Salinity Prediction Of Raw Water Using Temporal Kolmogorov-Arnold Network

CS338.P22

Presented by:

N. Gia Bao[1] P. Nguyen Anh[1]

Instructor: PhD. Duong Viet Hang[1]

[1]Department of Computer Science
University of Information Technology

CS338 Seminar Presentation - 4 June 2025

Table of Contents

Table of Contents

Problem

- Water is a vital resource for human existence and development. Ensuring safe and clean water is essential for protecting public health and improving the quality of life. Therefore, research and forecasting of raw water salinity before treatment is extremely important to ensure the efficient and sustainable use of water resources.
- Challenges of pollution and climate change have significantly impacted water quality and salinity in water production areas. By building an accurate forecasting model, predicting the salinity of raw water before treatment could help to make appropriate decisions and treatment plans to minimize the impact of salinity on water supply systems and treatment processes.

Problem

In this project, we will:

- Build an accurate forecasting model to predict Sai Gon River's water salinity
- Compare predicting efficiency between a well-known model: Long Short-Term Memory and new promising model: Temporal Kolmogorov-Arnold Network.
- Compare TKAN results with baseline models

Table of Contents

Pipeline

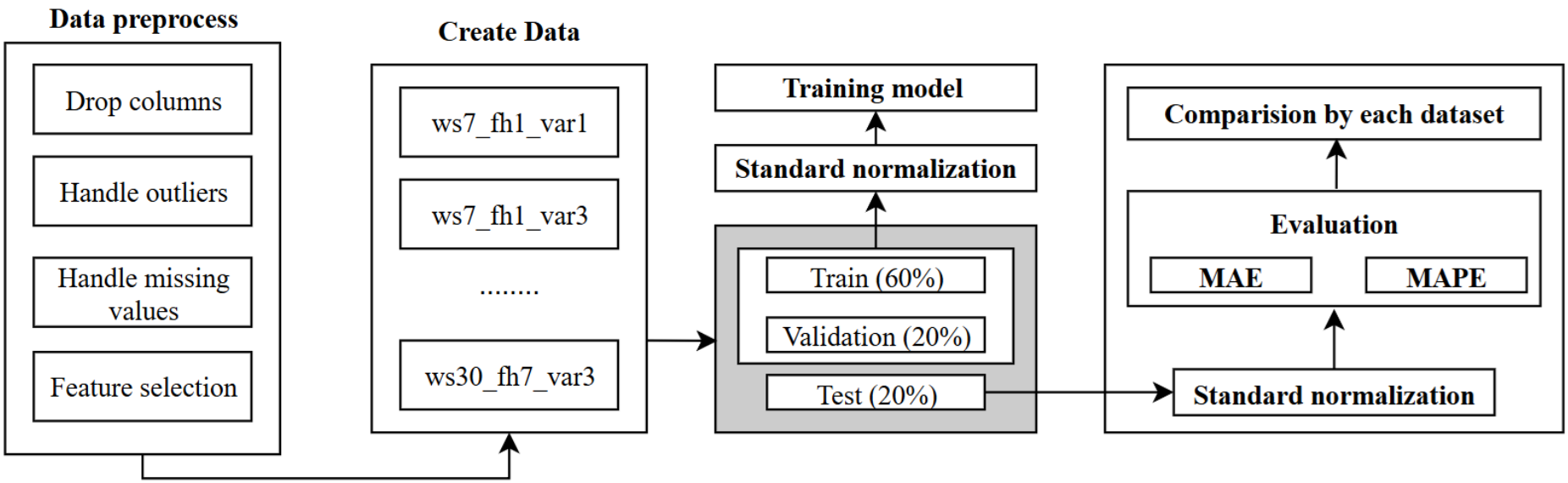


Figure: Pipeline

Data

Experimental data:

- Saigon River's water quality, which are monitored at the Hoa Phu pumping station, belongs to Tan Hiep Water Plant.
- Data are mean values of each day, collected over 6 years: from 1/1/2017 to 31/12/2022.
- Saved in single excel file, including **2191** entries.

Data preprocessing

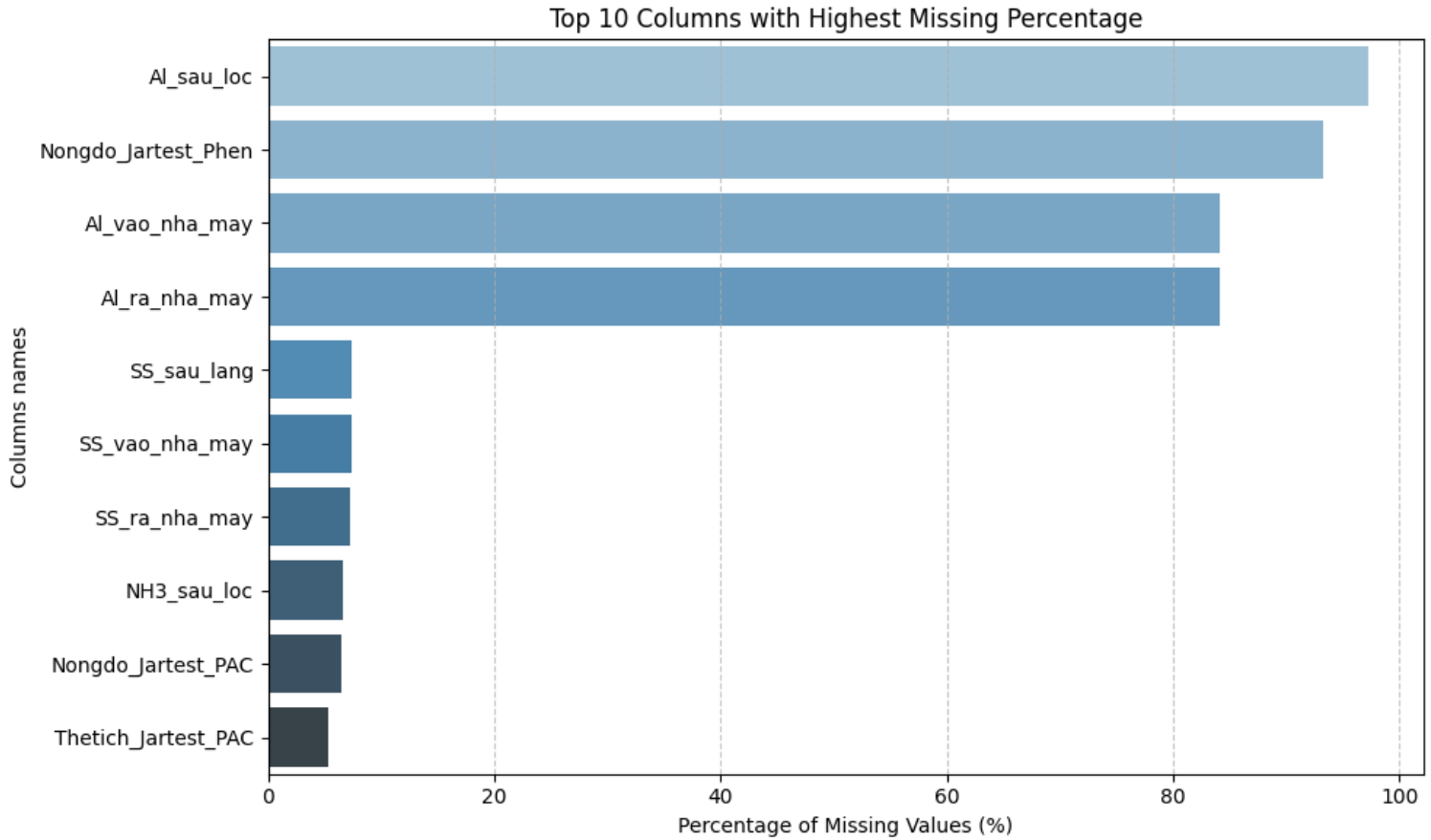
Handling columns with a high proportion of missing values:

- In this dataset, there many columns that are collected once in a few days, so they will have **considerably large number of missing values.**

Data preprocessing

Handling columns with a high proportion of missing values:

- In this dataset, there many columns that are collected once in a few days, so they will have **considerably large number of missing values.**



Data preprocessing

Handling columns with a high proportion of missing values:

- In this dataset, there many columns that are collected once in a few days, so they will have **considerably large number of missing values.**

$\Rightarrow$ We decided to drop these columns.

Data preprocessing

Handling columns with a high proportion of missing values:

- In this dataset, there many columns that are collected once in a few days, so they will have **considerably large number of missing values.**

$\Rightarrow$ We decided to drop these columns.

We will calculate percentage of missing values, and drop columns that have missing percentage higher than **10%**
$\Rightarrow$ Keep 46 columns, drop 4 columns.

Data preprocessing

Handling outliers:

- After examining the box plot, we observed that the data contains numerous outliers across all columns.

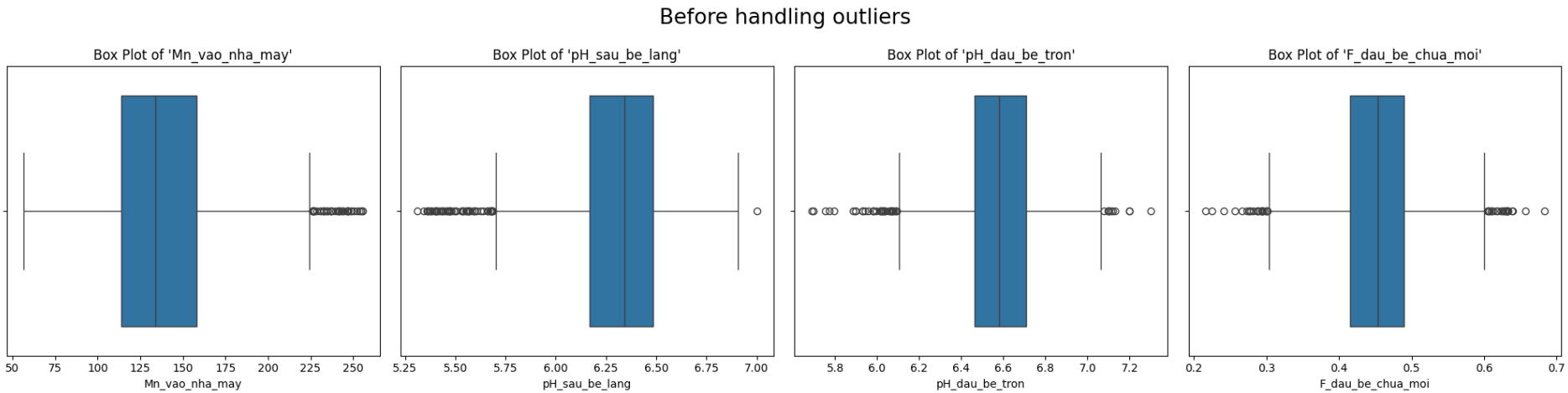


Figure: Box plot of 4 columns

Data preprocessing

Handling outliers:

- After examining the box plot, we observed that the data contains numerous outliers across all columns.
- We used 3SD-rules to address these outliers.

Data preprocessing

3SD rules, also known as **3σ rules**, is a statistical method for detecting outliers in datasets that follow a normal distribution.

Data preprocessing

3SD rules, also known as **3σ rules**, is a statistical method for detecting outliers in datasets that follow a normal distribution.
According to this rule:

- About 68% of data points fall within one standard deviation of the mean.
- About 95% fall within two standard deviations.
- Roughly 99.7% are found within three standard deviations.

Any data point beyond three standard deviations from the mean is considered an outlier.

Data preprocessing

Handling outliers:

- So, for each columns, we determine mean (μ) and standard deviation (σ). With any values that are out of bound $[\mu - 3\sigma; \mu + 3\sigma]$, will be replace with **NaN**.

Data preprocessing

Filling missing values:

- Although we drop columns with high missing proportion, there're still many missing values in kept columns.

Data preprocessing

Filling missing values:

- Although we drop columns with high missing proportion, there're still many missing values in kept columns.
- Because this is dataset is time series type, we will use **linear interpolation** to fill missing values.
- Since this technique requires knowing one data point before and one after the missing value in order to fill it, data points at the beginning of the list cannot be filled.
- We use **backward/forward fill** to address missing values like these.

Data preprocessing

Features selection:

- After cleaning dataset, we have a DataFrame of size $(2191, 46)$. Too many unnecessary features consume training resources and reduce performance, so we performed features selection to select the most important features for training.

Data preprocessing

Features selection:

- After cleaning dataset, we have a DataFrame of size $(2191, 46)$. Too many unnecessary features consume training resources and reduce performance, so we performed features selection to select the most important features for training.
- Since the objective is to predict salinity in raw water, we only consider measurements taken before the water undergoes any treatment at the plant.

Data preprocessing

Features selection:

- The selected features are:
  - `Man_song_saigon`: Salinity of water (objective function).
  - `Dodan_vao_nha_may`: Conduction of water.
  - `pH_Song_SG`: pH of water.

Create custom datasets

- To clarify how many variables, window size and forecast horizon should be used to get the best performance, we create datasets upon these parameters:
  - *ws*: window size, will be altered from 7 days to 15 days and 30 days.
  - *fh*: forcast horizon, will be changed from 1 day to 3 days and 7 days.
  - *var*: number of input variables. Should be 1 or 3.

Create custom datasets

- To clarify how many variables, window size and forecast horizon should be used to get the best performance, we create datasets upon these parameters:
  - *ws*: window size, will be altered from 7 days to 15 days and 30 days.
  - *fh*: forcast horizon, will be changed from 1 day to 3 days and 7 days.
  - *var*: number of input variables. Should be 1 or 3.
- The combination of the three variables results in a total of 18 datasets.
- From this point onward, each dataset will be referred to using the naming format:
  $ws\{window\_size\}\_fh\{forecast\_horizon\}\_var\{num\_vars\}$.

Pipeline

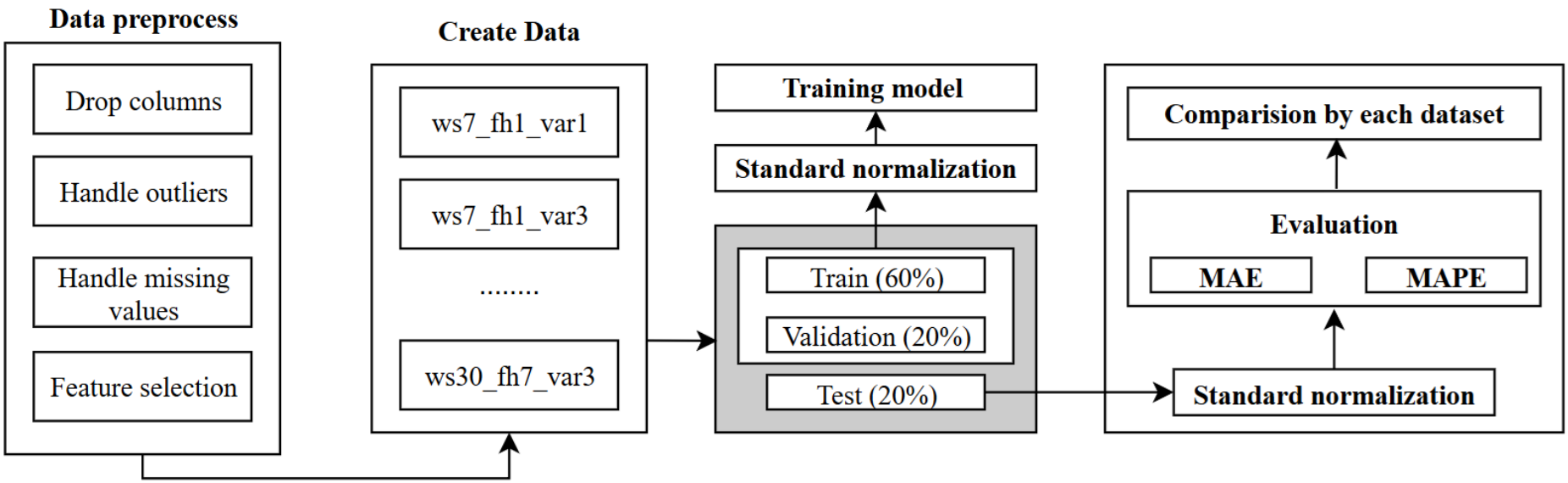


Figure: Pipeline

Table of Contents

Model Configuration: TKAN

- **Model:** Temporal Kolmogorov - Arnold Network (TKAN)
- **Forecast horizon:** 1, 3, and 7 days

Model Configuration: TKAN

- **Model:** Temporal Kolmogorov - Arnold Network (TKAN)
- **Forecast horizon:** 1, 3, and 7 days

Forecast horizon = 1/3/7

- TKAN units: 8/32/64
- sub_kan_input_dim: 15/30/40, sub_kan_output_dim: 15/30/40
- Activation: `tanh`
- Recurrent activation: `sigmoid`
- Dense layer with linear activation (units = $n\_ahead = forecasthorizon$)
- Batch size: 128
- Max epochs: 50

Model Configuration: LSTM

- **Model:** Long Short-Term Memory (LSTM)
- **Forecast horizon:** 1, 3, and 7 days

Model Configuration: LSTM

- **Model:** Long Short-Term Memory (LSTM)
- **Forecast horizon:** 1, 3, and 7 days

Forecast horizon = 1/3/7

- LSTM units: 8/32/64
- Activation: `tanh`
- Recurrent activation: `sigmoid`
- Dense layer with linear activation (units = $n\_ahead = forecasthorizon$)
- Batch size: 128
- Max epochs: 50

Optimization Settings

- **Optimizer:** Adam
- **Learning Rate:** 0.001
- **Callbacks:**
  - EarlyStopping with:
    - `monitor = 'val_loss'`
    - `patience = 20`
    - `restore_best_weights = True`

Model Configuration: Baseline

Reference Baseline: Taken from a related paper using the same dataset.

Models included in the baseline:

- **ARIMA**
  - Univariate only.
  - Suitable for stationary time series.
- **Artificial Neural Network (ANN)**
  - Traditional feedforward network with input, hidden, and output layers.
  - Applicable to both univariate and multivariate data.
- **Convolutional Neural Network (CNN)**
  - Uses 1D convolutions for time series.
  - Captures short-term patterns like trends and peaks.

Model Configuration: Baseline

- **Gated Recurrent Unit (GRU)**
  - Simpler alternative to LSTM.
  - Faster training and effective for short-term forecasting.
- **Long Short-Term Memory (LSTM)**
  - Capable of learning long-term dependencies.
  - Supports both univariate and multivariate sequences.
- **Temporal Convolutional Network (TCN)**
  - Utilizes dilated convolutions and residual blocks.
  - Prevents future-to-past information leakage.

Note: The best-performing result among these models is used as the baseline for comparison.

Model Configuration: Baseline

Dataset	Model	MAE	MAPE (%)
ws7_fh1_var1	CNN_1Var	1.40	4.31
ws7_fh1_var3	GRU_3Var	1.50	4.67
ws7_fh3_var1	GRU_1Var	3.20	10.00
ws7_fh3_var3	ANN_3Var	3.33	10.30
ws7_fh7_var1	ANN_1Var	5.38	16.70
ws7_fh7_var3	ANN_3Var	5.61	16.30
ws15_fh1_var1	CNN_1Var	1.43	4.40
ws15_fh1_var3	GRU_3Var	1.62	5.15
ws15_fh3_var1	CNN_1Var	3.24	10.00
ws15_fh3_var3	GRU_3Var	3.47	10.40
ws15_fh7_var1	CNN_1Var	5.77	17.30
ws15_fh7_var3	LSTM_3Var	5.61	15.90
ws30_fh1_var1	GRU_1Var	1.52	4.57
ws30_fh1_var3	GRU_3Var	1.56	4.90
ws30_fh3_var1	GRU_1Var	3.42	10.70
ws30_fh3_var3	TCN_3Var	3.58	10.80
ws30_fh7_var1	CNN_1Var	5.50	17.10
ws30_fh7_var3	ANN_3Var	5.79	18.80

Evaluation

- **Metrics:**
  - Mean Absolute Error (MAE)
  - Mean Absolute Percentage Error (MAPE)
- **Compare:**
  - **T**KAN
  - **B**aseline
  - **L**STM
- **Evaluation Strategy:**
  - Evaluate each model on the same test set

Table of Contents

TKAN vs LSTM: MAE (Mean Absolute Error)

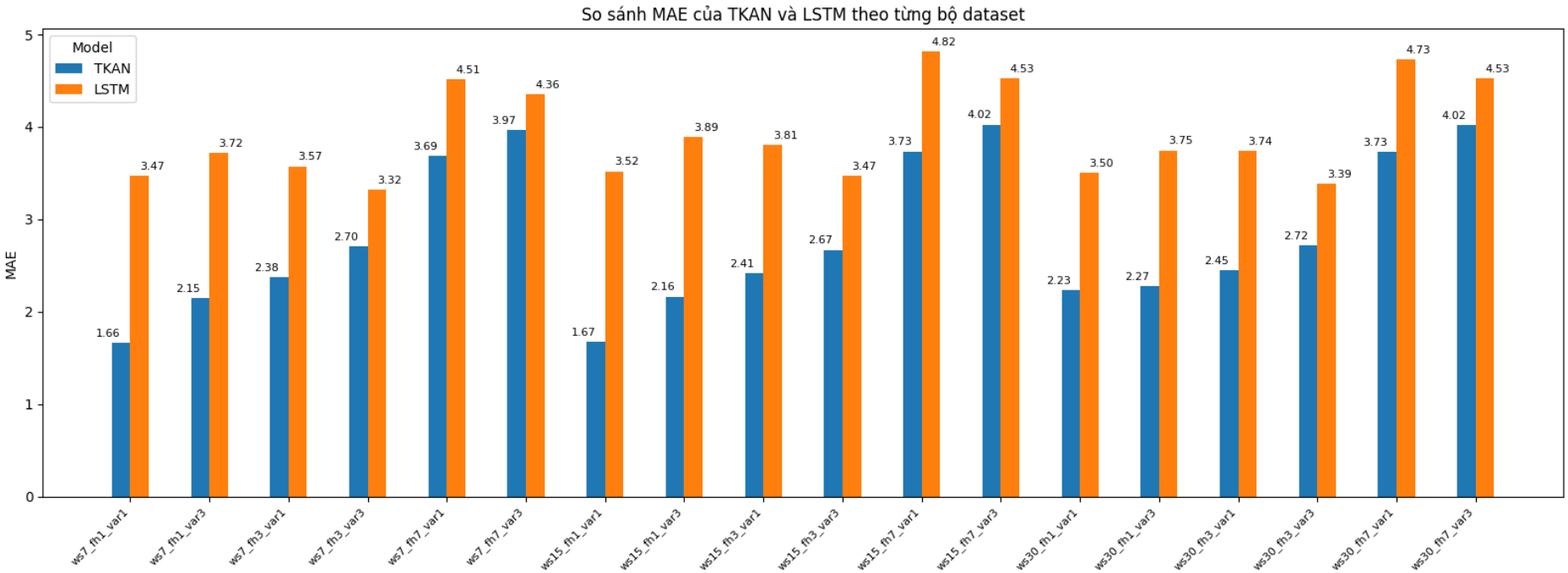


Figure: Average MAE over 50 runs

TKAN vs LSTM: MAPE (Mean Absolute Percentage Error)

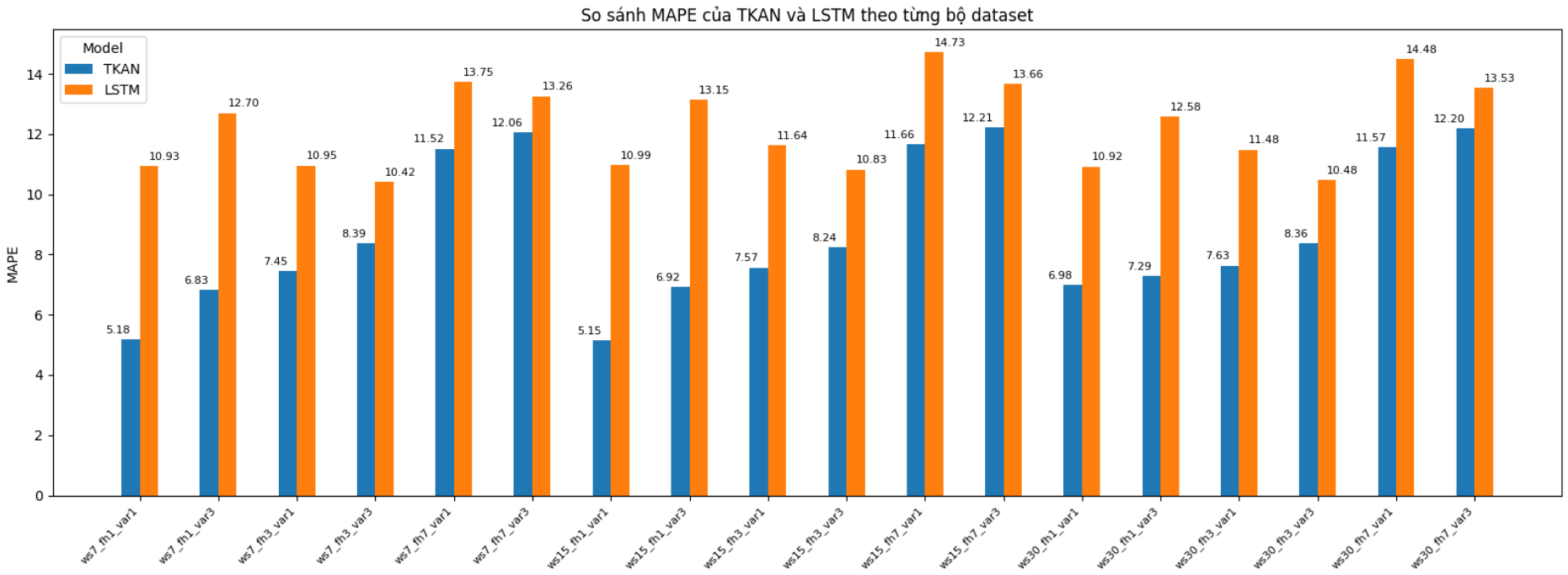


Figure: Average MAPE over 50 runs

TKAN vs LSTM: Training time

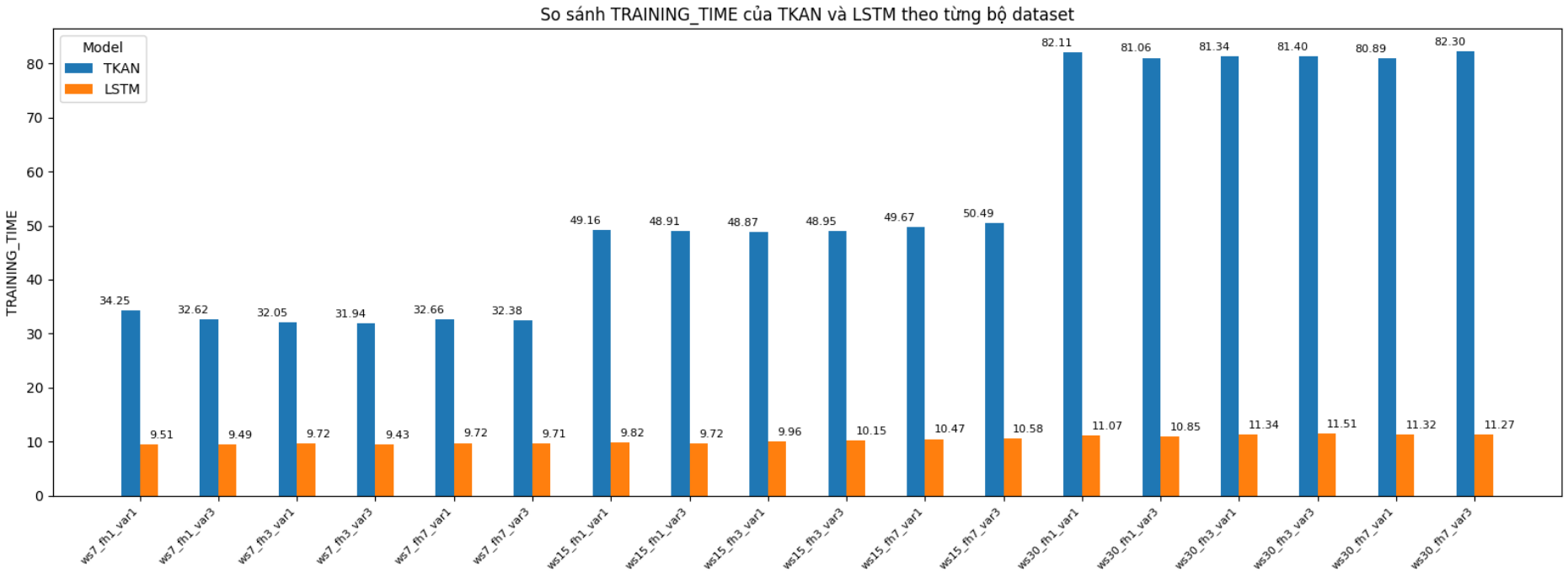


Figure: Average training time over 50 runs

TKAN vs LSTM: Analysis

TKAN vs LSTM:

- TKAN consistently achieves the lower MAE and MAPE across all datasets
- TKAN provides more stable performance over 50 runs.
- However, TKAN required more training time, whereas LSTM remained nearly unchanged. The increase in input dimensions led to longer training time, as both `sub_kan_input_dim` and `sub_kan_output_dim` were scaled up to accommodate higher-dimensional inputs.

TKAN vs LSTM: Analysis

TKAN vs LSTM:

- TKAN consistently achieves the lower MAE and MAPE across all datasets
- TKAN provides more stable performance over 50 runs.
- However, TKAN required more training time, whereas LSTM remained nearly unchanged. The increase in input dimensions led to longer training time, as both `sub_kan_input_dim` and `sub_kan_output_dim` were scaled up to accommodate higher-dimensional inputs.

$\Rightarrow$ **TKAN outperformed LSTM in both robustness and accuracy for time series forecasting. It is more suitable for high-accuracy tasks, albeit with increased training time.**

TKAN vs Baseline: MAE

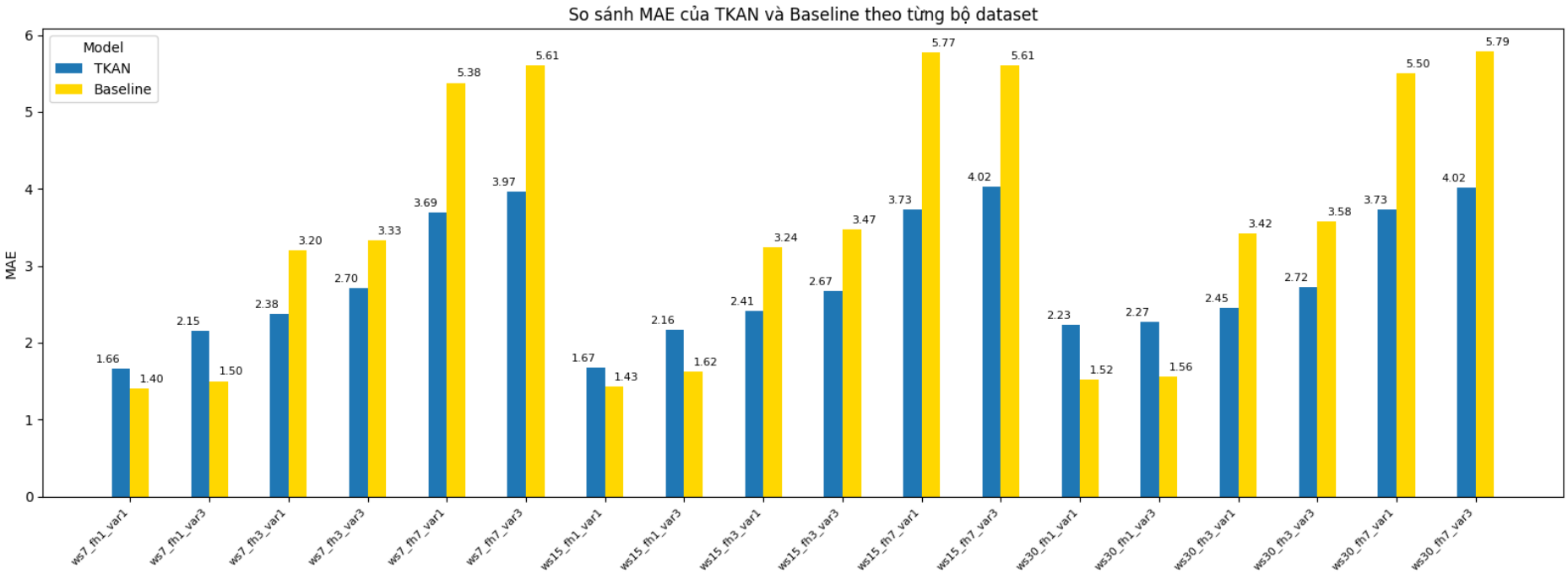


Figure: Average 50 runs TKAN vs Baseline

TKAN vs Baseline: MAPE

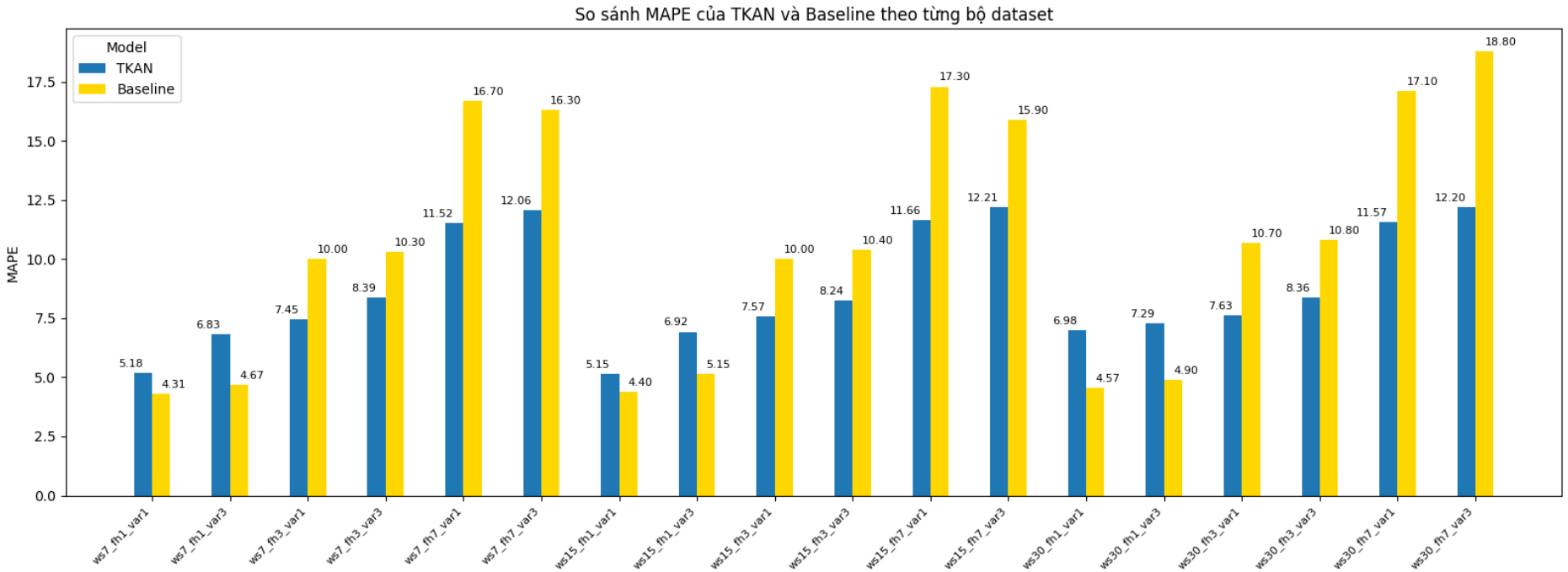


Figure: Average 50 runs TKAN vs Baseline

TKAN vs Baseline: Analysis

TKAN vs baseline models:

- TKAN achieves lower MAE in most of data
- TKAN performs best on datasets with forecast horizon of 3, 7. On datasets with forecast horizon of 1, its performance equal to or slightly worse than the Baseline.
- TKAN provides more stable performance in all datasets.

TKAN vs Baseline: Analysis

TKAN vs baseline models:

- TKAN achieves lower MAE in most of data
- TKAN performs best on datasets with forecast horizon of 3, 7. On datasets with forecast horizon of 1, its performance equal to or slightly worse than the Baseline.
- TKAN provides more stable performance in all datasets.

$\Rightarrow$ **TKAN demonstrates a more balanced performance compared to the baseline (i.e., the best among several other models). While it showed higher errors on some datasets, it significantly outperformed the baseline on others, particularly in long-term forecasts.**

Conclusion

- TKAN achieves higher accuracy and robustness compared to LSTM in time series forecasting tasks.
- It offers a more balanced performance than baseline models, especially excelling in long-term predictions.
- TKAN is well-suited for high-accuracy applications, though it requires longer training time.

THANK YOU FOR LISTENING!

TKAN vs Bayesian TKAN: MAE

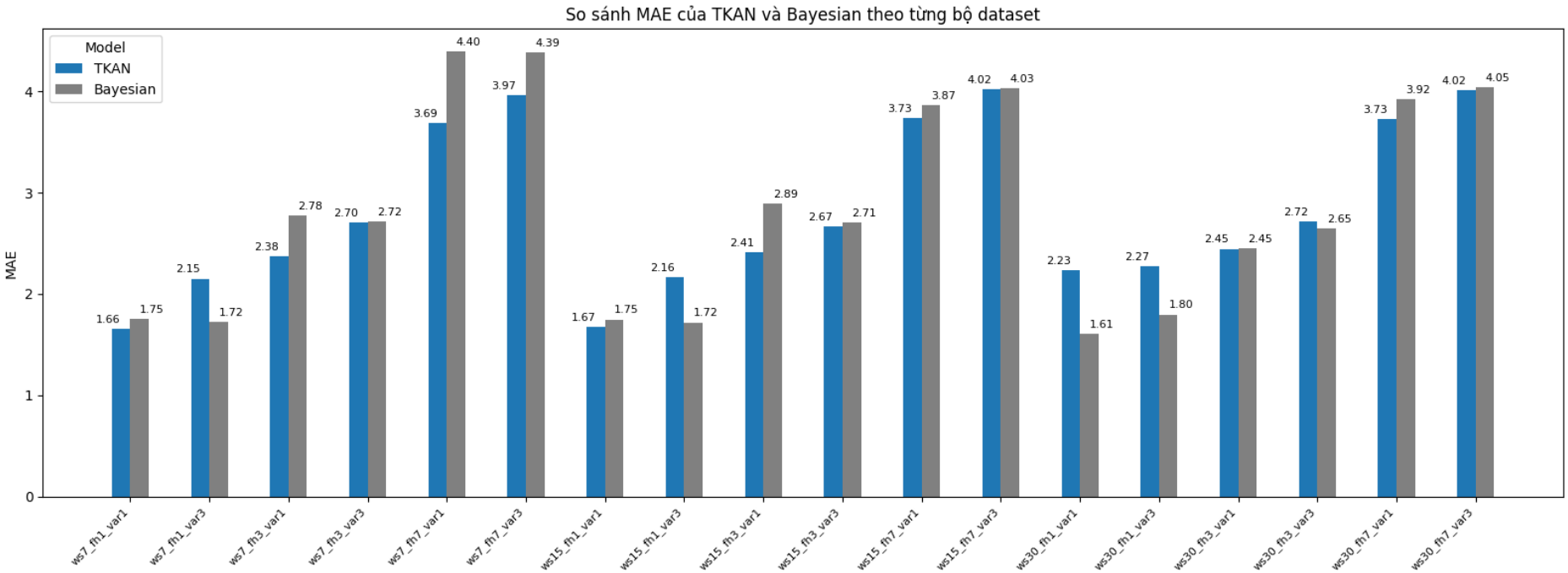


Figure: Average 50 runs TKAN vs average 10 runs Bayesian TKAN

TKAN vs Bayesian TKAN: MAPE

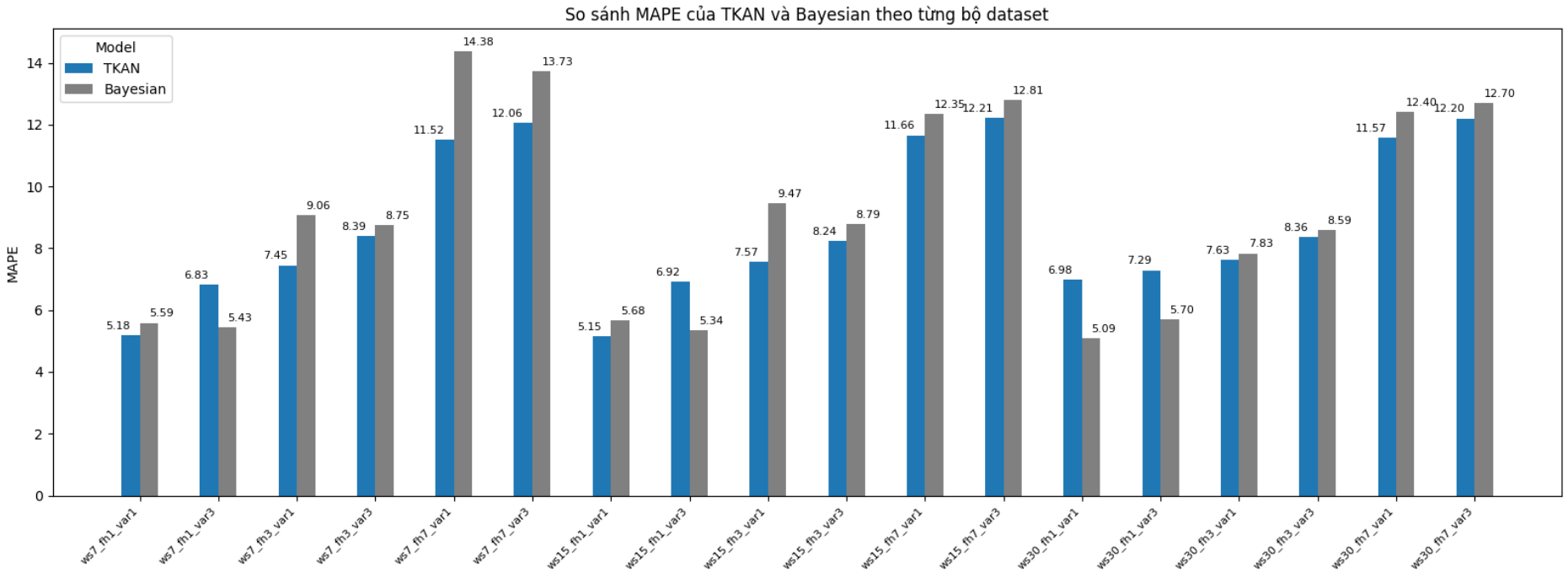


Figure: Average 50 runs TKAN vs average 10 runs Bayesian TKAN