Project Draft for CS598-DLH Spring 2024

Temporal Point wise Convolutional Networks for Length of Stay Prediction in the Intensive Care Unit (ICU)

Priyank Jain

Vibhor Jain

https://youtu.be/mGx-YIrTtj8

https://github.com/vibhor-github/length-of-stay

ABSTRACT

The duration of a patient's inpatient stay is crucial not only for their outcome but also for the effective planning and management of hospital resources. It directly influences readmission rates as well. Accurately predicting the length of stay can positively impact mortality rates and aid in efficient resource allocation and management, thus enhancing patient satisfaction by minimizing unnecessary readmissions. Active research has explored various traditional machine learning methods for estimating the length of stay early on. However, deep learning techniques have demonstrated superior performance in healthcare research compared to traditional approaches. In our study, we propose employing a deep learning-based approach, specifically a CNN model, to enhance the prediction of patient length of stay using routinely collected inpatient data.

1. RELATED WORK

Numerous significant research endeavors have been undertaken to forecast the length of stay in healthcare settings. Early investigations, such as that by David H. Gustafson [8], utilized Bayesian regression techniques to demonstrate the feasibility of cost-effective length-of-stay prediction. Suresh et al. [32] leveraged neural network concepts like backpropagation to discern patterns in patient data for predictive purposes. Similarly, Clark et al. [5] applied Poisson regression methodology to estimate lengths of stay.

Recent years have seen a surge in interest in employing deep learning methodologies for length-of-stay prediction. Gentimis et al. [11], for instance, employed a basic neural network on the MIMIC-III [19] dataset, illustrating the development of a generalizable model with high predictive accuracy across diverse health conditions. Rocheteau et al. [29] utilized temporal convolutional neural networks (CNNs) to capture temporal trends and inter-feature relations for length-of-stay prediction, achieving notable improvements over baseline models like standard LSTM, channel-wise LSTM, and transformer models. Clinical notes, being rich repositories of patient information, have emerged as valuable data sources for prediction models. Huang et al. [16] devised ClinicalBERT, based on the BERT [7] model, to extract insights from clinical notes and enhance predictions of hospital readmission.

Moreover, studies like that by Weissman et al. [33] have underscored the benefits of incorporating clinical notes in predictive models, showing enhancements in both length-of-stay and mortality predictions. Mullenbach et al. [24] contributed an approach for extracting ICD codes from clinical text, further enriching predictive modeling in healthcare contexts.

2. INTRODUCTION

Considerable efforts have been dedicated to predicting length of stay using various statistical machine learning and deep learning techniques. Additionally, researchers have increasingly recognized the value of clinical notes as a crucial source of patient health information. While previous studies have explored predicting length of stay using physiological data, none, to our knowledge, have investigated the potential of integrating both physiological data and clinical text for this purpose.

We introduce a neural architecture, NeuralLOS, designed to harness information from both clinical notes and physiological data. In the initial layers, each data source undergoes separate processing using distinct architectures to extract hidden states. These generated states are then merged to form a unified data source, which subsequently passes through additional layers to predict the remaining length of stay.

For the physiological data, we adopt a sliding window approach to capture temporal diagnostic information using a CNN model. Conversely, for the clinical notes, we generate embeddings using GloVe and apply either CNN or RNN architecture for further processing.

3. METHOD

3.1 Data

The data utilized in this project is sourced from the Mimic-III critical care database [19; 27], encompassing de-identified health-related information such as demographics, vital signs, laboratory results, procedures, medications, caregiver notes, and more. Structured as a relational database, Mimic-III offers versatility for various applications, including the focus of this project: length of stay prediction.

	Train	Test
# Patients	28,620	5,058
# ICU stays	35,621	6,281
Total samples	2,925,434	525,912

Table 1: Benchmark data for length of stay data set

Pre-processing of the data relies on a standard benchmark [12] code, albeit requiring significant modifications to ensure performance and inclusion of clinical notes, which were absent in the original benchmark models.

The benchmark generates cleansed tabular features and label values for predicting risk assessment (mortality prediction), physiologic compensation, phenotype, and length of stay. Additionally, it provides code for generating prediction baseline benchmarks for result comparison. Table 1 presents statistics regarding length of stay, including the distribution between training and test datasets.

To enrich the features generated by the benchmark, clinical notes for each visit are appended. These notes are then converted into embeddings for model input. Length of stay prediction labels are assigned for each period length of the episode, with the label decreasing as the period length increases. CSV files are also generated for each patient/episode, comprising approximately hourly physiological data points such as Glasgow Coma Scale measures, glucose levels, oxygen saturation, blood pressure, heart rate, and temperature. Currently, the data lacks temporal organization, necessitating further investigation to potentially consolidate data into hourly intervals to align with our intended CNN architecture, which requires fixed-length periods.

Since the benchmark dataset lacks clinical notes, constructing a clinical note dataset is imperative, possibly by grouping notes into fixed-length time windows. Determining the optimal time window necessitates further investigation, particularly considering that the Mimic-III dataset generally exhibits shorter stay lengths compared to the original paper utilizing clinical notes for health outcome prediction. Consequently, an 8-hour window may be excessively lengthy.

3.2 Pre-processing

The Mimic-III database stands as a cornerstone in healthcare research, widely embraced by researchers. To kickstart our analysis, we utilize the benchmark [13] to preprocess the raw Mimic-III data. This benchmark preprocessing yields tabular physiological data alongside true values indicating the remaining length of stay, complemented by additional patient information spanning the entire inpatient duration. This data is structured as a time-series, capturing all available observations over time for each patient during each episode of admission.

However, the original benchmark code does not handle clinical notes. Thus, we've enhanced the benchmark code to incorporate the retrieval of clinical notes corresponding to the physiological data time-series. These notes are segmented into sliding windows, and embeddings are generated from the raw notes, serving as inputs to our model.

We establish similar sliding windows for the preprocessed physiological data to encapsulate temporal information. Employing a 5-hour window size for both physiological data and note embeddings, we ensure a comprehensive representation of patient data. For instance, if a patient's stay spans 7 hours, the generated windows would be [0,1,2,3,4], [1,2,3,4,5], [2,3,4,5,6], [3,4,5,6,7]. Each number denotes the hour from admission, aggregating information across all features for every hour within the window.

3.3 Winsorization

In our preliminary data analysis, we noted the presence of several extreme values within the dataset, which skewed the overall representation of the data. To address this issue, we implemented a winsorization technique, setting the threshold at 94% to mitigate the impact of outliers during the training process. Winsorization, a widely used statistical method for outlier treatment, involves capping extreme values to reduce their influence on the analysis. With a winsorization of 94%, we effectively trim 3% of extreme data from both ends of the datasets, ensuring a more balanced and reliable dataset for subsequent analysis.

3.4 Benchmark Models

Based on the original benchmark [13], we are conducting comparisons with a linear regression model and a simple LSTM model. Unlike the original paper, which employed an ordered classification approach for predicting the number of days remaining, we opt for simple regression, as well as a custom metric to better reflect real-world length-of-stay (LOS) usage patterns. This custom metric divides the LOS range into ten buckets, including extremely short visits (less than one day), seven day-long buckets for each day of the first week, and two "outlier" buckets for stays over one and two weeks, respectively. By transforming the regression problem into an ordinal multiclass classification problem, we use a Kappa score to measure this classification, as it accommodates ordered classes and their correlations.

We incorporate the standard LSTM model trained with simple regression to provide a more direct comparison with the linear regression baseline, which only trains against the raw LOS value. Common metrics for regression tasks, such as Mean Squared Error or Mean Absolute Difference, are typically employed for model comparison. However, the original benchmark did not use Mean Squared Error.

Furthermore, we enhance the benchmark preprocessing by integrating clinical notes, an aspect overlooked in the original benchmark. Despite leveraging code from the benchmark for this project, significant modifications were necessary to ensure compatibility with the latest versions of Keras and TensorFlow. Additionally, multiprocessing support was implemented to enhance the efficiency of the LOS task, as creating tensors for training was previously bottlenecked.

3.5 Clinical notes

Numerous studies and research endeavors [16; 7; 18] have highlighted the wealth of valuable patient information contained within clinical notes, demonstrating their efficacy in deep learning models. In our investigation, we assessed BioClinicalBERT [1], a freely accessible BERT model derived from BioBERT [23] and fine-tuned with MIMIC-III clinical notes, alongside BioSentVec [4], which leverages PubMed [3] to generate embeddings from MIMIC-III clinical notes. Both methods yield embeddings of similar shapes.

Layer#	Layer Name	#Input Params	#Output Param
1	Conv2d	5,440	87,040
2	Conv2d	87,040	174,080
2	MaxPool2d	174,080	34,816
3	Conv2d	34,816	69,632
4	Linear	69,632	8,192
5	Dropout	8,192	8,192
6	Linear	8,192	4,096
7	Linear	4,096	1,024

Table 2: PhysioNet: Layerwise Parameters

Layer#	Layer Name	#Input Params	#Output Param
1	Conv2d	1,966,080	7,864,320
2	MaxPool2d	7,864,320	1,966,080
2	Conv2d	1,966,080	3,932,160
3	Conv2d	3,932,160	1,966,080
4	Linear	1,966,080	65,536
5	Dropout	65,536	65,536
6	Linear	65,536	16,384
7	Dropout	16,384	16,384
8	Linear	16,384	4,096
9	Linear	4,096	1,024

Table 3: NotesNet: Layerwise Parameters

These embeddings serve as input to our NotesNet [see Figure 2], enabling the extraction of hidden states from the embeddings.

3.6 LOS Design

The duration of an inpatient stay hinges on various factors, primarily the physiological data collected from the patient, such as blood pressure and temperature. However, these features can fluctuate throughout the duration of the stay. Healthcare centers typically monitor and record these features at regular intervals until the patient is discharged or deceased, introducing a temporal aspect to the dataset where the remaining length of stay is influenced by changes in the patient's condition over time.

Given the sequential nature of such datasets, opting for a recurrent neural network (RNN) model like LSTM [15] appears logical to capture the evolving information. Additionally, since MIMIC III [27] is a sizable dataset, processing as much information as possible for model training seems sensible. Neural network models generally perform better with larger datasets. However, training an RNN model on such a large dataset demands significant hardware resources.

To address these challenges, we propose a CNN [22]-based neural network architecture called NeuralLOS. We employ a sliding window approach to capture temporal information from the dataset. Each window comprises observations from the preceding few hours, which are then batched and shuffled to create the training dataset. The remaining length of stay at the end of the window serves as the true output. In our initial experiments, we utilize a 4-hour window. For example, if an inpatient admission record spans 6 hours, the windows would be [1,2,3,4], [2,3,4,5], and [3,4,5,6].

Regarding model design, we explored the possibility of using pre-trained CNN models like AlexNet [21] or ResNet [14] as our base model. However, many of these pre-trained models are optimized for image data, which may not suit our use case. Consequently, we opt for a simple 9-layer neural network for implementation.

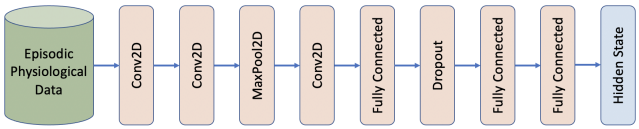

Figure 1: PhysioNet implementation

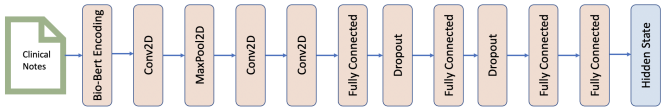

Figure 2: NotesNet implementation

The architecture of our model, depicted in figures [1, 2, 3], comprises multiple levels. Our top-level model, EpisodeNet, is a fusion of two distinct models: PhysioNet and NotesNet.

3.6.1 PhysioNet

Figure 1 illustrates the architecture of PhysioNet, which comprises three convolution layers, one pooling layer, one dropout layer, and three fully connected linear layers. Detailed parameters of the model are provided in Table 2. This model is utilized to process the tabular physiological data for each batch. The output of the model is a tensor with dimensions (#batch size, 32), representing the hidden learned state of the model derived from the tabular physiological data.

3.6.2 NotesNet

Figure 2 depicts the architecture of NotesNet, which comprises three convolution layers, one pooling layer, two dropout layers, and four fully connected linear layers. We utilize BioClinicalBERT [1] to generate embeddings from the batch's notes. These embeddings serve as input to NotesNet, enabling the processing and learning of information for a given batch. The model's input is a tensor with dimensions (#batch size, #sentences, #embeddings), with #sentences set to 80 and #embeddings set to 768 for our model. The output of the model is a tensor with dimensions (#batch size, 32), representing the hidden learned state of the model derived from the notes data.

3.6.3 EpisodeNet

EpisodeNet serves as the top-level model, integrating both PhysioNet and NotesNet to process each batch of data. The architecture is depicted in Figure 3. The tabular physiological data from each batch undergoes processing via PhysioNet, while the corresponding notes embedding is processed through NotesNet. Subsequently, the output hidden states from both PhysioNet and NotesNet are concatenated and forwarded through a sequence of fully connected linear layers to predict the remaining length of stay as a regression output.

3.7 Metrics

We selected 3 different metrics commonly used with regres-sion models for our evaluations.

3.7.1 Mean Squared Error

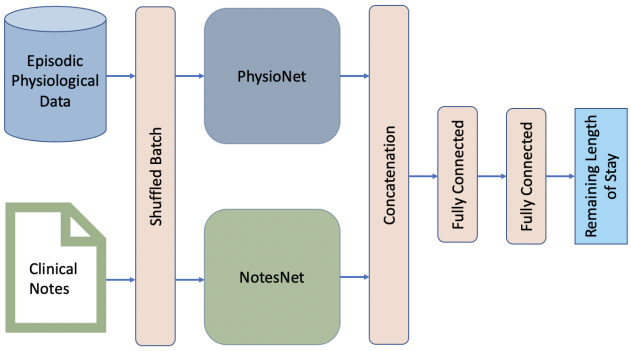

Figure 3: EpisodeNet implementation

MSE measures the average of the square of the errors. It is calculated by taking an average of the square of the differ-ence between the true values and the predicted values. It is sensitive to outliers and has higher penalty with greater deviation on true and predicted values. It is given as:

$$\frac{\sum(y_{true} - y_{pred})^2}{\#sample}$$

3.7.2 *Mean Absolute Error*

MAE measures the average of the absolute difference of the errors. It is calculated by taking an average of absolute value of the difference between the true values and the predicted values. It is less sensitive to outliers. It is given as:

$$\frac{\sum|(y_{true} - y_{pred})|}{\#sample}$$

3.7.3 *Mean Absolute Percentage Error*

MAPE measures the average of the ratio of the absolute difference of the errors to the true value. It gives the nor-malized version of the MAE by true values. It is given as:

$$\frac{100}{\#sample} * \sum|\frac{(y_{true} - y_{pred})}{y_{true}}|$$

3.8 Hyper-parameter tuning and selection

In the realm of deep learning models, hyperparameter tuning stands as a pivotal step in identifying the most effective parameters for optimal model performance. In our approach, we conducted experiments involving various combinations of different hyperparameters to ascertain the optimal settings for achieving the best results.

3.8.1 *Number of epochs*

We conducted experiments involving different numbers of epochs combined with various hyperparameters such as learning rate and batch size across multiple iterations. Throughout these experiments, we monitored the training loss and calculated metrics on the validation set for each epoch.

Our observations revealed that the training loss decreases rapidly and substantially during the initial few epochs, eventually flattening out at around 5 epochs for all iterations, as depicted in Figure 4. Additionally, while the mean squared error (MSE) for validation remains relatively stable up to 5 epochs, it begins to increase thereafter, as illustrated in Figure 5. This trend suggests potential overfitting of the model to the training set.

Based on these findings, we selected the number of epochs for our evaluation, considering the balance between model performance and avoidance of overfitting.

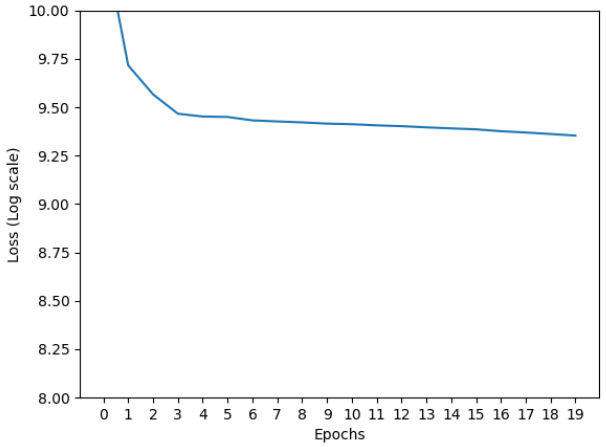

Figure 4: Trend of training losses over epochs

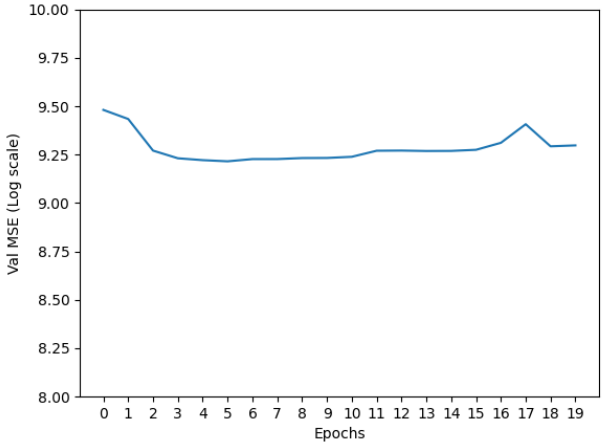

Figure 5: Trend of validation mean squared errors over epochs

3.8.2 *Batch size; Learning rate*

Similarly, we experimented with different batch sizes with a combination of different hyper-parameters. We selected 32, 128 and 256 as candidate batch sizes and ran multiple iter-ations with different learning rates. We collected different metrics with all the combinations as shown in tables [4, 5, 6].
Based on these metrics, the model performed best on test dataset when trained with batch size of 32 and learning rate of 0.0001.

3.8.3 *Optimizer and Loss function*

Since we have designed our model to predict a regression output, we experimented with two different loss functions: 1) L1Loss 2) MSELoss. Based on the test results over mul-tiple iterations, we observed that MSELoss is better suited to our model.
We explored Stochastic gradient descent and Adam[20] optimizers for our model. Based on the experimental results,

lr/batch size	32	128	256
0.01	6.821e+12	1.113e+12	3.555e+11
0.001	9.858e+4	1.009e+5	5.363e+4
0.0001	**1.744e+4**	9.187e+4	7.652e+4
0.00001	1.472e+5	1.427e+5	2.498e+5

Table 4: A comparison of mean square error values on test set for different batch sizes and learning rates

lr/batch size	32	128	256
0.01	3.099e+5	4.809e+4	1.601e+4
0.001	8.239e+1	8.474e+1	8.280e+1
0.0001	**8.070e+1**	8.297e+1	8.327e+1
0.00001	9.132e+1	1.001e+2	9.739e+1

Table 5: A comparison of mean absolute error values on test set for different batch sizes and learning rates

we selected Adam as the optimizer for our model.

4. RESULTS

4.1 Evaluation

The initial results of our benchmark models closely align with the findings reported in the benchmark study [12], with our results showing slight improvement. For instance, the original paper reported a mean absolute error (MAE) of 94.7 for a basic LSTM model, whereas our model achieved an MAE of 79.3. Notably, we conducted training using regression output rather than employing custom bins for classification, which was used in the original study. Our attempts at training with custom bins (e.g., 1 day, 2 days, 3 days, etc., up to 2 weeks) did not yield satisfactory results. Please refer to Table 7 for a comprehensive listing of results.

When evaluating a forecasting model, it is essential to understand two key aspects: a) the amount of historical data required to make accurate predictions and b) how closely the model's predictions align with the current state [28]. To assess the model's performance at different intervals, we initially applied Winsorization of 96% across the length of stay for each episode, removing the bottom 3% and top 97% of data points. Additionally, we filtered out episodes lasting less than 60 hours to ensure consistent comparison across different time periods. The choice of 60 hours was made because it is close to one standard deviation of the average length of stay (66 hours). Figure 6 illustrates the distribution of episodes over the length of stay after applying Winsorization.

Considering the test set comprises 1,555 episodes, we observe a consistent number of data points at each period up to 60 hours, as depicted in Figure 7. By plotting the mean squared error at these different time periods, we gain insights into how the models perform as they access more information over time.

lr/batch size	32	128	256
0.01	1.008e+4	1.528e+3	3.337e+3
0.001	**1.636**	1.845	1.765
0.0001	1.662	1.706	1.737
0.00001	2.349	2.839	2.596

Table 6: A comparison of mean absolute percentage error values on test set for different batch sizes and learning rates

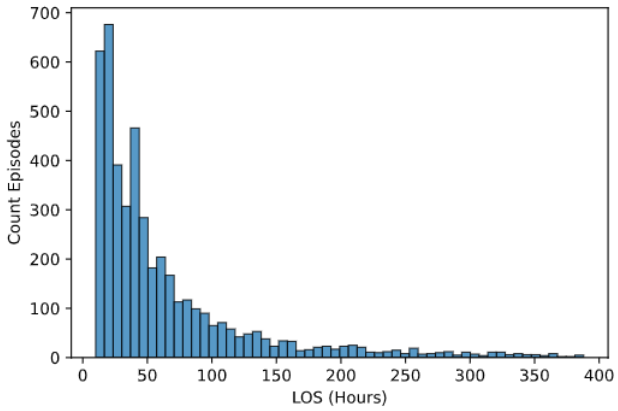

Figure 6: Histogram showing episodes over length of stay after Winsorization

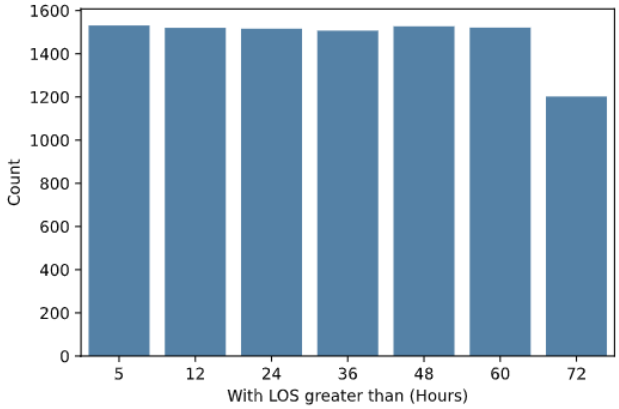

Figure 7: Distribution of Data Points Across Different Lengths of Stay

The results are presented in Figure 8.

As anticipated, the error decreases over progressive time periods for all models, except for linear regression, which decreases until 50 hours but then rises again. Other models also exhibit an inflection point at 60 hours, except for NeuralLOS with full data and LSTM.

When comparing NeuralLOS using only physiological (tabular) data with a model augmented with Bio-ClinicalNote embeddings [16], we included a version of NeuralLOS trained on the same dataset as the model with notes. Note processing consumes a significant amount of time, preventing training on the full dataset. To facilitate comparison, we trained NeuralLOS on the same smaller dataset to discern differences. The model with notes appears to perform better than the tabular-only model overall, but not when considering stays longer than 60 days.

We also investigated whether predictions become more accurate as the patient approaches the end of their stay. Using the same episodes, we categorized bins from 2 weeks (336 hours) to 12 hours. The number of data points in each bin is illustrated in Figure 9.

Model	Data types	MAE	MSE	MAPE
Linear regression	Tabular	121.69	12,805,595	3.15
LSTM	Tabular	**79.28**	16,889	81.10
LSTM	Notes	101.44	28,201	**0.72**
PhysioNet (full data)	Tabular	78.55	17,492	1.01
PhysioNet (Part data)	Tabular	80.50	17,450	1.66
PhysioNet+Notes (Part data)	Tabular+Notes	80.49	**16,122**	1.55

Table 7: Model results for length of stay prediction.

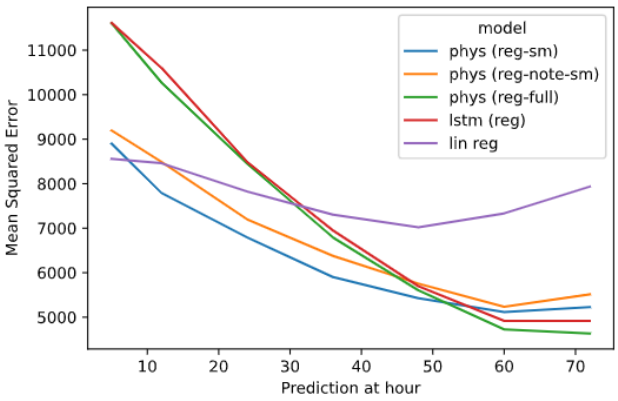

Figure 8: Mean Squared Error at different hours of stay

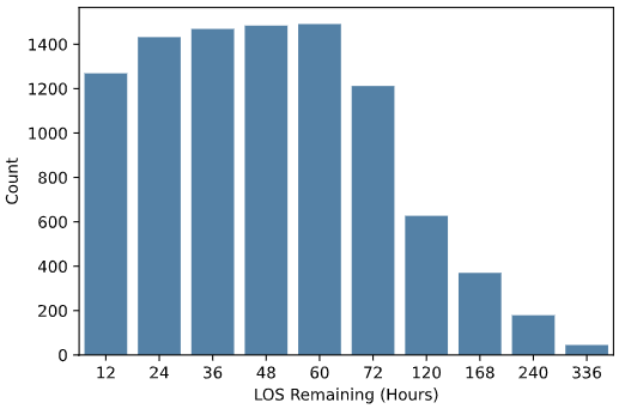

Figure 9: Number of data points at different remaining length of stay

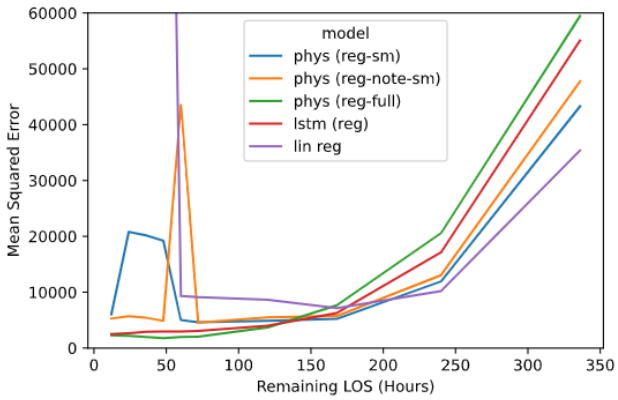

Figure 10: Mean Squared Error at different remaining lengths of stay

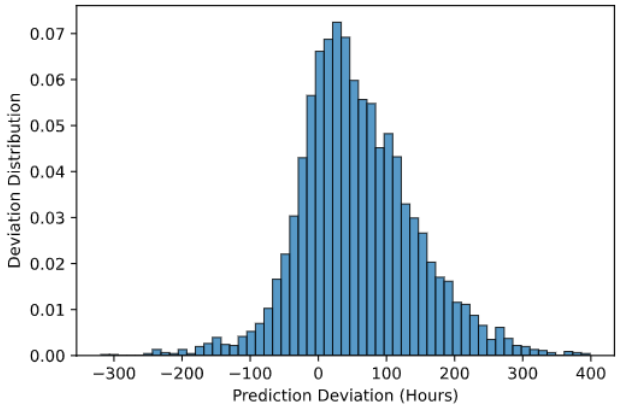

Figure 11: Linear regression deviation distribution

As anticipated, all models exhibit improvement as they approach the final stay. Interestingly, linear regression outperforms the other models initially, showing an inflection point at 168 hours before exceeding the chart limit at 60 hours. On the other hand, NeuralLOS and LSTM models start with lower effectiveness but show improvement around 120 hours.

While mean squared error (MSE) plots enable model comparison, visualizing the spread and degree of model accuracy is enhanced by plotting a histogram of deviation in hours. Figures 11 through 14 portray this distribution across the models.

The accuracy at each remaining length of stay is illustrated through a series of box plots in Figures 15 through 18.

4.2 Infrastructure

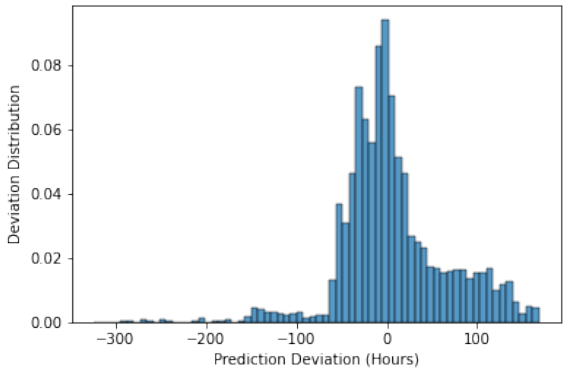

Figure 12: LSTM deviation distribution

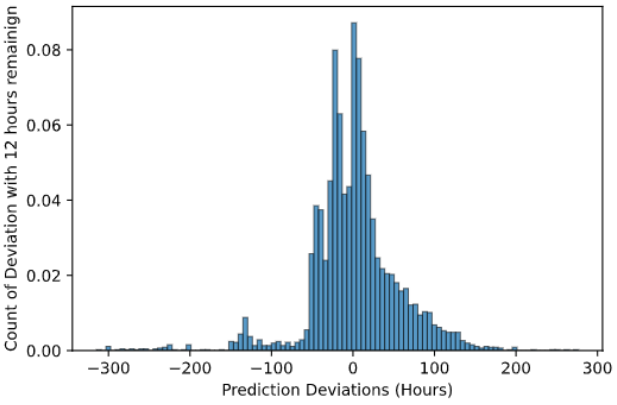

Figure 13: NeuralLOS deviation distribution

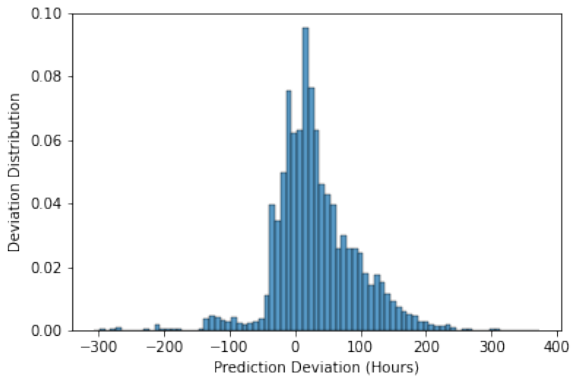

Figure 14: NeuralLOS with notes deviation distribution

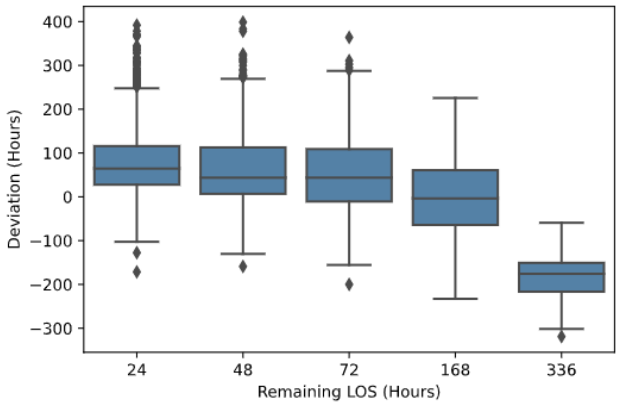

Figure 15: NeuralLOS with notes deviation distribution

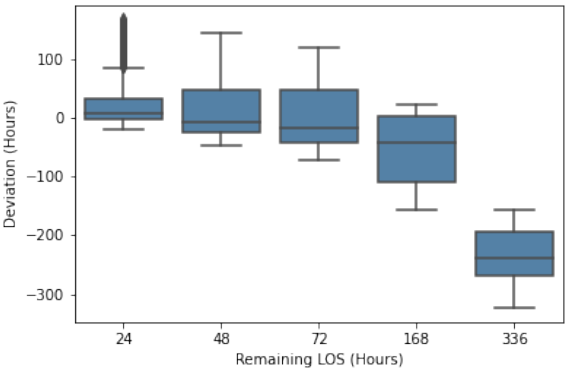

Figure 16: LSTM deviation distribution

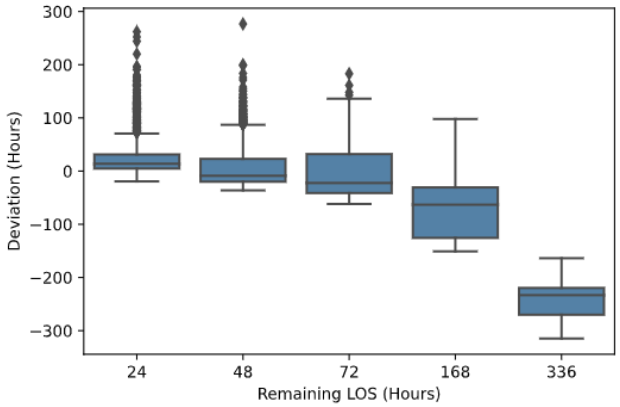

Figure 17: NeuralLOS deviation distribution

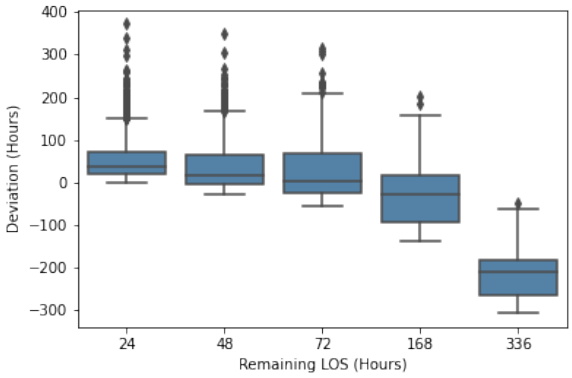

Figure 18: NeuralLOS with notes deviation distribution

We trained and evaluated our models on AWS Cloud Plat-form. The machine configuration is listed below:

Machine Type	Deep Learning OSS Nvidia
Platform	AWS
Memory	32 GB
GPU	Nvidia GPU TensorFlow 2.15
Storage	400 GB SSD

We utilize widely-used Python libraries, including but not limited to PyTorch, Keras, TensorFlow, scikit-learn, Matplotlib, and pickle. Our code is accessible through a GitHub repository. It's important to note that the benchmark code is constrained by data preprocessing-intensive tasks. Initially, the benchmark code lacked the capability to run in parallel and utilize GPU resources effectively. Consequently, we dedicated significant effort to implement multi-threaded data preprocessing, aiming to maximize GPU utilization.

5. TEAM CONTRIBUTIONS

It was a collaborative effort, both of us involved in various aspects, contributing to the planning, experimentation, and training phases of model development.

Vibhor spearheaded the setup of AWS environments for training LSTM and linear regression models. He also led the efforts to adapt and upgrade the benchmark code to ensure compatibility with newer versions of libraries like TensorFlow and Keras. Addressing speed challenges, Alan implemented multiprocessing capabilities in the preprocessing routines used for creating training tensors. Additionally, Alan authored the Data and Evaluation sections of the report and developed the program responsible for aggregating results from all models.

Priyank played a key role in designing the NeuralLOS model architecture and implementing the dataset windowing techniques. He actively participated in model training and metric generation.

We both significantly worked on the generation of BioSentVec and BioClinicalBERT embeddings for notes. They also played a crucial role in generating preprocessed data using the benchmark code.

6. CONCLUSIONS

Forecasting the length of a patient's stay is a critical challenge in healthcare. Obtaining an estimate of the remaining length of stay aids hospitals in better resource allocation for healthcare services. Additionally, it provides valuable insights for insurance companies to estimate expenses accurately. Leveraging NeuralLOS, we achieved impressive results in predicting length of stay. By comparing our model with various benchmark models and presenting results from different perspectives, we demonstrated its effectiveness. Although further refinement is required to enhance the model's performance, even in its current implementation, NeuralLOS yields superior results.

7. LIMITATIONS

One of the primary challenges we encountered was the scarcity of computational resources required to process the entire dataset. The embeddings of notes consume significant memory, and due to memory constraints, we were unable to accommodate the entire working set in memory. Additionally, since NeuralLOS involves computing a large number of parameters, utilizing GPUs was imperative to expedite training. Despite encountering some hurdles, we managed to secure access to a GPU in GCP with limited capacity. Consequently, we trained our EpisodeNet on a subset of the data. An intriguing observation we made was that the prediction accuracy of NeuralLOS improves for patients with longer stays. This improvement can be attributed to the accumulation of more information over time, enabling the model to make more accurate predictions.

8. RESOURCES

Github : https://github.com/vibhor-github/lenght-of-stay.git

9. ACKNOWLEDGEMENTS

We express our gratitude to Professor Jimeng Sun and all teaching assistants for their invaluable guidance and support throughout this work

10. REFERENCES

[1] E. Alsentzer, J. R. Murphy, W. Boag, W.-H. Weng, D. Jin, T. Naumann, and M. B. A. McDermott. Pub-licly available clinical bert embeddings, 2019.

[2] H. Baek, M. Cho, S. Kim, H. Hwang, M. Song, and S. Yoo. Analysis of length of hospital stay using electronic health records: A statistical and data mining approach. PloS one, 13(4):e0195901, 2018.

[3] K. Canese and S. Weis. Pubmed: the bibliographic database. In The NCBI Handbook [Internet]. 2nd edition. National Center for Biotechnology Information (US), 2013.

[4] Q. Chen, Y. Peng, and Z. Lu. Biosentvec: creating sentence embeddings for biomedical texts. 2019 IEEE International Conference on Healthcare Informatics (ICHI), Jun 2019.

[5] D. E. Clark and L. M. Ryan. Concurrent prediction of hospital mortality and length of stay from risk factors on admission. Health services research, 37(3):631–645, 2002.

[6] S. Cropley. The relationship-based care model: evaluation of the impact on patient satisfaction, length of stay, and readmission rates. JONA: The Journal of Nursing Administration, 42(6):333–339, 2012.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[8] G. DH. Length of stay: Prediction and explanation. Health services research, 3(1), 12–34., 1968.

[9] J. Fang, J. Zhu, and X. Zhang. Prediction of length of stay on the intensive care unit based on bayesian neu-ral network. In Journal of Physics: Conference Series, volume 1631, page 012089. IOP Publishing, 2020.

[10] R. Figueroa, J. Harman, and J. Engberg. Use of claims data to examine the impact of length of inpatient psychiatric stay on readmission rate. Psychiatric Services, 55(5):560–565, 2004.

[11] T. Gentimis, A. J. Alnaser, A. Durante, K. Cook, and R. Steele. Predicting hospital length of stay using neural networks on mimic iii data. In 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech), pages 1194–1201, 2017.

[12] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan. Multitask learning and benchmarking with clinical time series data. Scientific Data, 6(1), Jun 2019.

[13] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan. Multitask learning and benchmarking with clinical time series data. Scientific Data, 6(1):96, 2019.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

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

[16] K. Huang, J. Altosaar, and R. Ranganath. Clinicalbert: Modeling clinical notes and predicting hospital readmission. arXiv preprint arXiv:1904.05342, 2019.

[17] S.-J. Jang, I. Yeo, D. N. Feldman, J. W. Cheung, R. M. Minutello, H. S. Singh, G. Bergman, S. C. Wong, and L. K. Kim. Associations between hospital length of stay, 30-day readmission, and costs in st-segment–elevation myocardial infarction after primary percutaneous coro-nary intervention: a nationwide readmissions database analysis. Journal of the American Heart Association, 9(11):e015503, 2020.

[18] B. L. S. G. C. P.-P. J. M. A. V. M. M. Jienan Yao, Yuyang Liu and M. Ghassemi. Visualization of deep models on nursing notes and physiological data for predicting health outcomes through temporal sliding windows. In Explainable AI in Healthcare and Medicine, pages 115–129, 2021.

[19] A. E. Johnson, T. J. Pollard, L. Shen, L. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible crit-ical care database. Scientific data, 3:160035, 2016.

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. Advances in neural information processing sys-tems, 25:1097–1105, 2012.

[22] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio. Object recognition with gradient-based learning. In Shape, con-tour and grouping in computer vision, pages 319–345. Springer, 1999.

[23] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang. Biobert: a pre-trained biomedical lan-guage representation model for biomedical text mining. Bioinformatics, Sep 2019.

[24] J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein. Explainable prediction of medical codes from clinical text. arXiv preprint arXiv:1802.05695, 2018.

[25] K. J. Ottenbacher, P. M. Smith, S. B. Illig, R. T. Linn, G. V. Ostir, and C. V. Granger. Trends in length of stay, living setting, functional outcome, and mortality following medical rehabilitation. Jama, 292(14):1687–1695, 2004.

[26] A. Peimankar and S. Puthusserypady. Dens-ecg: A deep learning approach for ecg signal delineation. Ex-pert Systems with Applications, 165:113911, 2021.

[27] A. E. Pollard, Tom J abd Johnson. The mimic-iii clinical database. http://dx.doi.org/10.13026/C2XW26, 2016.

[28] A. e. a. Rajkomar. Scalable and accurate deep learning with electronic health records. 6:18, 2018.

[29] E. Rocheteau, P. Liò, and S. Hyland. Temporal pointwise convolutional networks for length of stay prediction in the intensive care unit. arXiv preprint arXiv:2007.09483, 2020.

[30] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. IEEE Transactions on Signal Process-ing, 45(11):2673–2681, 1997.

[31] M. Sotoodeh and J. C. Ho. Improving length of stay prediction using a hidden markov model. AMIA Sum-mits on Translational Science Proceedings, 2019:425, 2019.

[32] A. Suresh, K. Harish, and N. Radhika. Particle swarm optimization over back propagation neural network for length of stay prediction. Procedia Computer Science, 46:268–275, 2015. Proceedings of the International Conference on Information and Communication Technologies, ICICT 2014, 3-5 December 2014 at Bolgatty Palace Island Resort, Kochi, India.

[33] G. E. Weissman, R. A. Hubbard, L. H. Ungar, M. O. Harhay, C. S. Greene, B. E. Himes, and S. D. Halpern. Inclusion of unstructured clinical text improves early prediction of death or prolonged icu stay. *Critical care medicine*, 46(7):1125, 2018.