CSE6250: Big Data Analytics in Healthcare
Team 45: Final Paper
https://youtu.be/odGomiBgvzs

Sai Supreeth Manyam
smanyam3@gatech.edu

Sang M Park
spark930@gatech.edu

***Abstract*—Clincial staff faces many challenges in efficiently managing in-patient facilities in Intensive Care Unit for critical patients amidst rising patient demand and budget constraints. This report explores three variations of sequence to sequence algorithms in many to one problem setting to predict length of stay for patients admitted to intensive care unit and analyzes their behaviour on two medical information mart databases in terms of generalization and performance.**

I. Introduction

The length of stay (LoS) for in-patients significantly influences hospital costs in the United States, accounting for approximately 85-90% of inter-patient cost variation [1]. Prolonged hospital stays are linked to an elevated risk of hospital-acquired infections [2] and increased mortality rates [3]. Efficient hospital bed planning is essential to mitigate these risks and enhance patient experiences, especially in the intensive care unit (ICU), which incurs the highest operational costs [4] and operates with limited specialized staff and resources.

Currently, clinicians manually estimate discharge dates, but these estimates quickly become outdated and can be unreliable. For instance, Mak et al. [5] reported an average error of 3.82 days in clinician-made estimates. Automated systems leveraging electronic health records (EHR) hold promise in enhancing forecasting accuracy through cutting-edge models that can adapt to new data. This not only reduces the administrative burden on clinicians but also offers improved accuracy, enabling more sophisticated planning strategies such as scheduling high-risk elective surgeries on days with greater availability. Particularly, Long Short-Term Memory, Transformer and Temporal Pointwise Convolutional Networks based models are experimented on the EHR datasets to reliably predict the length of stay. The basic idea behind the author's approach is to capture temporal patterns and compute feature interactions from the historical data for handling irregular sampling, sparsity and skew which are common in EHR datasets.

II. Scope of Reproducibility

Our task is to predict the remaining length of stay until discharge in the patient's ICU stay using static features and time series. In this report, we aim to work on the following hypothesis in building our regression problem using sequence models:

- Replicate LSTM, Transformer and author's best performing model Temporal Pointwise Convolution for predicting length of stay on MIMIC-IV and eICU datasets and prove that TPC perform the best in capturing temporal patterns. LSTM, Transformer and TPC models learns representation from sequential data but the key differences are the architectures that can enhance the performance. Standard LSTM architecture has a mechanism to consider time steps from a long sequence of inputs that influence the predictions. When it comes to transformer based architectures, the multi-head self attention mechanism focuses on different aspects of the input sequence simultaneously allowing parallel computation and captures global context better. Unlike LSTM, transformer based models are adaptable to variation in sequence lengths. While TPC relies on 1x1 convolutions applied on the input sequence to capture temporal patterns and across channels to cature relationships. This reproducibility exercise focuses on effectiveness of TPC on prediction length of stay of in-patients in ICU.
- Experiment with Mean squared error and Mean squared logarithm error loss functions and prove that MSLE loss function handles positive skew in target class. Typically, length of stay in ICU has skew in the distribution which affects the performance in prediction using Deep Learning algorithms. This part focuses on approach to handle this issue by modifying the loss function. More details are shared in the dataset description section in this report.

III. Methodology

A. Model Description

All the models were manually fine-tuned to arrive at the optimal hyperparameters to reproduce the results. On an abstract level, all the three models follow the architecture flow as shown below in Fig:1:

B is the batch size.
F is the number of input features in the time series.
T is the number of time steps.
S is the number of static features.

Note: F, T, S values change based on the dataset type which are discussed below.

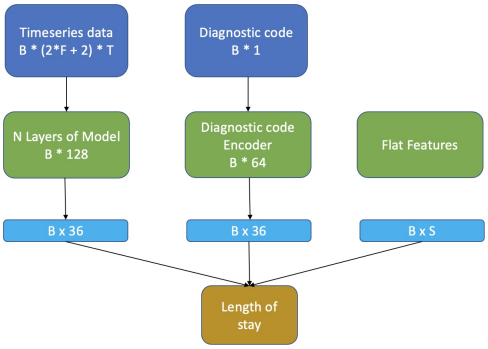

Fig. 1. Model flow

1) LSTM: Standard LSTM architecture is used to capture long term dependencies using temporal patterns on the input sequence length of T. 2 Layers of LSTM with 128 hidden units are used in uni-directional setting with a dropout of 0.25 and linear layers to reduce the dimensions to 36 before a dense layer to predict the length of stay. The linear layers are trained with ReLU activation function along with batch normalization. We used MSLE loss function and experimented with Adam optimizer. The model is trained for 8 epochs from scratch with no pre-trained weights to predict the length of stay.

2) Transformer: Transformer has gained traction in sequence-sequence problems that can process the whole sequence in parallel unlike an LSTM which performs sequentially and causes scaling issues. With the concept of multi-headed attention and positional embeddings, transformers are expected to capture variations in time series data better than LSTM variants. Following the author's implementation, we used 2 layers of Transformer Encoder architecture with one multi-head self attention followed by concatenation of diagnostic embedding representation along with static features to predict the length of stay. Similar to LSTM, we used MSLE loss function with Adam optimizer and trained the model from scratch for 15 epochs.

3) Temporal Pointwise Convolution: Author introduces a new approach using a combination of 1x1 convolutional layers to capture both temporal features and channel interactions. The basic idea behind TPC is that the temporal convolution layers that convolve over the time dimension capture the temporal dependencies and pointwise convolution capture interactions between features allowing the model to explore patterns which would have been difficult for LSTM and Transformer based models. Similar to LSTM and Transformer models, Temporal and pointwise convolution layers can be stacked on top of each other to extract rich trends from the input sequences. In addition to the stacking, we also employ skip connections between layers similar to DenseNet to arrange shared-source connection of information. TPC architecture does not share weights across features in Pointwise layer stack and share weights across time steps in Temporal layer stack. The weights are not shared in pointwise layers to avoid information loss during interactions between channels. In TPC, temporal and pointwise convolution are computed in parallel and the concatenated to output the final prediction. This approach enables scaling up process features independently and effectively carry temporal information using skip connections.

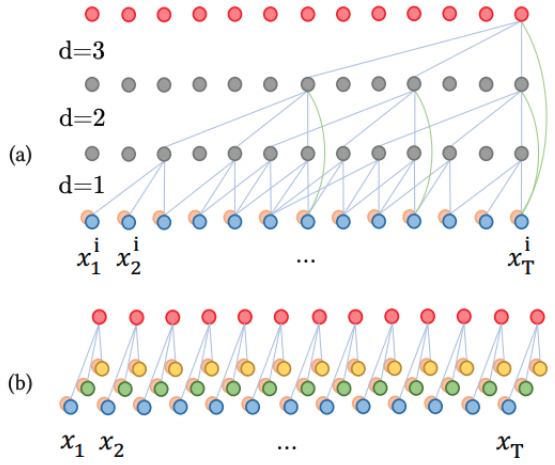

Fig. 2. a. Temporal convolution with skip connections b. Pointwise Convolution. This image is directly taken for brevity from the original paper: https://emmarocheteau.com/publication/tpc/tpc.pdf

The architecture of the Temporal pointwise convolutional model is as follows:

- Number of Temporal Pointwise convolutional layers = 8
- Number of epochs = 10
- Loss function = MSLE
- Optimizer = Adam
- Learning rate = 0.001

B. Data description

We used PostgreSQL and followed guidelines from the official resources to perform ETL operations and used python to create the preprocessed datasets and uploaded them to gatech box for running our experiments for both eICU data (https://physionet.org/content/eicu-crd/2.0/) and MIMIC data (https://physionet.org/content/mimiciv/0.4/) . Links to the preprocessed datasets are provided in the github repository.

1) eICU: eICU is a deidentified data collected from multiple hospitals across United States between 2014-2015 and has 118,535 unique patients who are adults (> 18 years) comprising 146,671 ICU stays. Similar to the author's approach we are using 87 time series features and 17 static features. Before feeding in to each of the model, we discretize the categorical features and scale the numerical features between -1 and 1. The data consists of duplicate patients with multiple ICU visits. Hence, to avoid leakage the cross validation split is performed on patients. We employed train(70%)-validation(15%)-test(15%) split on patients for the cross validation as suggested by the author of the original paper. One observation on the target distribution is that it is heavy skewed as is and after applying log transformation to the target variable the distribution

becomes closer to normal which theoretically should reduce model bias in predictions. Once the preprocessed files are generated, we ran a baseline mean-median model to compare with the author's output for sanity. The target variable is assigned as the remaining length of stay to each hour of stay, starting at 5 hours and ending when the patient dies or is discharged. Model is trained to predict at every hour of the stay. A maximum of first 14 days are included in the timeseries of any patient's stay.

Number of Patients	118,535
Train	82,973
Validation	17,781
Test	17,781
Number of stays	146,671
Train	102,749
Validation	22,033
Test	21,889

TABLE I
DATA DISTRIBUTION FOR EICU DATA

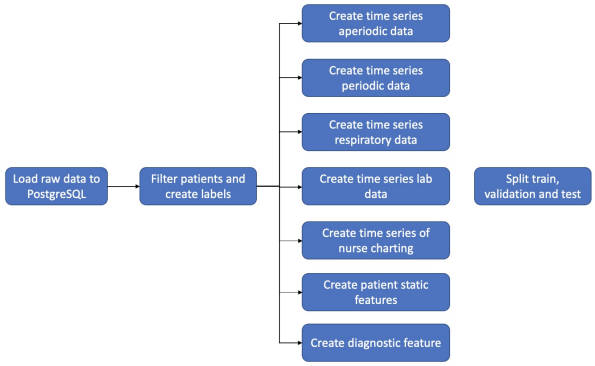

Fig. 3. eICU data Flow

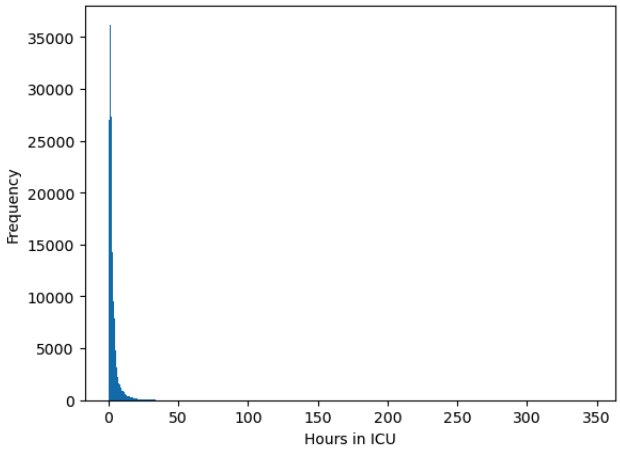

Fig. 4. Frequency distribution of the length of stay for eICU data

2) MIMIC-IV: MIMIC-IV data has 50,048 unique patients comprising 69,619 ICU stays admitted between 2008 and 2019. We are using 101 time series features based on lab,

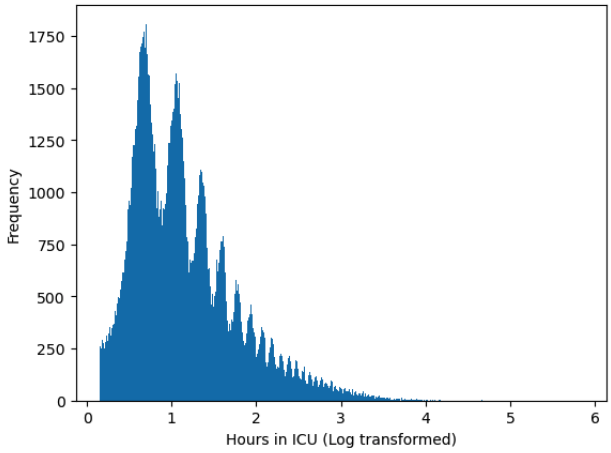

Fig. 5. Frequency distribution of the length of stay after log transformation for eICU data

nurse-charting and vitals monitoring and 12 static features. Similar approach is taken in the case of MIMIC-IV as eICU data for preprocessing. In MIMIC data, around 243 time series are extracted out of which 71 did not have any variation and hence were dropped to reduce noise in the data. The distribution of the target variable is observed to be similar as compared to eICU data with a mean of 3.98 hours in ICU. Fig:6 represents the data flow for ETL to create the final preprocessed data for running model experiments. Please note that there is no diagnosis code for MIMIC data.

Number of Patients	50,042
Train	35,028
Validation	7,507
Test	7,507
Number of stays	69,609
Train	48,848
Validation	10,497
Test	10,264

TABLE II
DATA DISTRIBUTION FOR MIMIC DATA

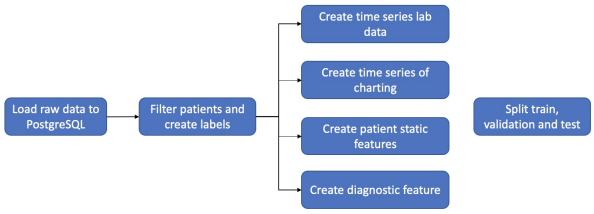

Fig. 6. MIMIC data Flow

C. Computational Implementation

- MIMIC-IV data is about 6.7GB and eICU data is about 5.1GB uncompressed. Extracting raw data and applying

necessary transformations are performed in PostgreSQL. We used Standard_DS13_v2 machine on Azure which has 8 cores and 56GB RAM to perform this step. We were able to run eICU ETL on local Macbook but MIMIC data required close to 42GB of RAM and hence, we utilised Azure virtual machines to perform ETL. This step creates the train-validation-test data splits for both datasets along with sample datasets. We ran this step only once and uploaded the datasets to gatech box storage.

- We used python as our base language and PyTorch as our base framework to implement models.
- We developed our modelling code on Google Colab on sample dataset.
- We used STANDARD_NC6S_V3 instance on Azure which has V100 GPU to run our final experiments. In order to facilitate multiple experiments we used low batch size of 8.

Although the paper employs precise learning rate from grid search tuning. We employed manual fine-tuning and used below hyperparameters to run models for both datasets:

Parameter	eICU	MIMIC-IV
Learning rate	0.001	0.001
Epochs	8	8
LSTM Dropout	0.2	0.25
Training time(hours)	0.6	1.1
Num of layers	2	1
Hidden layer size	128	128

TABLE III
LSTM HYPERPARAMETERS

Parameter	eICU	MIMIC-IV
Learning rate	0.0001	0.001
Epochs	15	15
Number of layers	6	2
Number of heads	2	1
Transformer Dropout	0	0.05
Training time(hours)	0.4	0.8
Feedforward size	64	64

TABLE IV
TRANSFORMER HYPERPARAMETERS

Parameter	eICU	MIMIC-IV
Learning rate	0.002	0.002
Epochs	8	10
Number of layers	8	8
Kernel size	4	5
Number of temporal kernels	12	11
Point convolution size	13	5
Transformer Dropout	0.05	0.05
Training time(hours)	1.5	2.4

TABLE V
TPC HYPERPARAMETERS

D. Code

- Data preprocessing using PostgreSQL for both datasets is adapted from the repository of the original paper.
- We developed the modelling code for LSTM, Transformer and TPC.
- All the code is committed to the gatech github: https://github.gatech.edu/smanyam3/cse6250_project

IV. RESULTS

A. Evaluation metrics

1) Mean absolute deviation: MAD is calculated as the mean absolute difference between actual length of stay and predicted length of stay.

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i|$$

2) Mean Absolute Percentage Error: MAPE is calculated as the mean absolute percentage difference between actual length of stay and predicted length of stay.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \times 100$$

3) Mean Squared Error:: MSE is calculated as the mean squared difference between actual length of stay and predicted length of stay.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

4) Mean Squared Logarithmic Error:: MSLE is calculated as the mean squared logarithmic difference between actual length of stay and predicted length of stay.

$$MSLE = \frac{1}{n}\sum_{i=1}^{n}(\log(1+Y_i) - \log(1+\hat{Y}_i))^2$$

5) Coefficient of determination:: R-squared is defined as the proportion of variance in the dependent variable that is predictable from the independent variables.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}$$

6) Cohen's Kappa score: : Kappa is defined as the possibility of agreement occurring by chance. Higher the kappa score, better the model's performance.

Data	Model	MAD	MAPE	MSE	MSLE	R^2	Kappa
eICU	Mean	3.58	418.24	35.09	2.91	0	0
	Median	3.09	194.77	39.04	2.19	-0.11	0
	LSTM	2.40	124.11	27.50	1.47	0.09	0.33
	Transformer	2.37	111.34	26.32	1.41	0.09	0.30
	TPC	**1.74**	**55.91**	**19.2**	**0.4**	**0.31**	**0.66**
MIMIC-IV	Mean	5.24	474.85	77.74	2.79	0	0
	Median	4.57	216.81	86.78	2.09	-0.11	0
	LSTM	3.67	100.85	67.71	1.26	0.13	0.42
	Transformer	3.68	118.08	64.71	1.23	0.16	0.44
	TPC	**2.28**	**30.94**	**43.00**	**0.19**	**0.44**	**0.85**

TABLE VI
PERFORMANCE OF TPC MODEL COMPARED TO BASELINE MODELS ON EICU AND MIMIC-IV DATASETS

Below are the observations of model behaviour on both datasets based on our experiments and results:

- Table VI suggests that the TPC model outperforms other baseline models on both datasets. On this particular problem setting, we can attribute the TPC model performance

Data	Model	MAD	MAPE	MSE	MSLE	R^2	Kappa
eICU	TPC (MSE)	2.20	124.12	21.4	0.9	0.27	0.55
	TPC (MSLE)	1.74	55.91	19.2	0.4	0.31	0.66
MIMIC-IV	TPC (MSE)	2.63	63.12	51.29	0.73	0.28	0.68
	TPC (MSLE)	2.28	30.94	43.00	0.19	0.44	0.85

TABLE VII
PERFORMANCE OF TPC MODEL WITH OPTIMIZING FUNCTIONS AS MSE AND MSLE ON EICU AND MIMIC-IV DATASETS

to the ability to capture the temporal variation and the feature interactions.

- From Fig:7 and Fig:8, it can be observed that LSTM and Transformer based models perform similarly. Perhaps the transformer based models could show further improvement by the right choice of number of layers and attention heads.
- Choosing MLSE as the loss function reduced the skew in target variable and there by reducing bias to considerably improve performance proving the hypothesis of this report. The comparison results are displayed in Table VII.
- Mean and median scores of MIMIC data exactly match with what provided in the original paper. However, we could not reproduce the results on mean-median baseline for eICU. This minor fluctuations could probably be due to changes in filtering patients in the ETL process.
- The metrics observed for all the models are not exactly the same but comparable to the metrics observed in the original paper.

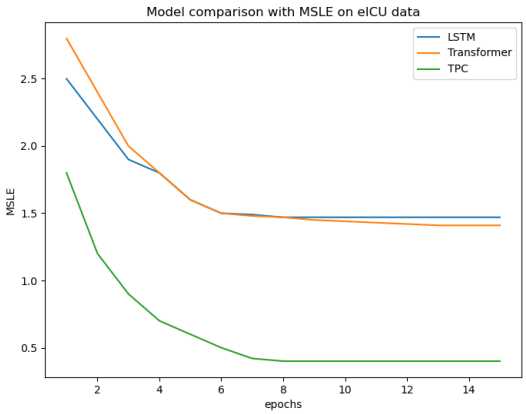

Fig. 7. Performance comparison on eICU data

V. DISCUSSION

The paper is reproducible with some additional considerations. The authors provide clear and detailed instructions on how to preprocess the data, train the models, and evaluate the results. We were able to replicate the results for predicting length of stay on both eICU and MIMIC-IV datasets. It was fairly easy to implement the models with inputs from the original paper. However, we encountered some challenges during the reproduction process. First, feature selection was performed based on feature variation in the MIMIC data and

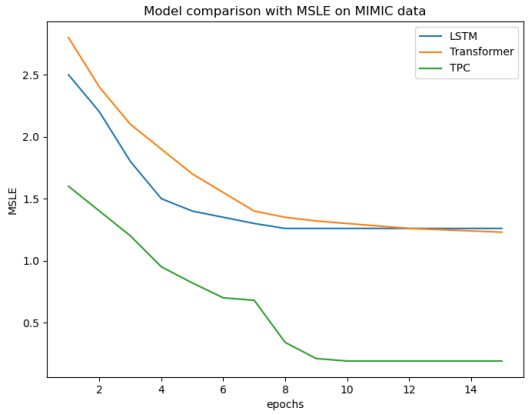

Fig. 8. Performance comparison on MIMIC data

we realized we could have dropped more time series features to reduce noise in the data and improve the performance of the models. More guidance or insights from the authors on time series specific features would have been beneficial further improve the models' performance. Specifically, we would be interested to implement feature selection techniques such as permutation importance. Second, author employed hyperparameter tuning as a grid search, however, this takes huge amount of time considering the dataset size. Instead of grid search, we manually tuned the algorithms to arrive at best hyperparameters instead of directly using what authors have provided to understand the behaviour of the models. It is unclear on the process of approaching hyper parameter tuning on such large datasets, having an intuition on the algorithms from the original paper on these particular datasets would have guided us well in reducing our time and effort during tuning process. Last, remaining challenges were the computational demands of the MIMIC-IV dataset, and the GPU requirements for training the TPC model. Preprocessing the MIMIC data required close to 42 GB of RAM, exceeding the capabilities of the local laptop. To solve this challenge, we used Azure virtual machine to perform ETL due to their high RAM capacity. Also, training the TPC model requires a GPU with sufficient compute power. We used STANDARD_NC6S_V3 instance on Azure, which has a V100 GPU, to train our models. Using a low batch size of 8 facilitated multiple experiments without compromising performance. Despite of these challenges, evaluating the results was relatively easy, as the authors provided clear instructions and evaluation metrics. This project helped us understand the variations of sequence based models and their effectiveness on one of the key uses cases in Medical industry.

VI. REFERENCES

A. Research papers

1) John Rapoport, Daniel Teres, Yonggang Zhao, and Stanley Lemeshow. 2003. Length of Stay Data as a Guide to Hospital Economic Performance for ICU

2) Mahmud Hassan, Howard Tuckman, Robert Patrick, David Kountz, and Jennifer Kohn. 2010. Hospital Length of Stay and Probability of Acquiring Infection. International Journal of Pharmaceutical and Healthcare Marketing 4 (2010), 324–338 Patients. Medical Care 41 (2003), 386–397.
3) Kevin B. Laupland, Andrew W. Kirkpatrick, John B. Kortbeek, and Danny J. Zuege. 2006. Long-term Mortality Outcome Associated With Prolonged Admission to the ICU. Chest 129, 4 (2006), 954 – 959.
4) Mathias C Blom, Karin Erwander, Lars Gustafsson, Mona Landin-Olsson, Fredrik Jonsson, and Kjell Ivarsson. 2015. The probability of readmission within 30 days of hospital discharge is positively associated with inpatient bed occupancy at discharge – a retrospective cohort study. BMC Emergency Medicine 15, 1 (2015), 37. https://doi.org/10.1186/s12873-015-0067-9
5) Deborah Dahl, Greg G Wojtal, Michael Breslow, Randy Holl, Debra Huguez, David Stone, and Gloria Korpi. 2012. The High Cost of Low-Acuity ICU Outliers. Journal of healthcare management / American College of Healthcare Executives 57 (2012), 421–434.
6) Gregory Mak, William D. Grant, James C McKenzie, and John B. Mccabe. 2012. Physicians' Ability to Predict Hospital Length of Stay for Patients Admitted to the Hospital from the Emergency Department. In Emergency medicine international.

B. Datasets

- **MIMIC**: https://physionet.org/content/mimiciv/0.4/
- **eICU**: https://physionet.org/content/eicu-crd/2.0/
- eICU preprocessed data URL - https://gatech.box.com/s/8xdayoqvdo3js9zvyk6rh4za7a2a5j99
- MIMIC-IV preprocessed data URL - https://gatech.box.com/s/qon27mejjbxs1n2ecag4uvphjiyrdzvs

C. Code

- TPC Length of stay prediction - https://github.com/EmmaRocheteau/TPC-LoS-prediction
- Team Project Repository - https://github.gatech.edu/smanyam3/cse6250_project

D. Powerpoint Presentation

- https://gatech.app.box.com/s/6qofu7rx4ppkh6xksu5hodz18o6z97c0

E. Presentation video

- https://youtu.be/odGomiBgvzs