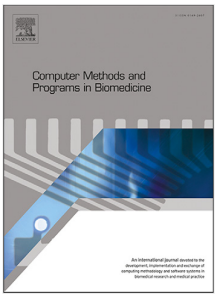
Journal Pre-proof



Understanding deep learning models for Length of Stay prediction on critically ill patients through latent space visualization

Lyse Naomi Wamba Momo, Vincent Scheltjens, Wouter Verbeke, Frank Rademakers, Bart De Moor

PII: S0169-2607(25)00249-4
DOI: https://doi.org/10.1016/j.cmpb.2025.108832
Reference: COMM 108832

To appear in: *Computer Methods and Programs in Biomedicine*

Received date : 2 October 2024
Revised date : 21 March 2025
Accepted date : 2 May 2025

Please cite this article as: L.N.W. Momo, V. Scheltjens, W. Verbeke et al., Understanding deep learning models for Length of Stay prediction on critically ill patients through latent space visualization, *Computer Methods and Programs in Biomedicine* (2025), doi: https://doi.org/10.1016/j.cmpb.2025.108832.

This is a PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article. Please note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

© 2025 Published by Elsevier B.V.

Revised Manuscript (clean) Journal Pre-proof Click here to view linked References

Understanding Deep Learning Models for Length of Stay Prediction on critically ill patients through Latent Space Visualization

Lyse Naomi Wamba Momo[a], Vincent Scheltjens [*a,b], Wouter Verbeke[b], Frank Rademakers[c], and Bart De Moor[a]

[a]KU Leuven, Department of Electrical Engineering (ESAT), Stadius Center for Dynamical Systems, Signal Processing and Data Analytics, Leuven, Belgium
[b]Faculty of Economics and Business, KU Leuven, Belgium
[c]Division of Cardiovascular Imaging and Dynamics, Department of Cardiovascular Sciences, KU Leuven, Leuven, Belgium

March 21, 2025

Journal Pre-proof

*Corresponding author: vincent.scheltjens@kuleuven.be
Address: ESAT-STADIUS, Stadius Centre for Dynamical Systems, Signal Processing and Data Analytics, Kasteelpark Arenberg 10, 3001 Leuven, Belgium.

Journal Pre-proof

Abstract

Background and Objective: Continuous, real-time monitoring of Length of Stay (LoS) for critically ill patients in Intensive Care Units (ICUs) is essential for anticipating patient needs, reduce the risk of adverse events, optimize resource allocation, plan incoming patients, and improve overall care. While previous research has focused primarily on predicting LoS, less attention has been given to how these prediction systems can be used by non-machine learning experts in real hospital environments for capacity planning.

Methods: In this work, we predict remaining ICU LoS using data from the Amsterdam University Medical Center dataset, which is the first freely available European critical care dataset and shows higher variability in LoS compared to U.S. datasets. We applied state-of-the-art sequence-to-sequence deep learning models—Long Short-Term Memory Networks, Gated Recurrent Units, Temporal Convolutional Networks with and without attention, and Transformer models—on 271 input features extracted from 20,481 ICU stays. Additionally, the latent spaces of these models were extracted, projected onto a 2D space and explored interactively to intuitively understand how the models learn patterns from the clinical data over time.

Results: The TCN model with attention (TCN-att) returned the best performance, reducing the prediction error by 2.24 days from a MAE of 6.94 days to a MAE of 4.70 days. Latent space analysis revealed that with just 5-6 hours of patient data, the models could clearly differentiate between short ($0 < \text{LoS} \leq 3$ days) and long ($7 \leq \text{LoS} < +\infty$ days) ICU stays, identify a patient cluster at risk of in-hospital mortality, and more.

Conclusion: Beyond reporting prediction error metrics, this study shows how an interactive dashboard can be used to gain insights into the inner workings of complex algorithms and how these can be integrated into clinical decision support systems. This approach allows for real-time intuitive understanding of how models learn and how patient predictions evolve, facilitating their use in real hospital environments.

1 Introduction

Inpatient Length of Stay (LoS) is an important economic and managerial metric used to evaluate performance, resource utilization and care quality improvement both at the hospital level as well as for specific units such as, the intensive care unit (ICU) [26, 18, 33]. LoS, defined as the length of time (often measured in days) of an inpatient between admission and discharge in a single care episode [29] is also a prognostic tool that gives an indication of the patient recovery trajectory. For critically ill patients admitted to the ICU, a unit highly solicited for care [27], continuous monitoring of their LoS is important with respect to both clinical and managerial aspects. On the managerial side, LoS monitoring will help in the efficient allocation of resources such as beds and other logistics to incoming patients. Regarding the clinical aspects, it will help in the early identification of risk of deterioration and/or death [9, 31, 41, 11, 28, 27]. Using historical patient data stored in electronic health records (EHRs), LoS prediction models can be developed and embedded into clinical decision support systems (CDSS) [35]. Such systems can help in the early detection of deterioration in inpatients' states, monitor patients' recovery trajectories, reduce the risk of occurrence of adverse events, and more [39, 11, 9, 17, 31, 41, 28, 27].

Modeling LoS is a complex task as has been highlighted by a number of authors [32, 1]. The first complexity arises from the interplay between the multiple factors that affect LoS which are dependent both on the individual patient specificity as well as the hospital or unit mode of functioning. The second complexity comes from the data-related challenges which include the sequential nature of the data and the skewness of the outcome. The time dimension naturally occurring when collecting clinical data during patient hospitalization and that needs to be taken into account during modeling in order to update LoS predictions, has made deep learning models, specifically, sequence-to-sequence models, more suitable for modeling as compared to static machine learning models. The skweness in the outcome distribution has often led a number of researchers to change the outcome structure from continuous to discrete classes, predicting the probability of remaining LoS under or above a pre-defined threshold, e.g., < 5 days, ≥ 7 days, etc., as in [17, 41, 19]. The third challenge stems from the use of deep learning models, which are built by successively combining non-linear functions [3]. These models, often referred to as black-box models due to their

Journal Pre-proof

complex inner workings structure, have contributed to slowdown the integration of these prediction systems into CDSS and their adoption by hospital management systems [9, 8, 4]. LoS prediction models in CDSS are valuable tools for efficient capacity planning that can be used to serve as support to both clinicians and capacity planners towards effective patient monitoring and discharge. For this purpose, more granular predictions are preferable to binary classifications. Additionally, developing explainability tools is crucial to understand both the features driving predictions and the internal mechanics of the models and the inference process. This transparency can help incentivize medical institutions to adopt such AI-driven models.

The Length of Stay prediction problem has attracted significant attention from researchers, as evidenced by numerous studies in the literature [27, 28, 19, 9, 31, 41, 11]. The intricate nature of LoS prediction, coupled with the substantial volume of patient-generated data during hospital stays, creates an ideal scenario for applying data-intensive deep learning models. Previous studies have employed these deep learning models for studying the complex interactions within clinical data for LoS and other clinical outcomes [9, 27, 31, 25, 39, 17, 23], often benchmarking them against static machine learning models. The latter typically required manually engineered features and generally reported lower performance in comparison to their deep learning counterparts [31, 25, 12]. When addressing model interpretability, researchers often focus on global model interpretability by ranking input features based on their contribution to the model's predictions. This method explains how the presence of a particular attribute in a patient's data impacts the predicted outcome. However, the end-users of CDSS might be interested in more granular information such as, the inner workings of the models at each time step, how patients' predictions may change over time and which attribute influences such a change (local interpretability). Such information is very important to plan future admissions. For deep learning models, gradient-based explanatory models such as the integrated gradients method [34] are often used. However, these methods are not very easy to understand by clinicians and hospital planners. This work aims to bridge this gap in model interpretability by introducing a simple and comprehensive approach to enhance model explainability.

In this study, we present a CDSS for remaining ICU LoS prediction for patients admitted to the Amsterdam University Medical Center (AUMC). The Amsterdam University Medical Center (AUMC) [36] is the first freely available European critical care database containing data from a 32-bed mixed surgical-medical academic ICU and a 12-bed medium care unit. In contrast to the existing US-based ICU databases where patients' stays are typically shorter, here we experience longer stays and a higher variability in the LoS of almost 10 days. Previous research on AUMC has established a benchmark for readmission and/or mortality prediction within 7 days of ICU discharge, using static machine learning models among which logistic regression and gradient boosted trees [36]. In this work, we establish a benchmark for remaining LoS prediction using deep learning models and we build an interactive dashboard to intuitively understand the prediction models over time, using the models' latent spaces. The key contributions of this work are as follows:

- We predict hourly remaining LoS for critically ill AUMC patients and compare the predictive performance of sequence-to-sequence models to that of static machine learning algorithms and naive baselines.
- We develop an interactive dashboard to visualize all sequential deep learning models' transformation of the inputs, i.e., models' embeddings, on unseen test observations over time. This not only showed the models' capabilities of clustering patients over time with respect to their true LoS but also revealed a neighborhood of patients at risk of experiencing in-hospital mortality with only 5 hours of post ICU admission data.
- We perform an additional analysis inspired by the visualization in which model inputs are constructed using a sliding window l, with the optimal window value obtained by hyperparameter optimization and show that this approach is both time and resource efficient.

The remainder of this work is organized as follows; In Section 2, details about the dataset, preprocessing steps, the prediction task, and all methods used are given. This is followed by Section 3 where the results obtained are

Journal Pre-proof

presented. Section 4 discusses all results obtained and, finally, Section 5 concludes the work by highlighting the contributions, limitations, and future work. The code developed in this project will be available on the following repository: github.com/LyzeNaomi/LoS_embeds. This code includes a data extraction and preprocessing pipeline adapted from [27] to manipulate the raw AUMC data of over 200GB on any computer.

2 Materials and Methods

This section discusses all components of the modeling process starting from the data used and preprocessing steps up to the prediction task and all the models used.

2.1 Dataset and preprocessing

The Amsterdam University Medical Center (AUMC) [36] is the first freely available European critical database containing detailed data points of 23,106 adult ICU stays from 20,109 patients admitted to the Amsterdam University Medical Center between 2003 and 2016. 115 input features (laboratory results and vital signs) from 20,481 stays pertaining to 19,360 unique adult patients with a LoS $>$ 5 hours were carefully extracted from the original database following a systematic procedure. The high level descriptive statistics of the data are given in Table 1. Only

Table 1: Descriptive statistics of data cohort

No. ICU stays	20481
Train	14329
Test	3066
Validation	3086
Basic demographic statistics	
Age class (mode)	60-69
Gender (Male)	64%
In-hospital mortality	31.57%
LoS statistics in days	
mean	4.60
median	1.08
std	9.61
skewness	6.04

laboratory features present in at least 25% of stays and vital signs present in at least 35% of stays were retained for subsequent modeling and are presented in Table 4 in Appendix A. After data extraction, each feature was hourly aggregated using the mean. Imputation of missing information was carried out similarly as in [27] by first forward filling the last recorded value per feature and per patient. Next, the original feature space was augmented with a matrix of an identical size and filled with decay curves. These curves indicate the time points at which data was imputed and the time from which imputation was done [27, 5, 20]. All demographic information available in the database was extracted and after one-hot encoding, the final input space resulted into 271 features consisting of time varying features (from lab results and vital signs) and repeated demographic information over the hours.

Further data preprocessing was needed for the static machine learning models. Combining strategies from previous works [31, 40, 12], different statistical measures were extracted from each of the original 104 original time series (without the decay curves). These statistics include the range (maximum-minimum), the difference between the last and first values, the standard deviation, the frequency of measurement, and the mean.

The data was divided into training, test, and validation sets using a 70:15:15 ratio.

Journal Pre-proof

2.2 Prediction task

Hourly remaining ICU LoS between admission and discharge or mortality is predicted, using data recorded up to 14 days (336 hours) post ICU admission. Formally, the prediction task is to approximate the true remaining LoS, y_{t+1} by $\hat{y}_{t+1}$ from learning the input transformation $f(\boldsymbol{X}_t)$ via the function f. The target y_{t+1}, is obtained from subtracting each hour $t(1, 2, ...)$ from the observed LoS (in hours).

The input sequences that are given to the models are constructed by initially using the first 5 hours of data to predict the remaining LoS at hour 6. Subsequently, one hour is added at a time to the sequence to predict the future remaining LoS. In the remainder of this work we will refer to this approach as the stacking approach.

2.3 Models

Naive models (mean and median training LoS), traditional machine learning models (linear regression and a 2-layer feed-forward neural network as used in previous works [31, 13]) and sequential models were all used for modeling.

2.3.1 Naive mean and median models

The naive models were obtained by extracting the mean and median LoS from the training set which were later used for prediction on the test set.

2.3.2 Linear Regression (LR)

Linear Regression (LR) is the simplest model in the regression setting where the outcome of interest is expressed as a linear combination of the input features, that is;

$$\hat{\boldsymbol{y}}_t = \boldsymbol{W}\boldsymbol{X} + \boldsymbol{b},$$

where $\boldsymbol{X}$ is the input set of hand-engineered features, $\hat{\boldsymbol{y}}_t$ is the predicted hourly remaining LoS, $\boldsymbol{W}$ is the weight parameter matrix updated during training and $\boldsymbol{b}$, the additive bias term.

2.3.3 Feed-forward Neural Network (FFN)

Together with the LR model, a 2-layer FFN was used with hand-engineered features as used in [31, 13]. This network consists of an input layer, two hidden layers and an output layer arranged sequentially and connected through weighted edges and non-linear functions. Information flows in one direction from one layer L to the next, starting from the input, through each hidden layer to the output. The output at each layer is computed using the following equation:

$$\boldsymbol{h}^L = g(\boldsymbol{W}^L\boldsymbol{h}^{L-1} + \boldsymbol{b}^L),$$

where, $\boldsymbol{h}^0 = \boldsymbol{X}$, $\boldsymbol{h}^3 = \hat{\boldsymbol{y}}_t$ and $\boldsymbol{W}, \boldsymbol{X}, \hat{\boldsymbol{y}}_t$ are defined as before. $g()$ is a non-linear activation function applied on the results of matrix operations. Here, the rectified linear unit (ReLU) activation function was used.

2.3.4 Long Short-Term Memory (LSTM)

LSTM [15] belongs to the family of recurrent neural networks (RNNs) that processes information one time step at a time, with gating mechanisms that control information flow. These mechanisms help decide at each discrete time step t how much of the information furthest in the past to forget and most recent to keep. In LSTM, these gates are the forget ($\boldsymbol{f}_t$), input ($\boldsymbol{i}_t$) and output gates ($\boldsymbol{o}_t$) that operate onto the memory cell state $\boldsymbol{c}_t$ to output timely

Journal Pre-proof

predictions $\boldsymbol{h}_t$. The LSTM network used here [10] is defined as follows,

$$\begin{aligned}\tilde{\boldsymbol{c}}_t &= g\left(\boldsymbol{W}_{\tilde{c}h}h_{t-1} + \boldsymbol{W}_{\tilde{c}x}\boldsymbol{x}_t + \boldsymbol{b}_{\tilde{c}}\right)\\ \boldsymbol{c}_t &= \boldsymbol{f_t}\cdot\boldsymbol{c}_{t-1} + \boldsymbol{i}_t\cdot\tilde{\boldsymbol{c}}_t\\ \boldsymbol{h}_t &= \boldsymbol{o}_t\cdot g(\boldsymbol{c}_t),\end{aligned} \tag{1}$$

where $\boldsymbol{h}_t$ is the hidden state at each discrete time t , $\boldsymbol{W}_{ab}$ are weight parameters from layers a to b and $\cdot$ is the element-wise product. The gate functions $\boldsymbol{f}_t, \boldsymbol{o}_t$ and $\boldsymbol{i}_t$ control the amount of information flow via a sigmoid (logistic) activation function and are defined as,

$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_{fh}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{fx}\boldsymbol{x}_t + \boldsymbol{b}_f\right), \boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_{ih}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{ix}\boldsymbol{x}_t + \boldsymbol{b}_i\right), \boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_{oh}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{ox}\boldsymbol{x}_t + \boldsymbol{b}_o\right)$$

When updating the cell and hidden states in (1), $\boldsymbol{i}_t$ determines which new information is to be stored in the cell state, $\boldsymbol{f}_t$ determines which information is to be disregarded and $\boldsymbol{o}_t$ determines the output from the cell state. The final prediction $\hat{\boldsymbol{y}}_t$ is obtained by extracting the last time step of the hidden state $\boldsymbol{h}_t \in \mathbb{R}^{B\times D}$ and passed through a linear layer with a non-linear ReLU activation function, i.e.,

$$\hat{\boldsymbol{y}}_t = ReLU(\boldsymbol{W}\boldsymbol{h}_t + \boldsymbol{b}), \tag{2}$$

B is the batch size and D the hidden dimension of the LSTM network.

2.3.5 Gated Recurrent Unit (GRU)

The GRU [6] is very similar to LSTM though with reduced complexity in terms of computation and implementation since they have only two gates, the reset $\boldsymbol{r_t}$ and update $\boldsymbol{z_t}$ gates that operate on the hidden states $\boldsymbol{h_t}$. While $\boldsymbol{z_t}$ controls how much of the past information $\boldsymbol{h_{t-1}}$ contributes to $\boldsymbol{h_t}$, $\boldsymbol{r_t}$, controls how much of $\boldsymbol{h_{t-1}}$ is to be ignored when considering the future. The underlying equations of the GRU model are defined as,

$$\tilde{\boldsymbol{h}}_t = \tanh\left(\boldsymbol{W}_x\boldsymbol{x}_t + r_t\cdot\left(\boldsymbol{W}_h\boldsymbol{h}_{t-1}\right) + \boldsymbol{b}_h\right), \boldsymbol{h}_t = \boldsymbol{z}_t\boldsymbol{h}_{t-1} + \left(\boldsymbol{1} - \boldsymbol{z_t}\right)\tilde{\boldsymbol{h}}_t$$

where the gates that control information flow through the GRU $\boldsymbol{r}_t$ and $\boldsymbol{z}_t$ are given by,

$$\boldsymbol{r}_t = \sigma\left(\boldsymbol{W}_{rx}\boldsymbol{x}_t + \boldsymbol{W}_{rh}\boldsymbol{h}_{t-1} + \boldsymbol{b}_r\right), \boldsymbol{z}_t = \sigma\left(\boldsymbol{W}_{zx}\boldsymbol{x}_t + \boldsymbol{W}_{zh}\boldsymbol{h}_{t_1} + \boldsymbol{b}_z\right)$$

The output prediction $\hat{\boldsymbol{y}}_t$ is obtained as in Equation 2.

2.3.6 Transformer network

The attention-based transformer network from [38] that captures dependencies in the data by weighting different parts of the input sequence was also used. From the original architecture, only the encoder block was used, where first, input data was compressed using a 1D-convolution network, to which deterministic positional encodings were added and then attention scores were calculated using h_e number of heads obtained from optimization. Three main operations were carried on in this work: input embedding using a 1D-convolution network in (3), multiple attention scores calculation in (4) and deterministic positional encoding calculation in (5). That is,

$$\tilde{\boldsymbol{e}}(N, C_{out}) = \sum_{k=0}^{C_{in}-1} \boldsymbol{W}_{\tilde{e}x}(C_{out}, k) \star \boldsymbol{X}(N, k) + \boldsymbol{b}_{C_{out}}, \tag{3}$$

Journal Pre-proof

where k is the kernel size used to slide over the input sequence, C_{in}, the number of input channels (corresponds to the input feature size), C_{out}, the number of output channels, N is the batch size and $\star$, the inner product.

Given $Q^{h_e}(\boldsymbol{x}) = W_{qh_e}\boldsymbol{x}, K^{h_e}(\boldsymbol{x}) = W_{kh_e}\boldsymbol{x}, V^{h_e}(\boldsymbol{x}) = W_{vh_e}(\boldsymbol{x})$ as the query, key and value matrices obtained from the sum of input embeddings $\tilde{\boldsymbol{e}}$ and positional encodings $\boldsymbol{p}$, the attention scores $\boldsymbol{\alpha}$ and the hidden outputs $\boldsymbol{h}$ are computed as,

$$\begin{aligned}
\boldsymbol{\alpha}^{h_e}(Q,K) &= \text{softmax}\left(\frac{\langle Q^{h_e}(e), K^{h_e}(e)\rangle}{\sqrt{d_k}}\right) \\
\tilde{\boldsymbol{u}} &= \sum_{a=1}^{H} W_{\tilde{u}}\alpha^a V^a(e) \\
\boldsymbol{u} &= LayerNorm(\boldsymbol{e} + \tilde{\boldsymbol{u}}; \gamma_1, \beta_1) \\
\tilde{\boldsymbol{h}} &= ReLU(\boldsymbol{u}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2 \\
\boldsymbol{h} &= LayerNorm(\boldsymbol{u} + \tilde{\boldsymbol{h}}; \gamma_2, \beta_2),
\end{aligned} \tag{4}$$

where $\boldsymbol{e} = \tilde{\boldsymbol{e}} + \boldsymbol{p}, \gamma, \beta$ are learnable affine transform parameters, $\boldsymbol{W}$ are weight parameters, $d_k = C_{out}$ is the dimension of the input embeddings, and H the number of attention heads. Deterministic positional encodings were used as in [38] and given below,

$$p_{pos,2i} = \sin\left(\frac{pos}{1000^{2i/d_k}}\right), \; p_{pos,2i+1} = \cos\left(\frac{pos}{1000^{2i/d_k}}\right). \tag{5}$$

The output prediction $\hat{\boldsymbol{y}}_t$ is obtained as in Equation 2.

2.3.7 Temporal Convolutional Network

The temporal convolution network (TCN) by Bai et al. [2] was also used to learn the temporal dynamics in the patient data. This architecture is made up of a 1D-Convolution Neural Network (1D CNN) that convolves over the input sequence with a filter of size $k-1$, where k is the kernel size. The causal convolutions are dilated using multiple dilation factors d (incremented with network depth) to increase the network receptive field and therefore learn very long sequences. Mathematically, the dilated convolution operation F on an element at time t of the time series sequence is defined as:

$$F(t) = \sum_{i=0}^{k-1} f(i) \cdot \boldsymbol{X}_{t-d\cdot i},$$

where, $f : \{0, ..., k-1\} \to \mathbb{R}$ is the filter and $X \in \mathbb{R}^n$ is a 1-D input sequence and $\cdot$ is the dot product. Moreover, given the network can be very deep, residual connections [14] are inserted at the end of each layer to learn the layer inputs transformation function. The output of the TCN network $\boldsymbol{h}$ is then obtained as, $\boldsymbol{h} = g(\boldsymbol{X} + \mathcal{F}(\boldsymbol{X}))$ where $\mathcal{F}(\boldsymbol{X})$ is the output of the sequence of operations performed on the input $\boldsymbol{X}$ and g is the ReLU non-linear activation function. The final prediction $\hat{\boldsymbol{y}}_t$ is obtained as in Equation 2.

2.3.8 Temporal Convolution Network with attention

The Temporal Convolution Network with attention (TCN-att) was obtained by applying a self-attention mechanism to the last layer of the TCN network. That is, given the outputs at the last time step as $\boldsymbol{h} = g(\boldsymbol{X} + \mathcal{F}(\boldsymbol{X}))$, with $\boldsymbol{h} \in \mathbb{R}^{B\times C_{out}}$ where B is the batch size and C_{out} is the number of output channels of the 1D CNN, we apply a self-attention on $\boldsymbol{h}$ to extract the cross-correlations $\boldsymbol{\alpha}$ across the different output channels which is later added back to the output $\boldsymbol{h}$ before passing this to the prediction layer. The attention scores $\boldsymbol{\alpha}$ are computed following the first two lines of Equation 4 where $Q = K = V = \boldsymbol{h}$. The final output $\hat{\boldsymbol{y}}_t$ was then calculated as,

$$\hat{\boldsymbol{y}}_t = ReLU(\boldsymbol{W}_2 tanh(\boldsymbol{W}_1(\boldsymbol{h} + \boldsymbol{\alpha}))$$

Journal Pre-proof

2.4 Latent space projection and dashboard construction

In order to understand how the models learn the clinical information embedded in the patient data, we extracted the latent spaces of all models and applied dimensionality reduction techniques. These reduced representations were then visualized interactively on a dashboard. The t-Distributed Stochastic Neighbor Embeddings (t-SNE) [37] and the Uniform Manifold Approximation and Projection (UMAP) [22] were used to project the high dimensional embeddings spaces to a two-dimensional space. Though UMAP has proven to be superior to t-SNE in terms of scalability, speed and its ability to preserve the global structure in the data (that is, the inter-cluster distance between "semantically" closer structures), both methods are very similar in that they aim to minimize the divergence between the original high-dimensional data representation and the targeted lower-dimensional representation. The latter onto which the data points are projected[1]. The dashboard was constructed using both the Dash and Plotly python frameworks [30, 7].

The overall modeling process is shown in Figure 1.

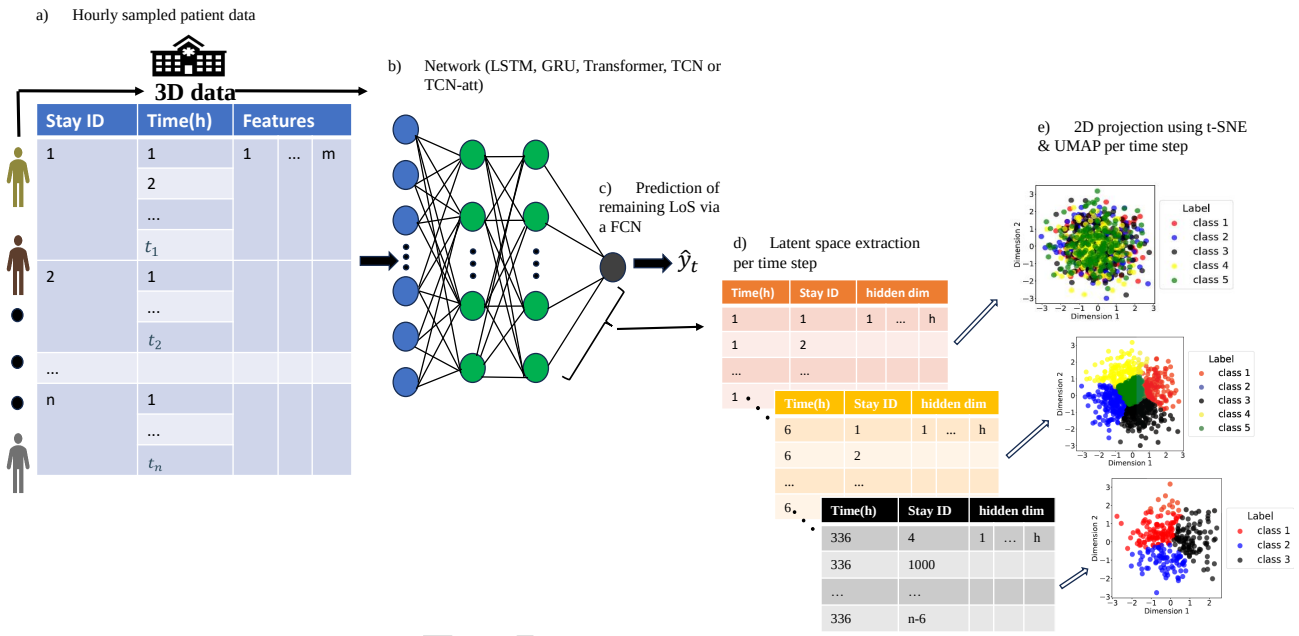


Figure 1: Overview of modeling process. In **a)** laboratory and vital signs extracted data are hourly sampled using the mean to produce a 3D tensor input where the time dimension varies across patients, i.e., $t_1 \neq t_2 \neq t_n$. The processed data is sent through five different neural networks (LSTM, GRU, Transformer, TCN and TCN-att) in **b)** and then through a single-neuron FCN for predictions as seen in **c)**. All models' embeddings obtained at the last layer in b) before being sent to c) are then extracted at each time step in **d)** between hour 1 - 336 and projected onto a 2D space in **e)**.

2.5 Evaluation metrics

The mean squarred log error (MSLE) was used as objective function during model training (to handle the skewness in the target distribution) and model performance was reported using the following metrics:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t|, MSLE = \frac{1}{n}\sum_{t=1}^{n}\left(\log(\boldsymbol{y}_t + 1) - \log(\hat{\boldsymbol{y}}_t + 1)\right)^2, MSE = \frac{1}{n}\sum_{t=1}^{n}(\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t)^2$$

[1]The UMAP [22] dimension reduction technique showed similar observations to the t-SNE and are presented in Appendix C

Journal Pre-proof

3 Results

This section presents all results obtained from hyperparamter optimization to model performance and latent space reduction and visualization, which will be subsequently discussed in Section 4.

3.1 Hyperparameter Optimization

All models were trained using the AdamW optimizer [21]. Hyperparameter optimization was carried out using both the grid search approach and population based training (PBT)[16]. PBT is an efficient optimization algorithm that initializes multiple models with hyperparameters randomly sampled from a given distribution. These models then run in parallel with poorly performing models being replaced by better-performing ones, which in turn receive new hyperparameters. This approach ensures a broad hyperparameter exploration while minimizing computational waste [16]. Grid search was employed to determine the number of hidden units and the number of layers before initiating PBT. Fixing these parameters was necessary to ensure there is a match between the new hyperparameters and the dimensions of the weight parameter matrix, which depend on both batch size and network depth. The obtained hyperparameters for each model are shown in Appendix B and Table 5. Given the relatively small dataset size (14,329 patients) and a high LoS variability of 9.61 days, the batch size appeared as an important hyperparameter during optimization. A large batch size led to poor performance, as the models struggled to learn patterns amidst the substantial variability in LoS.

3.2 Model performance

Upon stacking patient inputs one hour at a time for continuous remaining LoS prediction, each model performance and associated training time are reported in Table 2.

Table 2: Performance on the test set. The best performance is shown in bold. Mean ± standard deviation of metrics reported are calculated based on 15 iterations for each model. τ is the training time for one complete model run measured in hours.

Model	MAE	MSE	MSLE	τ
Mean	8.28	284.3	1.24	
Median	9.01	333.09	2.26	
LR	6.95±0.01	235.08±2.23	0.79±0.00	
FFN	6.94±0.02	233.01±2.20	0.77±0.01	
LSTM	6.59±0.05	227.62±6.81	0.68±0.01	6.17
GRU	6.61±0.12	228.10±4.94	0.68±0.01	**6.04**
Transformer	6.62±0.07	230.93±6.20	0.68±0.02	7.56
TCN	6.61±0.07	225.54±4.06	0.69±0.02	6.42
TCN-att	**4.70**±0.12	**175.09**±5.87	**0.27**±0.02	8.22

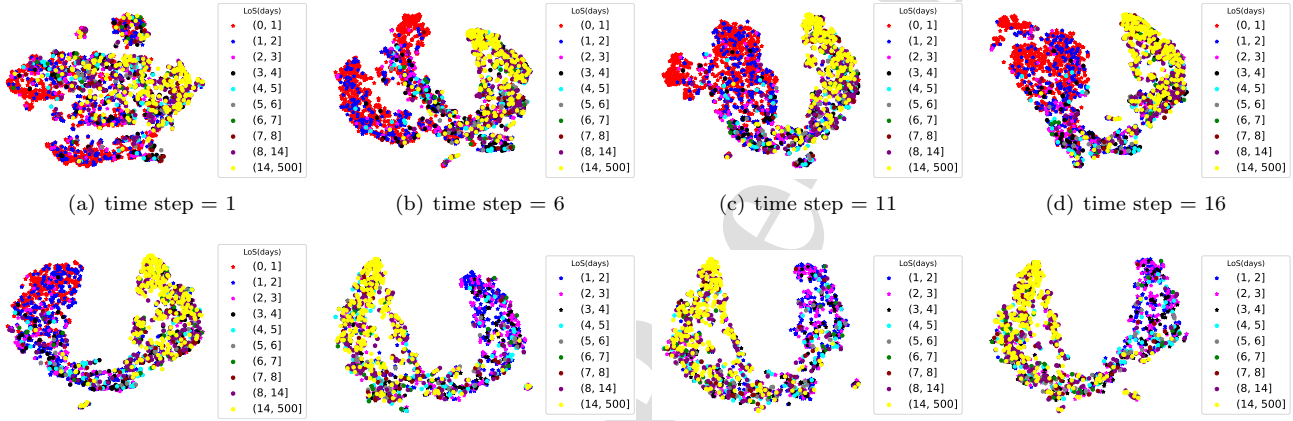
3.3 Latent space visualization

During modeling, the input data was passed through different layers of each model in parallel and the final representations at the last layer were sent to a FCN for predictions. The adequate number of neurons, i.e., the hidden dimension of each model was determined by hyperparameter optimization as 128 for LSTM, 64 for GRU, 32 for Transformer, 11 for TCN, and 64 for TCN-att. The trained models in the stacking input approach were applied on the unseen test data at inference. These enriched data representations, i.e., model embeddings were then extracted (before prediction) and projected onto a 2D space using both t-SNE and UMAP. In essence, for each model and at each discrete time step, the embeddings were extracted to create a new dataframe, in total producing 336 dataframes

(corresponding to each hour of data), each populated with embeddings of those patients not yet discharged at hour t (as shown in Figure 1d)) and then projected. The 2D projections, with each point on the scatter plot representing a test patient, were then labeled using the corresponding true LoS converted to ten classes. This exercise was carried out in an attempt to understand how each model learned the stacked input data at each time step thereby contributing to enhance model explainability.

3.3.1 Model understanding of clinical information

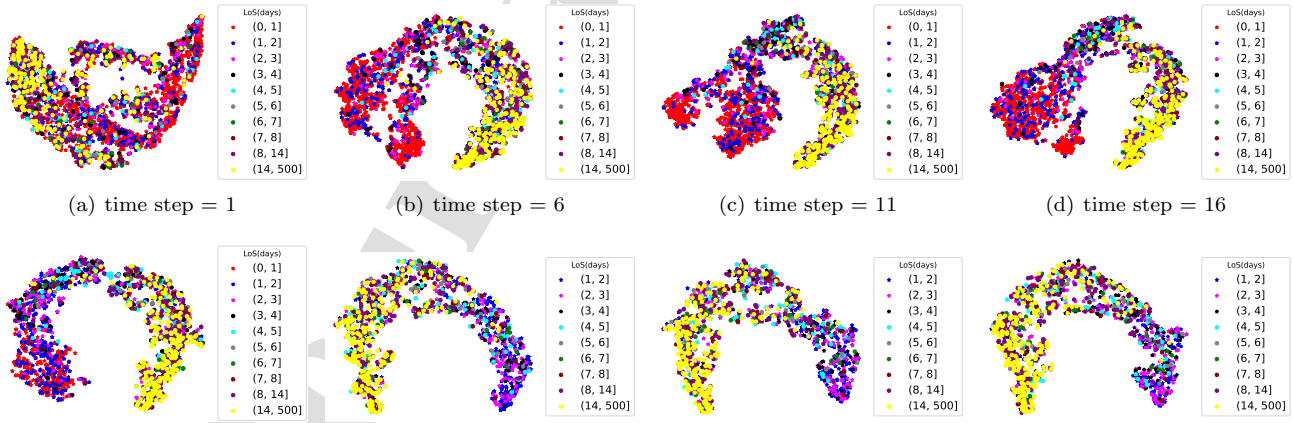
The t-SNE projections of all models' embeddings at specific time points (hours) are shown in Figure 2, Figure 3, Figure 4, Figure 5, and Figure 6 for LSTM, GRU, the Transformer model, the TCN, and the TCN-att respectively. These visualizations are seen on the interactive dashboard demo provided as supplementary material.



(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 2: t-SNE 2D projections of LSTM model embeddings, labeled with LoS.

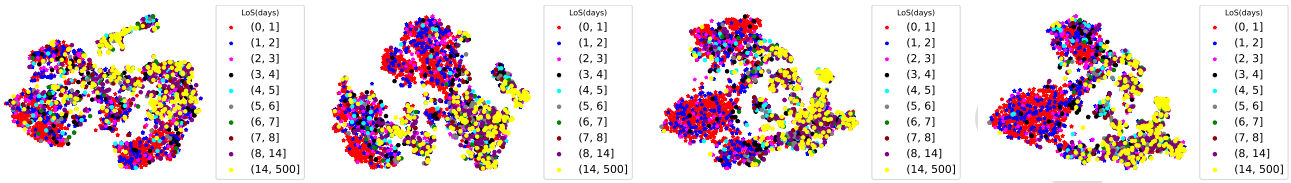


(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

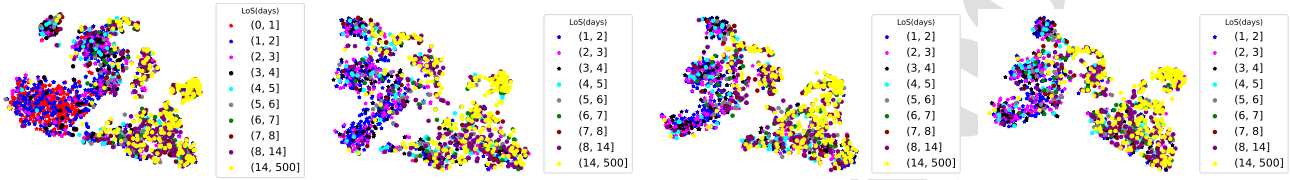
(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 3: t-SNE 2D projections of GRU model embeddings, labeled with LoS.

Journal Pre-proof

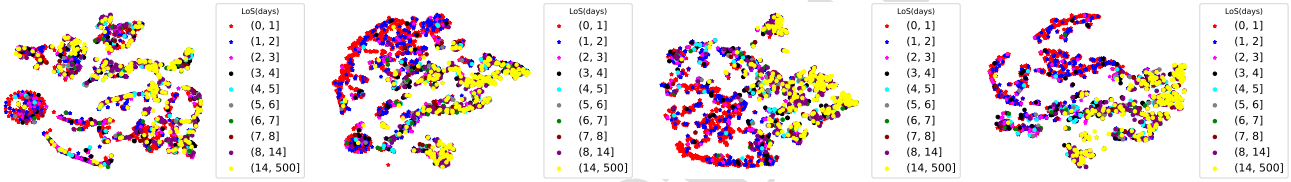


(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

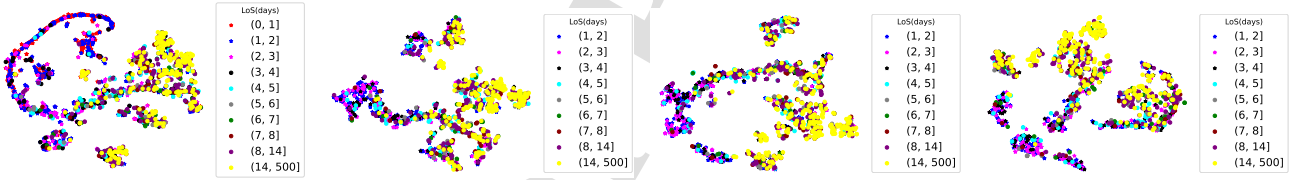


(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 4: t-SNE 2D projections of the Transformer model embeddings, labeled with LoS.

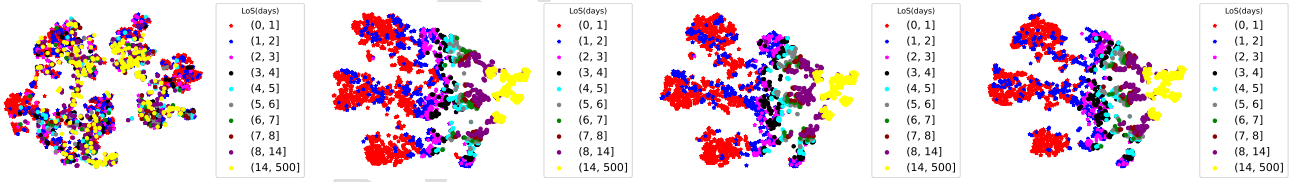


(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

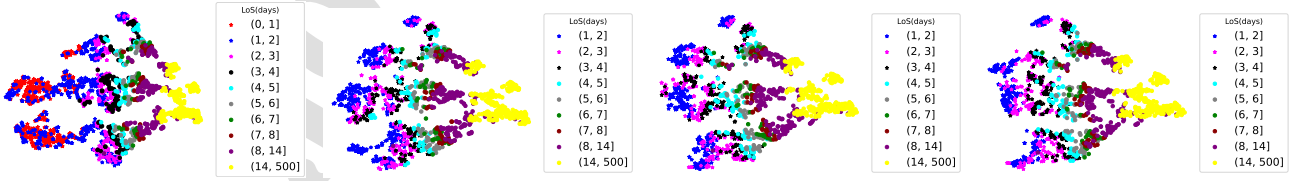


(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 5: t-SNE 2D projections of TCN model embeddings, labeled with LoS.



(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16



(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

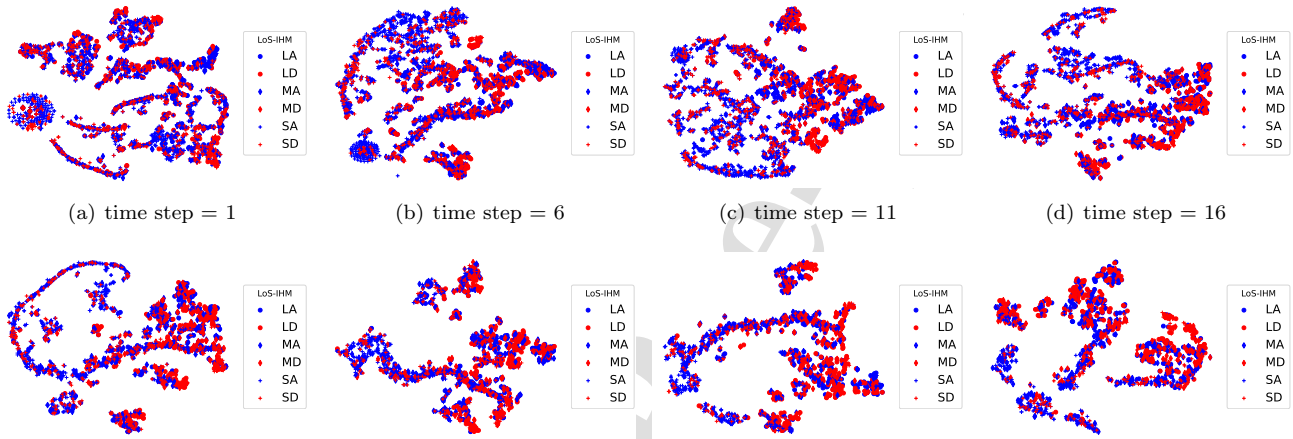
Figure 6: t-SNE 2D projections of the TCN-att model embeddings, labeled with LoS.

Journal Pre-proof

3.3.2 Model identification of patients at risk of in-hospital mortality

In-hospital mortality used as labels

The extracted and projected model embeddings per time step were also labeled using in-hospital mortality, a feature not used during modeling, neither as inputs nor output. For improved readability, the labels were split into 6 classes, i.e., SA, SD, MA, MD, LA, LD denoting short (S), medium (M) and long (L) stays for which the patients are either alive (A) or dead (D) at the time of discharge. More specifically, the intervals for the different LoS are constructed such that, S: $0 < \text{LoS} \leq 3$ days, M: $3 < \text{LoS} \leq 7$ days and L: $7 < LoS < +\infty$ days.
Figure 7 shows the t-SNE projections of TCN embeddings, labeled with both LoS and in-hospital mortality. On Figure 7, survivors are colored in blue and patients that eventually die appear in red.



(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 7: t-SNE 2D projections of TCN model embeddings, labeled with both LoS and in-hospital mortality.

Early identification of a cluster of patients at high risk of in-hospital mortality (IHM)

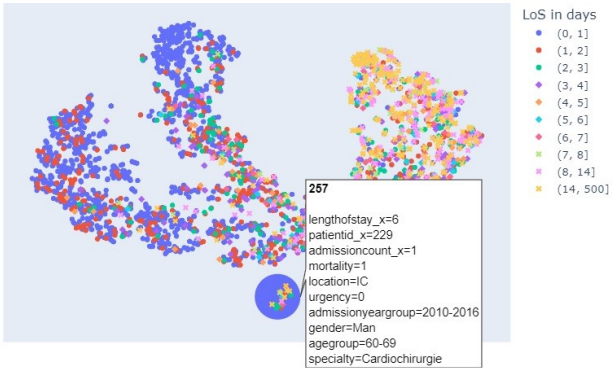
Using the interactive dashboard and the LSTM model, we monitored patient id 257 from hour 1 to hour 5 (just before dead at hour 6). At hour 5, when performing the last prediction of this patient, we identified that the neighborhood of patient id 257 (see Figure 8(a)), a neighborhood of exactly 50 patients including id 257 had an observed mean LoS=5.4 days (129.64 hours) and median LoS=0.81 days (19.5 hours) with a 94% mortality rate as shown in Figure 8(b) (this is visible from minute 01:48 to minute 02:20 of the dashboard demo) [2].

Identification of a near-to-discharge patient

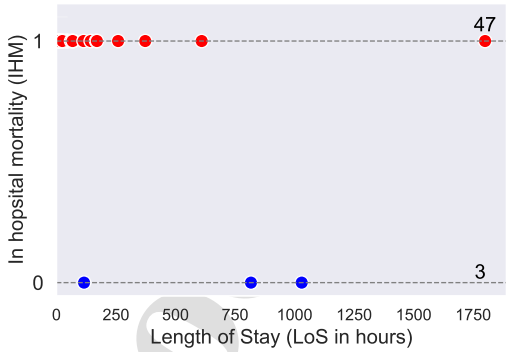
Using the interactive dashboard and the Transformer model, we discovered that patient id 6, a short stay patient with observed LoS=48 hours, was initially predicted as a long stay from hour 1 to hour 34 (as seen on Figures 9(a) and (b)). Subsequently, from hour 35 till hour 47 (just before discharge at hour 48), this patient was predicted correctly as a short stay. This is also seen on the dashboard demo from minute 02:23.

Investigating this with the original data, it appears that, patient id 6 was admitted to ICU after a neuro-surgery and placed in a coma from admission (hour 1) to hour 31 (as seen on Figure 10 where mechanical ventilation was administered).

[2]The same observation is made with the TCN and the Transformer models which also produce clusters of patients. Although there is no clear separate cluster on the TCN-att model, the neighborhood of patient 257 also shows a mortality rate of over 95%

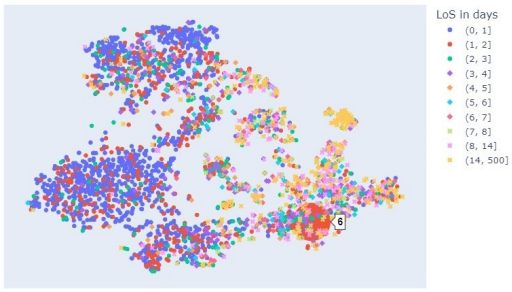


(a) LSTM embeddings at time step = 5 with id=257 highlighted



(b) LoS vs IHM plot of all 50 patients in the neighborhood of id=257. 94% (47 out of 50 patients) experienced IHM, some with very short LoS.

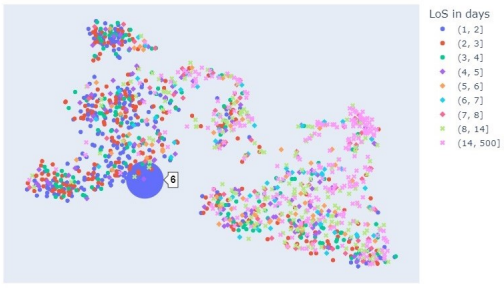
Figure 8: Cluster of 50 patients identified at hour 5 (left figure) where 94% of the patients will experience in-hospital mortality at different hours of their stays and for which 75% has a LoS ≤ 3 days (right figure).



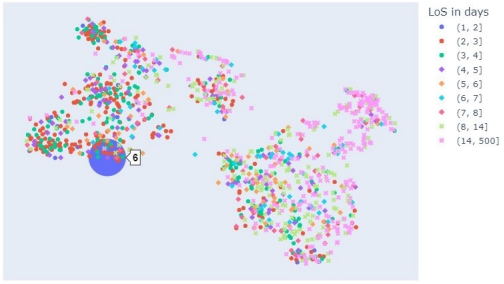
(a) time step = 11



(b) time step = 34



(c) time step = 35



(d) time step = 47

Figure 9: Patient id=6 predictions' over time shown at hours 11, 34, 35 and 47 using the Transformer model. This observation was made with all other models.

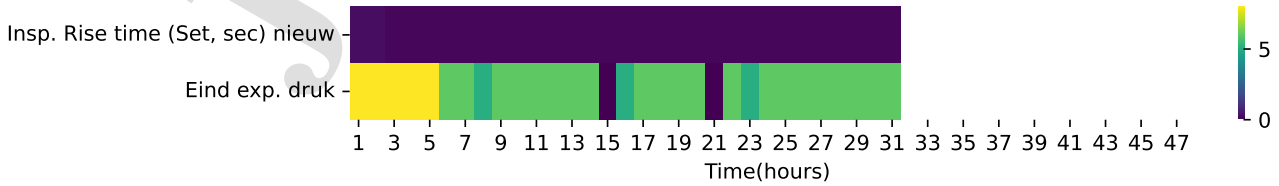
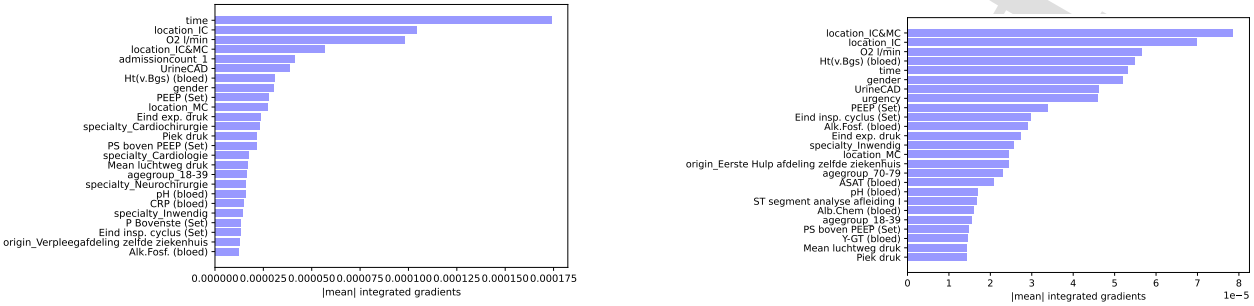


Figure 10: Heatmap of mechanical ventilation indicators from admission to hour 47

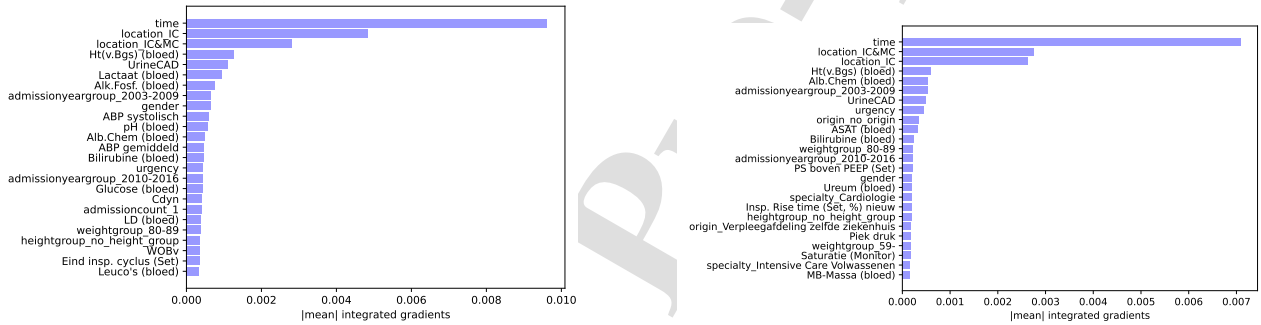
3.4 Feature Importance (Global Model Interpretability)

While the dashboard offers local interpretability for each patient at every hour, global model interpretability was achieved by identifying the key features influencing LoS predictions. These features were computed across all five models using the integrated gradients method [34] on the unseen test data and ranked based on their importance.



(a) LSTM (b) GRU

Figure 11: Top 25 most important features on test unseen patients using LSTM (a) and GRU (b)



(a) Transformer (b) TCN

Figure 12: Top 25 most important features on test unseen patients using Transformer (a) and TCN (b)

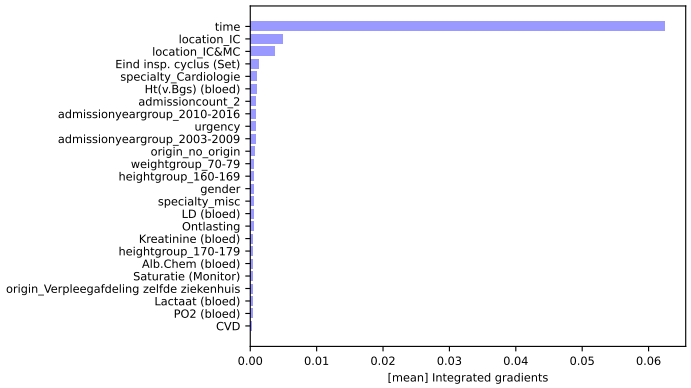


Figure 13: Top 25 most important features on test unseen patients using the TCN Attention model

3.5 Additional Analysis

The results from the visualization showed that the models are able to learn or gain knowledge from early data inputs (5 hours of data only), updating their predictions with time. Thus, instead of stacking input sequences (one

Journal Pre-proof

hour at a time) as often done in the literature, model input sequences were also constructed using a sliding window. By doing this, we also assess the relevance of using a long history length, i.e., early admission laboratory results and vital signs to predict remaining LoS 200+ hours later. The length of the window l was added to the list of hyperparameters. Intuitively, by testing different values of l, the models experiment with different history lengths, with a minimum of $l = 2$, i.e., using the most recent 2 hours of available data to predict remaining ICU at the next hour.

Table 3 reports model performance of the best performing model in this analysis and the associated value of the window l. Model performance with arbitrary and high values of the window l are also reported.

Table 3: Performance on test set with associated window length l reported. The best performance is shown in bold. Mean ± standard deviation calculated based on 15 iterations for each model. τ is the training time for one complete model run measured in hours.

Model	l	MAE	MSE	MSLE	τ
LSTM	4	6.57±0.06	226.65±4.95	0.66±0.01	4.20
GRU	3	6.44 ±0.04	221.60±3.67	0.66±0.01	**4.04**
Transformer	2	6.59±0.06	231.95±4.71	0.68±0.01	5.38
TCN	6	6.64±0.06	226.37±5.08	0.68±0.01	4.55
TCN-att	5	**4.70±0.12**	**179.50±7.42**	**0.28±0.02**	6.56
TCN-att	12	5.08±0.13	185.77±7.47	0.31± 0.02	5.21
TCN-att	24	5.44±0.69	200.86±32.33	0.31±0.83	5.37
TCN-att	48	6.82±1.79	260.38±67.17	0.86±1.50	3.12
TCN-att	72	6.67±0.19	256.72±11.81	0.42±0.03	2.94

4 Discussion

This section discusses all results previously presented.

4.1 Model Performance

Table 2 shows that the TCN-att model reduces the MAE by 2.24 days from 6.94 days to 4.70 days with just 2 additional hours of training.

4.2 Latent space visualization

4.2.1 Model understanding of clinical information

At time step=1, i.e., hour 1, all five models have not yet learned any pattern in the data, and with just a few more hours, at hour 6, a clear separation is visible between short ($0 < \text{LoS} \leq 3$ days represented as red, blue and magenta stars) and medium to long stays (LoS > 3 days) in Figure 2(b), Figure 3(b), Figure 4(b) and Figure 5(b). In particular, the TCN-att model in Figure 6(b) demonstrates a more distinct and precise separation between the different LoS classes at hour 6, highlighting its superiority over the other four models. Subsequently at hour 16, a third cluster appears for the LSTM, GRU and the TCN networks where medium stays ($3 < \text{LoS} < 7$ days represented in black, cyan and grey circles) are separated further from both the short stays (star-shaped in red, blue and magenta) and long stays ($7 \leq \text{LoS} < +\infty$ days represented as circled-shaped in yellow, purple, dark-red and green). These three clusters are more visible in Figure 2, Figure 3 and Figure 5, and the separation amplifies even further at hour 21. For the Transformer network, the medium stays ($3 < \text{LoS} < 7$ days represented in black, cyan and grey circles) form a separate cluster from the short and long stays only at hour 21, and just like TCN, the embeddings at this hour form multiple clusters compared to the three clusters seen in the recurrent architectures.

Journal Pre-proof

At further time steps (hours 26-31), this separation is again visible when the shortest 1 day (24 hours) patients are discharged and the remaining patients with similar LoS cluster more together. The TCN-att model on the other hand continuously improves its ability to separate the different classes over time, with only occasional confusion at the boundaries where classes with close LoS values overlap. Intuitively, all models gradually learn the clinical patterns that separate patients with different LoS, first isolating short stays, then medium stays. The TCN-att model shows the best ability in this regard.

As mentioned in the introduction, LoS prediction, i.e., anticipation on bed occupancy rate is important especially for hospital planning. Timely LoS predictions can enable effective planning of surgeries, resource allocation and more in ICU as well as in the entire hospital. In a true hospital setting, when patients are admitted, it is important to be able to separate the day patients from the overnights and within the overnights, short from medium and long stays. This will enable to plan incoming patients (when the short stays have been identified), and allocate more adequate resources to the identified long stays. The latent space visualization exercise then helps in this regard as it gives the intuition that early on admission data (only first 5-6 hours of post ICU admission data) can enable stays clustering in ICU.

4.2.2 Model identification of patients at risk of in-hopsital mortality

In-hospital mortality used as labels

From Figures 5 and 7, we observe that the long stays region (region in which yellow is most dominant on Figure 5) coincides with the dead patients region (red-dominant region on Figure 7) which is inline with the known correlation that exists between long stays and increased risk of mortality [36]. Given that all models were trained to predict solely the Length of Stay (LoS) without considering in-hospital mortality as either an input or output feature, the agreement between Figure 5 and Figure 7 (as explained above) shows that the trained models[3] effectively learned the clinical patterns in the patient parameters that can predict more than one clinical outcome. Thus, the models did not learn random noise.

In a clinical setting, this implies that patients likely to experience long stays and those at high risk of in-hospital mortality can be identified and prioritized as early as 5-6 hours after ICU admission. Risk stratification in the ICU is crucial, and mortality prediction has been extensively studied in the literature. Typically, mortality predictions are made using data from the first 24 hours post-ICU admission, similar to the APACHE score (Acute Physiology and Chronic Health Evaluation II), which serves as an early warning tool to identify at-risk patients and allow clinicians to adjust the care protocol accordingly [24]. However, our findings suggest that risk assessment can begin much sooner, within just 5-6 hours.

Early identification of a cluster of patients at high risk of in-hospital mortality (IHM)

By monitoring the predictions of patient id 257, that experiences IHM at hour 6, we observed that the neighborhood of this patient, constituted of 50 patients with over the majority having a LoS of less than 3 days (see Figure 8) eventually experience IHM. In a hospital, using prospective data, it means that, if a patient dies early on, few hours after admission, and appears in a cluster with other patients, these patients should then be monitored very closely as they may present a high risk of IHM as well.

Identification of a near-to-discharge patient

For capacity planners, it is very important to be able to identify patients that can be discharge in the near future so that upon confirmation with the unit head, subsequent plannings can be done on the previously occupied bed. Patient id 6 is initially predicted as a long stay patient from hour 1 to hour 34 (Figure 9) when the patient is

[3]This pattern obsered on the TCN model was also observed on all other three models.

Journal Pre-proof

maintained in a state of coma and is under mechanical ventilation from hour 1 to hour 31 (see Figure 10). When the patient extubated from hour 32, the model then adjust its prediction at hour 35 and predicts him as a short stay till discharge. The model could not predict that the patient was doing better and will eventually not need mechanical ventilation because this is indicated by a subsided cerebral oedema and a drop in the intracranial pressure, which information, was not available to the model as none of the 104 time series information extracted can be used as proxy for determining these.

4.3 Feature importance

Similarly to the embeddings plots where LSTM and GRU show a similar clustering, which is different from the Transformer model and the TCN network, here, the top discriminating features show a similar pattern. The LSTM and GRU models focus on the unit of admission of the patient (*location_IC, location_IC&MC, location_MC*) (this is also visible on the dashboard from minute 00:57 to minute 01:30), ventilation-related features (*O_2l/min, PEEP(Set), Eind insp.cyclus (Set), Eind exp.druk*), low-blood level indication (*Ht(v.Bgs)*) the gender (*male*), and a potential bedridden condition (UrineCAD which means catheterurine). These features which could indicate circulatory problems and the need for ventilation (that is, patients need to be intubated) are more likely to extend the stay.

The Transformer and TCN models in Figure 12 indicate the time variable as the most important, followed by the admission location, low-blood level (*Ht(v.Bgs)*), UrineCAD, emergency admission, circulatory problems related features (*Lactate, Alkanine Fosfatase, Albumin*). The time factor indicates that, the more time spent in ICU, the higher the risk of staying even longer.

Similar to the transformer and the TCN networks, TCN-att model in Figure 13 also highlights the time variable as an important feature, followed by the admission units of the patients. This model highlights similar features as the previous models including ventilation-related (*Eind insp.cyclus (Set), Saturatie (Monitor), PO2(bloed)*, low-blood level (*Ht(v.Bgs)*), circulatory problems related features (*Lactate, Albumin).* However, we note that this model uses primarily static demographic and admission-related features for modeling the LoS.

4.4 Additional analysis

Table 3 shows the results of the sliding window approach. The results indicate that each model either shows slight improvements in performance or delivers comparable performance to its stacking counterpart, while requiring nearly 2 hours less of training time. Moreover, Table 3 shows that short window lengths (of only 2 hours as is the case for the Transformer network), can be effectively employed for model training, yielding performance similar to that achieved by stacking multiple hours of input data. This falls in line with gathered medical expertise (after discussion with experts) that most recently recorded time-varying patient information of laboratory tests and vital signs are most valuable for taking the decision to discharge a patient. In addition, it helps to circumvent memory issues encountered when training with very long input sequences (e.g. 300+ lines/hours of patient parameters per patient in the batch during stacking).

5 Conclusions and Limitations

In this study, we predicted hourly remaining Length of Stay (LoS) for critically ill patients admitted to Amsterdam UMC. Our results demonstrated that deep learning models outperform traditional machine learning methods and naive mean and median models, reducing the prediction error by 2.24 days (from a MAE of 6.94 days to a MAE of 4.70 days). Of the five sequential deep learning models evaluated, the Temporal Convolutional Network with attention (TCN-att) returned the best performance using both the stacking input approach and the sliding window

Journal Pre-proof

approach to input construction. The sliding window approach, in particular, delivered the highest performance while being computationally efficient, requiring fewer training resources and less time. Next, in order to provide an intuitive understanding of the inner workings of the models and therefore enhance model explainability, we applied dimensionality reduction techniques on all models' latent spaces and viewed these interactively with a developed dashboard. This analysis showed that with very few hours of data, i.e., 5-6 hours, all models are able to separate short stays ($0 < \text{LoS} \leq 3$ days) from medium ($3 < \text{LoS} < 7$ days) and long stay patients ($7 \leq \text{LoS} < +\infty$ days). A separation that amplified over time (between hours=16-21) where the medium stays were now also separated from the long stays. The TCN-att model demonstrated a distinct separation between extreme LoS classes very early on (5-6 hours of data), although some overlap remained among patients belonging to neighboring LoS classes. Moreover, by labeling these clusters not only with LoS but also in-hospital mortality, the model showed that it is able to capture the clinical information that associates long stays with increased risk of mortality. Other valuable insights drawn from visualizing the models embeddings included the early identification of a cluster of patients with a high risk of in-hospital mortality and the identification of a patient close to discharge. These insights are valuable both for capacity planning and for patient care.

One of the limitations of this work is that patient data over 14 days (336 hours) was not further extracted to be exploited in the sliding window format given the good performance reported and the lesser computational resources used in this setting. This will be tackled in future work. Subsequently, other architectures including graph neural networks that explicitly model the similarity between patients (similarity can be based on common medications, diagnoses, etc.) will be implemented to try to reduce the prediction error even further. Another limitation of this work is that the dashboard only displayed static features to show how the models utilized them during training. Time-varying features will be incorporated into the dashboard in future updates.

Funding

This work was supported by KU Leuven: Research Fund (projects C16/15/059, C3/19/053, C24/18/022, C3/20/117, C3I-21-00316), Industrial Research Fund (Fellowships 13-0260, IOFm/16/004, IOFm/20/002) and several Leuven Research and Development bilateral industrial projects; Flemish Government Agencies: FWO: EOS Project no G0F6718N (SeLMA), SBO project S005319N, Infrastructure project I013218N, TBM Project T001919N; PhD Grants (SB/1SA1319N, SB/1S93918, SB/1S1319N), EWI: the Flanders AI Research Program VLAIO: CSBO (HBC.2021.0076) Baekeland PhD (HBC.20192204) and Innovation mandate (HBC.2019.2209) European Commission: European Research Council under the European Union's Horizon 2020 research and innovation programme (ERC Adv. Grant grant agreement No 885682); Other funding: Foundation 'Kom op tegen Kanker', CM (Christelijke Mutualiteit)

Appendix

A Inputs

Extracted features from the original database are give in Table 4 below.

Journal Pre-proof

Journal Pre-proof

Table 4: Input Features

Static	Lab	Non-Lab	Computed feature
admission count	ALAT	ABP diastolisch	Time in ICU
admission year group	APTT	ABP gemiddeld	
age group	ASAT	ABP systolisch	
gender	Act.HCO3	Adem Frequentie (Set)	
height group	Alb.Chem	Ademfreq.	
location	Alk.Fosf.	Ademfrequentie Monitor	
origin	Amylase	Backup druk (Set)	
specialty	Anion-Gap	Barometer druk	
urgency	B.E	CO2 min prod	
weight group	Bezinking	CO2 tidal productie	
	Bilirubine	CVD	
	CK	Cdyn	
	CO-Hb	Eind exp. druk	
	CRP	Eind insp. cyclus (Set)	
	Ca-ion (7.4)	End tidal CO2 concentratie	
	Calcium totaal	EtCO2 (%)	
	Chloor	Exp. minuut volume	
	Cl(onv.ISE)	Exp. tidal volume	
	Eryś	Hartfrequentie	
	Fosfaat	Insp. Minuut volume	
	Glucose	Insp. Rise time (Set, %) nieuw	
	Hb	Insp. Rise time (Set, sec) nieuw	
	Hb(v.Bgs)	Insp. Time (sec) nieuw	
	Ht	Insp. tidal volume	
	Ht(v.Bgs)	MVe Spontaan nieuw	
	K (onv.ISE)	Maaghevel	
	Kalium	Mean luchtweg druk	
	Kreatinine	Niet invasieve bloeddruk diastolisch	
	LD	Niet invasieve bloeddruk gemiddeld	
	Lactaat	Niet invasieve bloeddruk systolisch	
	Leuco's	O2 concentratie	
	MB-Massa	O2 concentratie (Set)	
	Magnesium	O2 l/min	
	Meettemp	Ontlasting	
	Met-Hb	PAP diastolisch	
	Na (onv.ISE)	PAP gemiddeld	
	Natrium	PAP systolisch	
	O2-Content	PC boven PEEP (Set)	
	O2-Saturatie	PEEP (Set)	
	OxyHb-Fr.	P Bovenste (Set)	
	PO2	PS boven PEEP (Set)	
	Prothrombinetijd	Piek druk	
	Prothrombinetijd	SBI (Zwakke ademhalingsindex)	
	Thrombo's	SIMV ademcyclus (Set)	
	TroponineT	ST segment analyse afleiding	
	Ureum	ST segment analyse afleiding I	
	UrineCAD	ST segment analyse afleiding II	
	WOBv	Saturatie (Monitor)	
	Y-GT (bloed)	Temp Axillair	
	eGFR(MDRD)	Thoraxdrain1 Productie	
	pCO2	Thoraxdrain1 Stand	
	pH	Ti (sec) nieuw	

Journal Pre-proof

Journal Pre-proof

B Hyperparameters Optimization

Hyperparamters obtained are showned in Table 5 below. For all four sequential deep learning models, the batch size hyperparameter refers to the number of unique stay ids randomly loaded at each epoch during training, validation and inference. When data for this unique number of stay ids is loaded, within the data loader, multiple sequences are then created (with an average number of sequences around 200) to predict remaining ICU LoS.

Journal Pre-proof

Journal Pre-proof

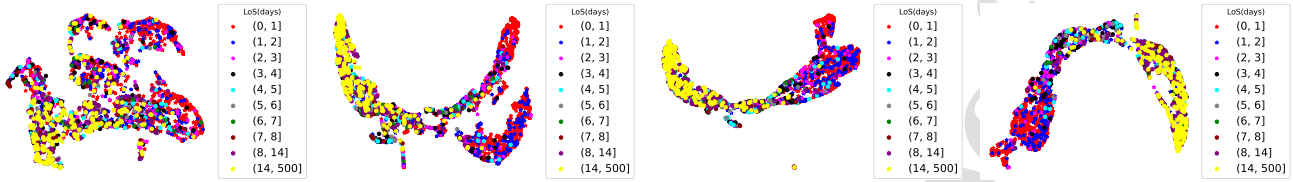
Table 5: Hyperparamters values obtained upon optimization

Models	Hyperparameters										
	No.layers	No.units	bsize	lr	dropout	wdecay	l	feed-forward dim	input embedding	No. heads(H)	No.channels ksize
LR			128	$8.36e^{-4}$		$3.98e^{-4}$					
FFN	2	256	128	$4.28e^{-5}$	0.15	$5.48e^{-5}$					
LSTM	2	128	4*	$9.46e^{-4}$	0.25	$6.21e^{-3}$	4				
GRU	3	64	4*	$5.64e^{-4}$	0.15	$1.67e^{-3}$	3				
Transformer	2	32	4*	$2.94e^{-4}$	0.05	$8.12e^{-3}$	2	32	1D Conv	2	
TCN	8		4*	$3.02e^{-3}$	0.05	$5.06e^{-4}$	6				11 4
TCN-att	3		4*	$9.4e^{-5}$	0.05	$9.88e^{-4}$	5			4	64 4

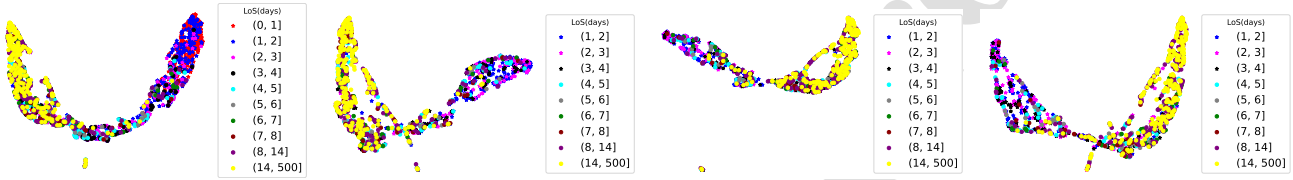
Journal Pre-proof

C UMAP visualization of model embeddings

Visualization of UMAP projection of all models' latent spaces.

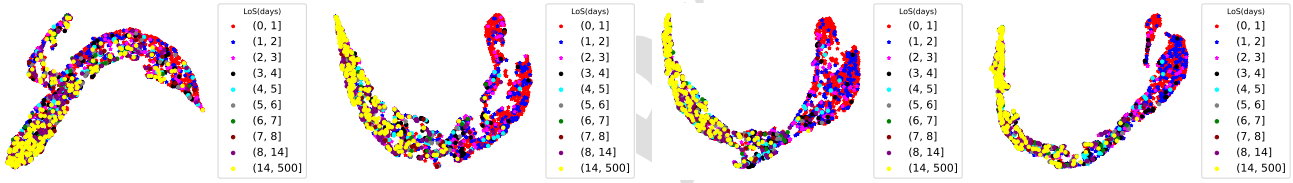


(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

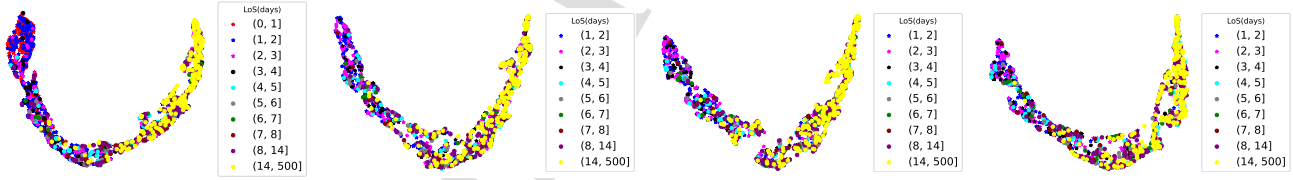


(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 14: UMAP 2D projections of LSTM model embeddings, labeled with LoS.



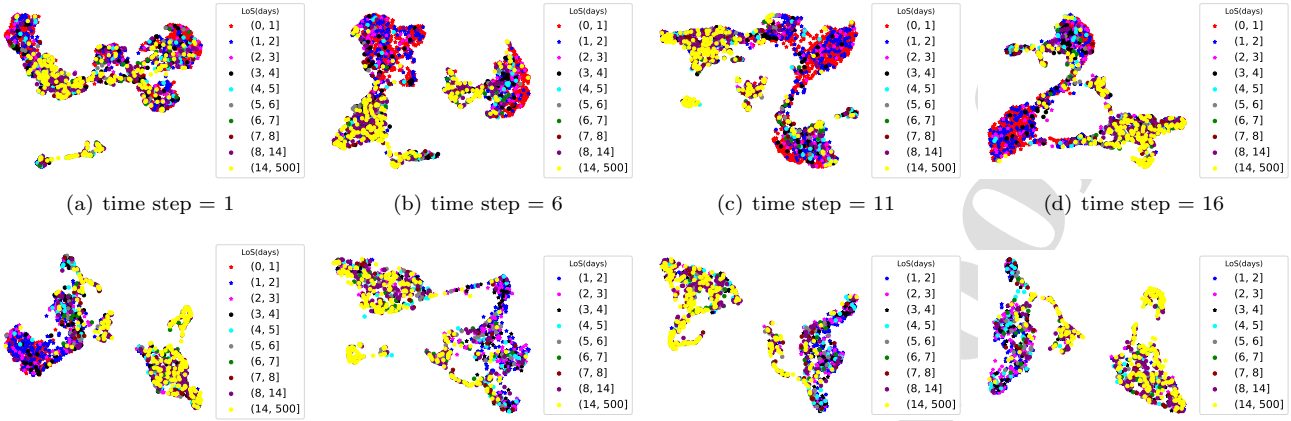
(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16



(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 15: UMAP 2D projections of GRU model embeddings, labeled with LoS.

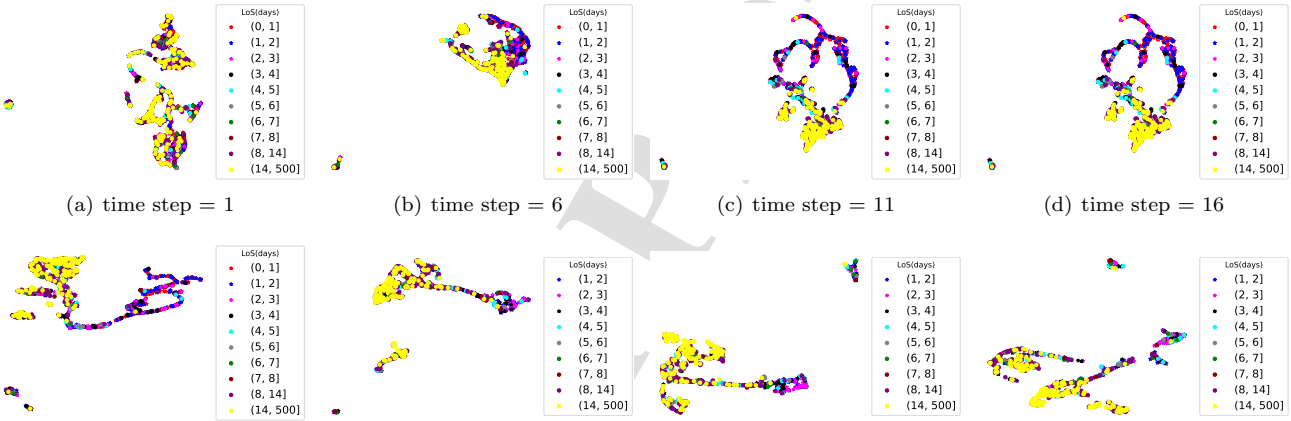
Journal Pre-proof



(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

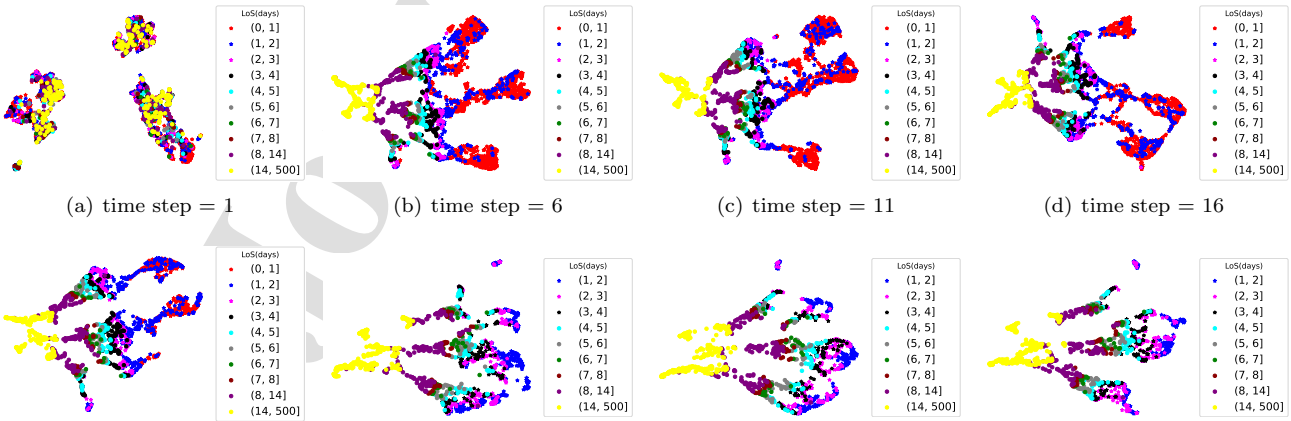
Figure 16: UMAP 2D projections of the Transformer model embeddings, labeled with LoS.



(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 17: UMAP 2D projections of TCN model embeddings, labeled with LoS.

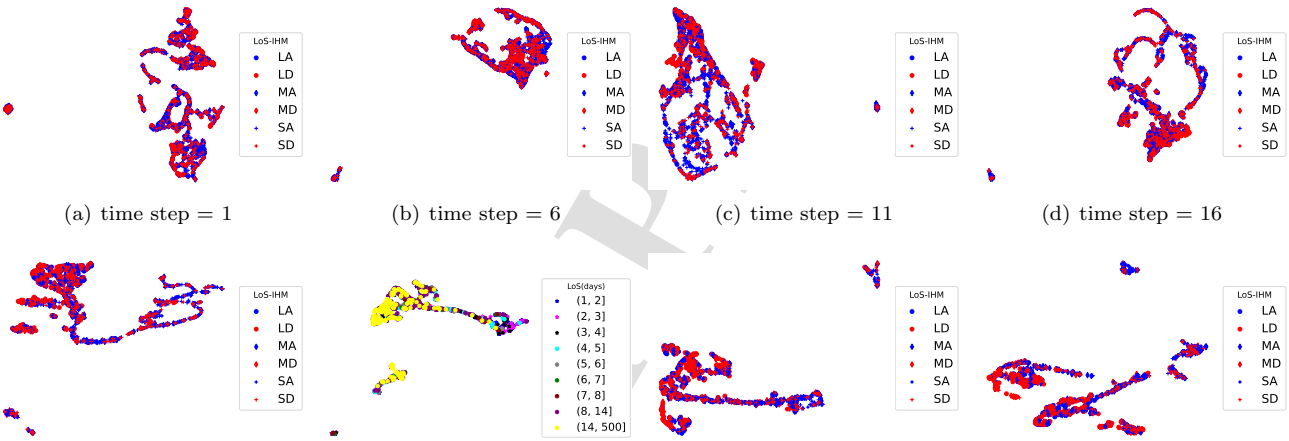


(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 18: UMAP 2D projections of TCN-ATT model embeddings, labeled with LoS.

Journal Pre-proof



(a) time step = 1 (b) time step = 6 (c) time step = 11 (d) time step = 16

(e) time step = 21 (f) time step = 26 (g) time step = 31 (h) time step = 36

Figure 19: UMAP 2D projections of TCN model embeddings, labeled with both LoS and in-hospital mortality.

Journal Pre-proof

Journal Pre-proof

References

[1] Aditya Achanta, Ask Nordestgaard, Napaporn Kongkaewpaisan, Kelsey Han, April Mendoza, Noelle Saillant, Martin Rosenthal, Peter Fagenholz, George Velmahos, and Haytham MA Kaafarani. Most of the variation in length of stay in emergency general surgery is not related to clinical factors of patient care. *Journal of Trauma and Acute Care Surgery*, 87(2):408–412, 2019.

[2] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[4] David Chardin, Cyprien Gille, Thierry Pourcher, Olivier Humbert, and Michel Barlaud. Learning a confidence score and the latent space of a new supervised autoencoder for diagnosis and prognosis in clinical metabolomic studies. *BMC bioinformatics*, 23(1):361, 2022.

[5] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):1–12, 2018.

[6] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[7] Elias Dabbas. *Interactive Dashboards and Data Apps with Plotly and Dash: Harness the power of a fully fledged frontend web framework in Python–no JavaScript required*. Packt Publishing Ltd, 2021.

[8] Amil Dravid, Florian Schiffers, Boqing Gong, and Aggelos K Katsaggelos. medxgan: Visual explanations for medical classifiers through a generative latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2936–2945, 2022.

[9] Junyi Gao, Cao Xiao, Lucas M Glass, and Jimeng Sun. Dr. agent: Clinical predictive model via mimicked second opinions. *Journal of the American Medical Informatics Association*, 27(7):1084–1091, 2020.

[10] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471, 2000.

[11] Gaspard Harerimana, Jong Wook Kim, and Beakcheol Jang. A deep attention model to forecast the length of stay and the in-hospital mortality right on admission from icd codes and demographic data. *Journal of Biomedical Informatics*, 118:103778, 2021.

[12] Hrayr Harutyunyan, Hrant Khachatrian, David C. Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific Data*, 6(1):96, 2019.

[13] Hrayr Harutyunyan, Hrant Khachatrian, David C Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific data*, 6(1):1–18, 2019.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[16] Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

[17] Swaraj Khadanga, Karan Aggarwal, Shafiq Joty, and Jaideep Srivastava. Using clinical notes with time series data for icu management. *arXiv preprint arXiv:1909.09702*, 2019.

[18] Ashish K Khanna, Sonia O Labeau, Kathryn McCartney, Stijn I Blot, Mieke Deschepper, DecubICUs study Team, European Society of Intensive Care Medicine, et al. International variation in length of stay in intensive care units and the impact of patient-to-nurse ratios. *Intensive and Critical Care Nursing*, page 103265, 2022.

[19] Ashwani Kumar and Hamideh Anjomshoa. A two-stage model to predict surgical patients' lengths of stay from an electronic patient database. *IEEE journal of biomedical and health informatics*, 23(2):848–856, 2018.

[20] Zachary C Lipton, David C Kale, Randall Wetzel, et al. Modeling missing data in clinical time series with rnns. *Machine Learning for Healthcare*, 56, 2016.

[21] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017.

[22] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[23] Lyse Naomi Wamba Momo, Nyalleng Moorosi, Elaine O Nsoesie, Frank Rademakers, and Bart De Moor. Length of stay prediction for hospital management using domain adaptation. *Engineering Applications of Artificial Intelligence*, 133:108088, 2024.

Journal Pre-proof

[24] Hassan Mumtaz, Muhammad K Ejaz, Muhammad Tayyab, Laiba I Vohra, Shova Sapkota, Mohammad Hasan, and Muhammad Saqib. Apache scoring as an indicator of mortality rate in icu patients: a cohort study. *Annals of Medicine and Surgery*, 85(3):416–421, 2023.

[25] Sanjay Purushotham, Chuizheng Meng, Zhengping Che, and Yan Liu. Benchmarking deep learning models on large healthcare datasets. *Journal of biomedical informatics*, 83:112–134, 2018.

[26] John Rapoport, Daniel Teres, Yonggang Zhao, and Stanley Lemeshow. Length of stay data as a guide to hospital economic performance for icu patients. *Medical care*, pages 386–397, 2003.

[27] Emma Rocheteau, Pietro Liò, and Stephanie Hyland. Temporal pointwise convolutional networks for length of stay prediction in the intensive care unit. In *Proceedings of the Conference on Health, Inference, and Learning*, pages 58–68, 2021.

[28] Emma Rocheteau, Catherine Tong, Petar Veličković, Nicholas Lane, and Pietro Liò. Predicting patient outcomes with graph representation learning. *arXiv preprint arXiv:2101.03940*, 2021.

[29] Segen's Medical Dictionary. length of stay, 2011.

[30] Shammamah Hossain. Visualization of Bioinformatics Data with Dash Bio. In Chris Calloway, David Lippa, Dillon Niederhut, and David Shupe, editors, *Proceedings of the 18th Python in Science Conference*, pages 126 – 133, 2019.

[31] Seyedmostafa Sheikhalishahi, Vevake Balaraman, and Venet Osmani. Benchmarking machine learning models on multi-centre eicu critical care dataset. *PloS one*, 15(7):e0235424, 2020.

[32] Kieran Stone, Reyer Zwiggelaar, Phil Jones, and Neil Mac Parthaláin. A systematic review of the prediction of hospital length of stay: Towards a unified framework. *PLOS Digital Health*, 1(4):e0000017, 2022.

[33] Lahn D Straney, Andrew A Udy, Aidan Burrell, Christoph Bergmeir, Sue Huckson, D James Cooper, and David V Pilcher. Modelling risk-adjusted variation in length of stay among australian and new zealand icus. *PloS one*, 12(5):e0176570, 2017.

[34] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[35] Reed T Sutton, David Pincock, Daniel C Baumgart, Daniel C Sadowski, Richard N Fedorak, and Karen I Kroeker. An overview of clinical decision support systems: benefits, risks, and strategies for success. *NPJ digital medicine*, 3(1):17, 2020.

[36] Patrick J Thoral, Mattia Fornasa, Daan P de Bruin, Michele Tonutti, Hidde Hovenkamp, Ronald H Driessen, Armand RJ Girbes, Mark Hoogendoorn, and Paul WG Elbers. Explainable machine learning on amsterdamumcdb for icu discharge decision support: uniting intensivists and data scientists. *Critical care explorations*, 3(9), 2021.

[37] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[39] Yanbo Xu, Siddharth Biswal, Shriprasad R Deshpande, Kevin O Maher, and Jimeng Sun. Raim: Recurrent attentive and intensive model of multimodal patient monitoring data. In *Proceedings of the 24th ACM SIGKDD international conference on Knowledge Discovery & Data Mining*, pages 2565–2573, 2018.

[40] Hugo Yèche, Rita Kuznetsova, Marc Zimmermann, Matthias Hüser, Xinrui Lyu, Martin Faltys, and Gunnar Rätsch. Hirid-icu-benchmark–a comprehensive machine learning benchmark on high-resolution icu data. *arXiv preprint arXiv:2111.08536*, 2021.

[41] Dongdong Zhang, Changchang Yin, Jucheng Zeng, Xiaohui Yuan, and Ping Zhang. Combining structured and unstructured data for predictive models: a deep learning approach. *BMC medical informatics and decision making*, 20(1):1–11, 2020.

Journal Pre-proof

Journal Pre-proof

2025-03-20

Paper: Understanding Deep Learning Models for Length of Stay Prediction on critically ill patients through Latent Space Visualization

Ethics Statement

This study utilized anonymized, publicly available data that did not include any personally identifiable information. As such, the data were exempt from institutional review board (IRB) or comparable ethical oversight requirements. All data-handling procedures were in accordance with relevant regulations regarding privacy and data protection, and no additional ethical approvals were necessary for conducting the research.

Journal Pre-proof

Journal Pre-proof

HIGHLIGHTS

- We predict hourly remaining Length of Stay (LoS) for critically ill patients admitted to the Amsterdam University Medical Center. The Amsterdam University Medical Center (AUMC) is the first freely available European critical care database and in contrast to the existing US-based ICU databases where patients' stays are typically shorter, here we experience longer stays and a higher variability in the LoS of almost 10 days.
- We compare the predictive performance of sequence-to-sequence models to that of static machine learning algorithms and show the superiority of the former models.
- We develop a dashboard to provide an intuitive understanding of the model inner workings by interactively exploring the models' latent spaces over time.
- The latent space exploration showed the models' capabilities of clustering patients over time with respect to their true LoS but also revealed neighborhoods of patients at risk of experiencing in-hospital mortality with only 5 hours of post ICU admission data.
- We show how such interactive dashboards can be integrated into clinical decision support systems to gain valuable insights towards capacity planning.

Journal Pre-proof

Journal Pre-proof

Declaration of interests

☐ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☒ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Bart De Moor reports financial support was provided by European Research Council. Bart De Moor reports financial support was provided by Flanders AI Research Program. Bart De Moor reports financial support was provided by Flemish Government Agencies FWO. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Journal Pre-proof