in this work was extracted EHR data from the MIMIC-IV dataset. The goal was to predict clinical care conditions that are associated with patients from the dataset. The RAINDROP model leverages the strengths of graph neural networks and neural message passing in order to model time-vaying dependencies between sensors. This allows the model to avoid an imputing stage to obtain a regular sampled time series.

The main experiment in this work was based on a comparison between two models: one was trained on a full dataset, which included multiple ICU stays per patient, and the other model was trained on a reduced dataset, only containing one ICU stay per patient. The motivation of this comparison was to investigate whether the correleation between multiple samples of the same patient has an impact on the models' performance. The full model performed slightly better than the partial model. This result suggests that training on a larger dataset might slightly improve the performance of the model. Overall the the performance of both models is not satisfying and the obtained scores are relatively low compared to other benchmark results.

As mentioned in Section IV-D several attempts were made in order to boost the performance of the model. The code in the RAINDROP repository [5] is not well organized and documented and this made it hard to asses whether the imperfections arose due to the model or if there were other issues. Further the work of Hayat et al. [11] does not really well document the preprocessing steps of their apdapted pipeline which made it hard to check whether the data was successfully pre-processed in the same way. Another problem of the dataset was the high class imbalance, which is also not discussed by Hayat et al. [11]. Balancing the batches improved the performance of the model model but other approaches that took the imbalanced into consideration were without success. A limitation of the MIMIC-IV dataset is the absence of timestamps which indicate when a disagnosis with a particular disease was made. This makes it impossible to perform a reliable disease forecasting with the given EHR data. Further approaches to check the sanity of the model and the data pre-processing would include training the model on different

[5] https://github.com/mims-harvard/Raindrop/tree/main

datasets and using baseline models on the pre-processed data. It might also be useful to use different pre-processing pipelines or to develop a new one.

Despite the limitations of this work, it offers a possibility for further research and improvements. The approach of RAINDROP is promising and needs to be explored further. In conclusion this work provides insights into using RAINDROP for phenotype classification based on multivariate irregular time series from EHR data of the MIMIC-IV dataset but more research is needed in order to optimize the models' performance and enable practical use cases and deployment to real-world scenarios.

Acknowledgements

I want to thank Dr. Jasmina Bogojeska for her support during the Semester and the Centre for Artificial Intelligence for the possibility to conduct my Master's at the institute.

References

[1] Alistair E. W. Johnson et al. "MIMIC-IV, a freely accessible electronic health record dataset". In: *Scientific Data* 10.1 (Jan. 2023). DOI: 10.1038/s41597-022-01899-x. URL: https://doi.org/10.1038/s41597-022-01899-x.

[2] Hrayr Harutyunyan et al. "Multitask learning and benchmarking with clinical time series data". In: *Scientific Data* 6.1 (2019). DOI: 10.1038/s41597-019-0103-9. URL: https://doi.org/10.1038%2Fs41597-019-0103-9.

[3] Xiang Zhang et al. "Graph-Guided Network for Irregularly Sampled Multivariate Time Series". In: *arXiv preprint arXiv:2110.05357* (2021). DOI: 10.48550/ARXIV.2110.05357. URL: https://arxiv.org/abs/2110.05357.

[4] Andrew Baumgartner et al. "Imputing Missing Observations with Time Sliced Synthetic Minority Oversampling Technique". In: *arXiv preprint arXiv:2201.05634* (2022). DOI: 10.48550/ARXIV.2201.05634. URL: https://arxiv.org/abs/2201.05634.

[5] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *arXiv preprint arXiv:1406.1078* (2014). DOI: 10.48550/ARXIV.1406.1078. URL: https://arxiv.org/abs/1406.1078.

[6] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[7] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179. URL: https://aclanthology.org/D14-1179.

[8] Ashish Vaswani et al. "Attention Is All You Need". In: *Advances in neural information processing systems* 30 (2017). DOI: 10.48550/ARXIV.1706.03762. URL: https://arxiv.org/abs/1706.03762.

[9] Brian J. Wells et al. "Strategies for Handling Missing Data in Electronic Health Record Derived Data". In: *eGEMs (Generating Evidence &amp Methods to improve patient outcomes)* 1.3 (Dec. 2013), p. 7. DOI: 10.13063/2327-9214.1035. URL: https://doi.org/10.13063/2327-9214.1035.

[10] Steven Cheng-Xian Li and Benjamin Marlin. "A scalable end-to-end Gaussian process adapter for irregularly sampled time series classification". In: *arXiv e-prints* (2016), arXiv–1606. DOI: 10.48550/ARXIV.1606.04443. URL: https://arxiv.org/abs/1606.04443.

[11] Nasir Hayat, Krzysztof J Geras, and Farah E Shamout. "MedFuse: Multi-modal fusion with clinical time-series data and chest X-ray images". In: *Machine Learning for Healthcare Conference*. PMLR. 2022, pp. 479–503. DOI: 10.48550/ARXIV.2207.07027. URL: https://arxiv.org/abs/2207.07027.

[12] Alistair E.W. Johnson et al. "MIMIC-III, a freely accessible critical care database". In: *Scientific Data* 3.1 (May 2016). DOI: 10.1038/sdata.2016.35. URL: https://doi.org/10.1038/sdata.2016.35.

[13] Sanjay Purushotham et al. "Benchmarking deep learning models on large healthcare datasets". In: *Journal of Biomedical Informatics* 83 (July 2018), pp. 112–134. DOI: 10.1016/j.jbi.2018.04.007. URL: https://doi.org/10.1016/j.jbi.2018.04.007.

[14] Aishwarya Mandyam et al. "COP-E-CAT". In: *Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*. ACM, Aug. 2021. DOI: 10.1145/3459930.3469536. URL: https://doi.org/10.1145/3459930.3469536.

[15] M. Gupta et al. "An Extensive Data Processing Pipeline for MIMIC-IV". In: *Proc Mach Learn Res* 193 (2022), pp. 311–325.

[16] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *arXiv preprint arXiv:1412.6980* (2014). DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980.