

Configuration Manual

MSc Research Project
MSc In Cloud Computing

Sumanth Varma Kallepalli
Student ID: 22244263

School of Computing
National College of Ireland

Supervisor: Shaguna Gupta

National College of Ireland
MSc Project Submission Sheet
School of Computing



Student Name: Sumanth Varma Kallepalli
Student ID: 22244263
Programme: MSc In Cloud Computing **Year:** 2023-2024
Module: MSc Research Project
Lecturer: 16/09/2024
Submission Due Date: 16/09/2024
Project Title: Machine Learning Based Improved Cold Start Latency Prediction Frame Work In Serverless Computing
Word Count: 912 **Page Count:** 5

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Sumanth Varma Kallepalli

Date: 16/09/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Sumanth Varma Kallepalli
Student ID: 22244263

1 Introduction

The purpose of this configuration Manual is to guide users in installing and running the Cold Start Latency Prediction project. The project uses various machine learning (ML) and deep learning (DL) models, including Linear Regression, Decision Tree Regression, XGBoost, LSTM, and BiLSTM, to perform cold start latency predictions and alleviate such behavior in serverless computing environments. The sections below present the detailed guideline on software and hardware prerequisites, setting up the environment, handling data, training models, and evaluation.

2 Required Tools and Software

2.1 Software Components

- **Google Colab:** It is a cloud platform providing a Jupyter notebook kind of environment for Python code execution. This project uses Google Colab for its computational resources, including GPU support, to facilitate deep learning model training.
- **Python:** Ensure that we have installed the python 3.10. As it is very crucial, as it can be compatible with different libraries and tools used in a project.
- **Web Browser:** Ensure to have a supported updated web browser such as Google Chrome, Mozilla Firefox, or Safari so that you can use Google Colab.

2.2 Integration Of Google Drive

We are using Google Drive to store datasets, save the model, and manage output. Here are the steps to integrate Google Drive with Google Colab.

Step 1: Load The Dataset

- We have to load the dataset form the google drive
- Click on the Cold_Start_Latency.ipynb to open the file

My Drive > ColdStartLatency ▾

Type ▾ People ▾ Modified ▾





Name	Owner
 cold_start_latency_dataset_10k.csv	 me
 Cold_Start_Latency.ipynb	 me

Figure 1: Dataset and ipynb Files in Google Drive

Step 2: Mount google Drive

- Open your Google Colab notebook.



Figure 2: Google drive is mounted for the cold start latency

Step 3 : Navigate to the project directory

- After mounting, you can browse your files saved in Google Drive. Just make sure your dataset is correctly placed in the path under your Google Drive refer the figure 3.

```
9 # Load the dataset
10 file_path = '/content/drive/MyDrive/cold_start_latency_dataset_10k.csv'
```

Figure 3: Path to the dataset

2.3 Enabling the GPU Support

- Click on the runtime in google colab
- Go to the change run time option in runtime.
- Change the runtime and click on save

3 Python Environment and Library Setup

Installing Required Libraries: Although Google Colab already has a bunch of pre-installed libraries, you may still need to install other libraries required for your project refer the figure 4.

```
1 !pip install numpy pandas scikit-learn xgboost tensorflow keras matplotlib seaborn
2 |
```

Figure 4: Libraries Installation

The following command will have the libraries installed:

- **NumPy:** Very essential in numerical operations and array manipulation.

- **Pandas:** Very useful in data manipulation, especially in the DataFrame structure.
- **Scikit-learn:** A library offering a certain set of tools for machine learning that includes different algorithms and utilities for data preprocessing.
- **XGBoost:** A powerful library for implementing gradient boosting algorithms.
- **TensorFlow & Keras:** To construct and train deep learning models, such as LSTM and BiLSTM.
- **Matplotlib & Seaborn:** Used for making basic to advanced static and interactive visualizations.

3.1 Verifying Installation

Refer the figure 5

```
1 import numpy as np
2 import pandas as pd
3 import sklearn
4 import xgboost as xgb
5 import tensorflow as tf
6 import matplotlib.pyplot as plt
7 import seaborn as sns
```

Figure 5: Verifying Installation

If the above imports run successfully with out any errors, then your environment is ready and it is setup correctly.

4 Data Handling and Preprocessing

4.1 Loading the dataset

To start working on the data, load your dataset from Google Drive refer figure 6:

```
[3] 1 import pandas as pd
2
3 # Load the dataset from Google Drive
4 data_path = '/content/drive/MyDrive/cold_start_latency_dataset_10k.csv'
5 dataframe = pd.read_csv(data_path)
6
7 # Display the first few rows to understand the structure
8 dataframe.head()
9
```

	timestamp	machine_id	product_id	air_temperature	process_temperature	rotational_speed	torque	tool_wear	machine_failure	cold_start	execution_time	memory_used	function_size
0	2024-01-01 00:00:00	2	B	28.805342	30.737393	1549.411551	58.478127	162.707488	0	True	268.792551	537.373553	1024
1	2024-01-01 00:05:00	7	B	21.369154	35.751627	1568.730936	38.973967	227.147590	0	True	149.707344	406.515233	512
2	2024-01-01 00:10:00	5	B	26.659935	38.132205	1680.515346	41.310587	20.813391	0	False	63.007224	114.659275	128
3	2024-01-01 00:15:00	7	C	25.015071	29.245849	1451.471764	48.173933	240.750077	0	False	45.056165	96.122188	128
4	2024-01-01 00:20:00	5	B	26.353454	37.737616	1546.641193	46.060018	180.143729	0	True	108.284500	94.040355	128

Figure 6: Loading the dataset

This is the Synthatic dataset, it emulates conditions and variables contributing to cold-start latency in a serverless computing environment. Real-world scenarios were used in creating the data in such a manner that there would be a mix of features and outcomes diverse enough to properly train and evaluate any possible models.

4.2 Data Preprocessing

Preprocessing of the data is important because it helps make the machine learning models work effectively. The steps that are part of the preprocessing pipeline are explained below:

Handling The Missing Values Identify the missing data and treat it. You may choose to replace the missing values by the statistical measures such as mean, median, or mode, or drop them completely refer figure 7.

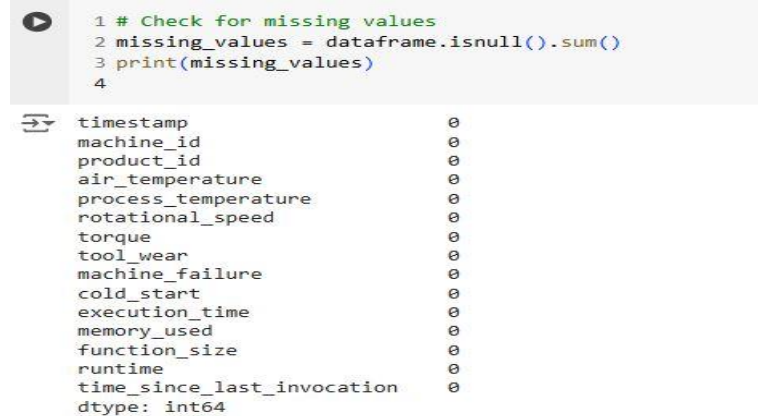


Figure 7: Missing Values

4.3 Dataset Visualization

Feature Scaling: Scale features to lie in a similar range using Min-Max scaling. This can make a machine learning model converge appropriately refer figure 8.

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)
X_scaled = pd.DataFrame(X_scaled, columns=X.columns)
X_scaled.head()
```

Figure 8: Feature Scaling

Feature Selection: Select the most relevant features using techniques like Recursive Feature Elimination (RFE), which iteratively removes the least important features based on a model's importance score refer figure 9.

```
rfe = RFE(model, n_features_to_select=10)
rfe.fit(X_scaled, y)
selected_features = rfe.support_
X_selected = X_scaled.loc[:, selected_features]
X_selected.head()
print("Selected Features:", X.columns[selected_features])
```

Figure 9: Feature Selection

5 Data Visualization

The visualizations exhibit key aspects of the cold-start problem. In Figure 10, there is a heat map of correlations that identify strong relationships of execution time with memory usage and function size, critical for predicting cold starts. We now turn to Figure 11, which exposes clearly how the execution time varies over multiple invocations by showing the decrease in time of execution as the system gets warmed up. Finally, Figure 12 compares memory usage and the execution time at a point in time; hence the importance of constant memory usage and different execution times that are crucial in implementing performance optimization for serverless functions.

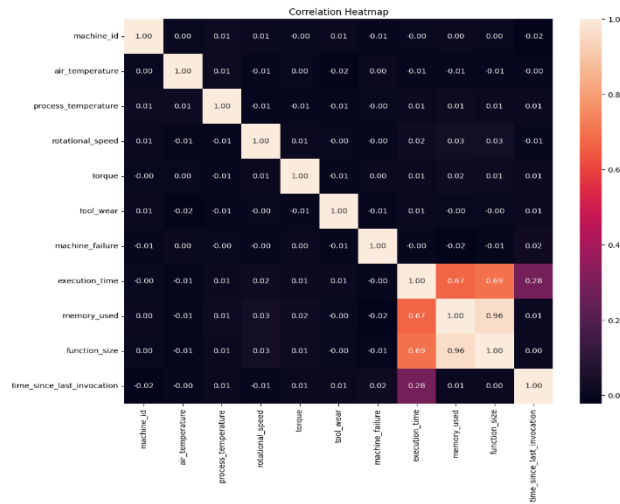


Figure 10: Co relation Heat Map Data visualization for cold start problem

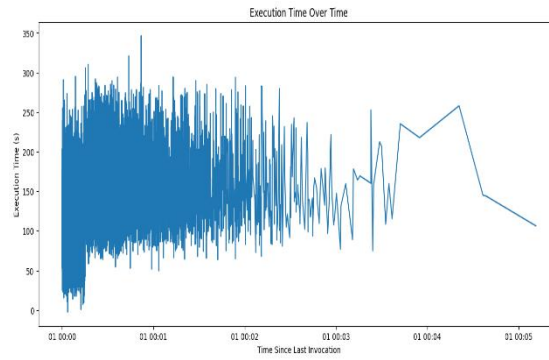


Figure 11: Data Visualization for cold start

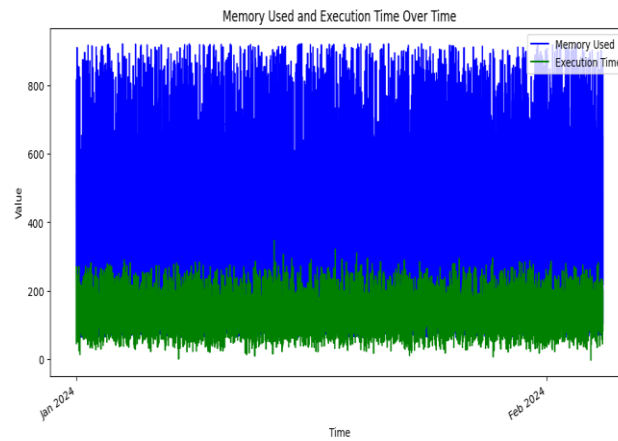


Figure 12: Memory used and execution over time for cold start

6 Conclusion

This setup manual aims to provide a step-by-step guide to setting up, running, and evaluating the Cold Start Latency Prediction project in Google Colab. The following will get you all set to manage your environment in advance, preprocess your data, train different models of machine learning, and compare their performance in prediction on cold start latencies over a serverless computing platform.