Industrial Internship Report on

”Smart City Traffic Recoganization Pattern”

Prepared by

Meet Gandhi

Executive Summary
This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner Uni-Convergence Technologies Pvt Ltd (UCT). This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 6 weeks’ time. My project was Traffic Recognization Pattern i.e to study about the traffic patterns at different junctions. This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solution for that. It was an overall great experience to have this internship.

TABLE OF CONTENTS

1 Preface

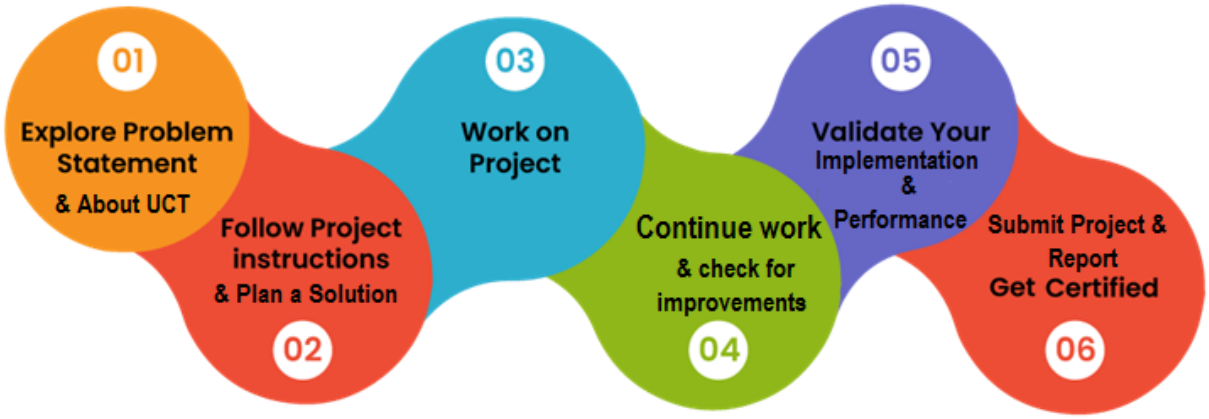
Summary of the whole 6 weeks' work.

About need of relevant Internship in career development.

Brief about Your project/problem statement.

Opportunity given by USC/UCT.

How Program was planned



Your Learnings and overall experience.

Thank to all (with names), who have helped you directly or indirectly.

Your message to your juniors and peers.

2 Introduction

2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various **Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.

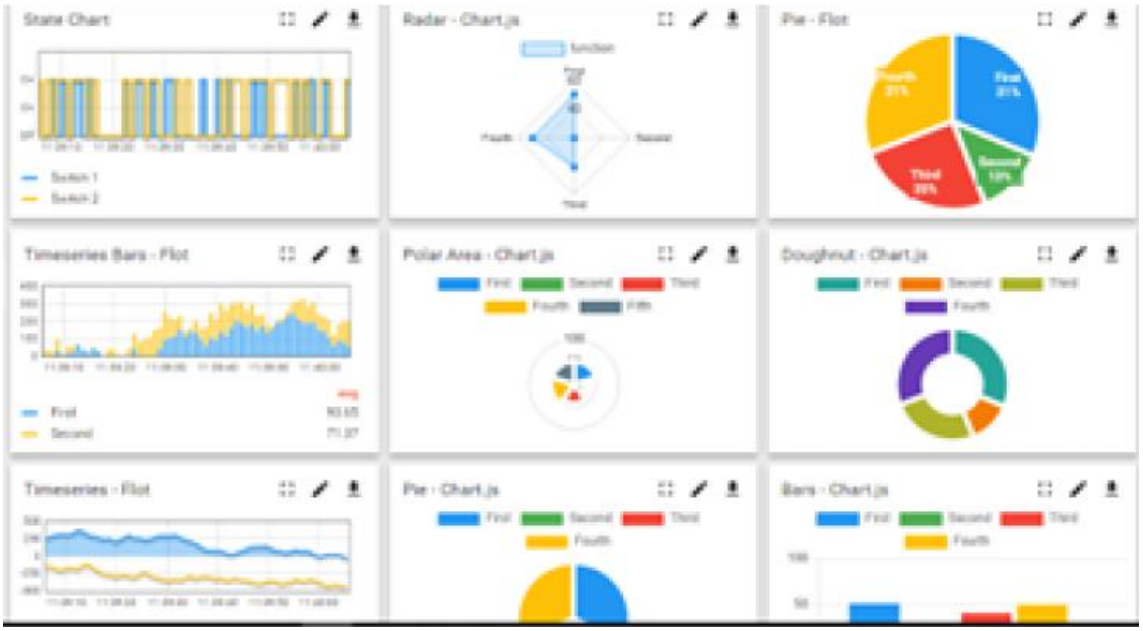
i. UCT IoT Platform (UCT Insight)

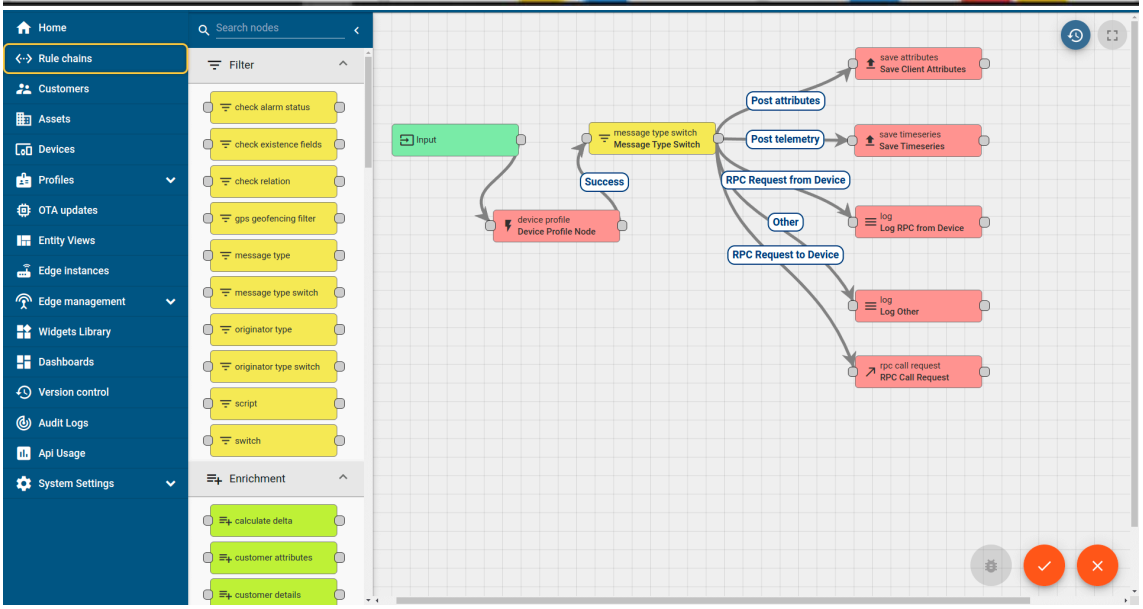
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- It enables device connectivity via industry standard IoT protocols - MQTT, CoAP, HTTP, Modbus TCP, OPC UA
- It supports both cloud and on-premises deployments.

It has features to

- Build Your own dashboard
- Analytics and Reporting
- Alert and Notification
- Integration with third party application(Power BI, SAP, ERP)
- Rule Engine





ii. Smart Factory Platform (FACTORY WATCH)

Factory watch is a platform for smart factory needs.

It provides Users/ Factory

- with a scalable solution for their Production and asset monitoring
- OEE and predictive maintenance solution scaling up to digital twin for your assets.
- to unleased the true potential of the data that their machines are generating and helps to identify the KPIs and also improve them.
- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solutions as per their demands.

Its unique SaaS model helps users to save time, cost and money.

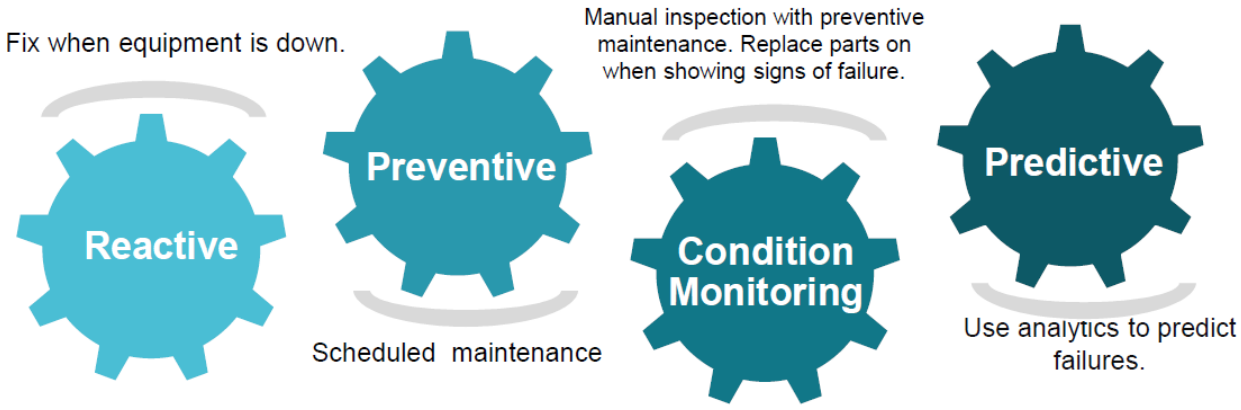
Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i

iii. LoRaWAN based Solution

UCT is one of the early adopters of LoRAWAN teschnology and providing solution in Agritech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

iv. Predictive Maintenance

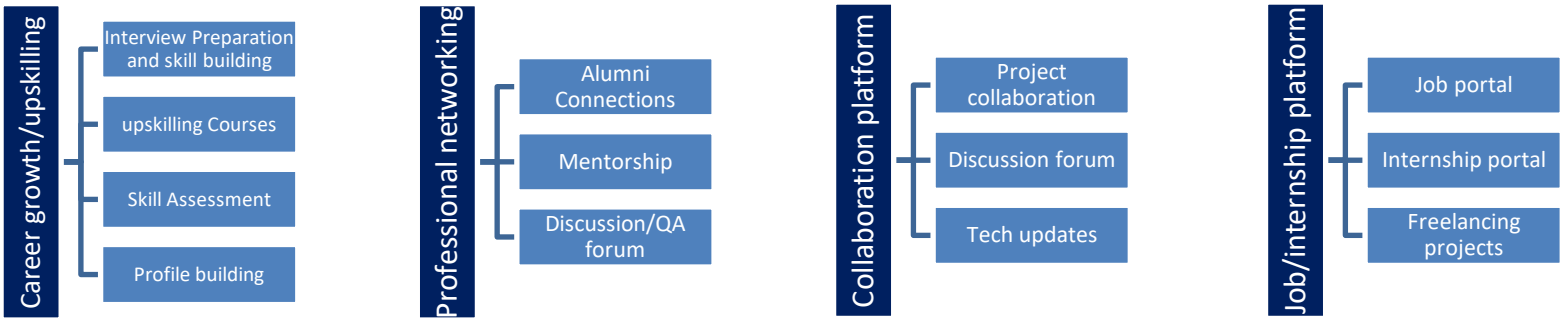
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.



2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

2.4 Objectives of this Internship program

The objective for this internship program was to

- ☛ get practical experience of working in the industry.
- ☛ to solve real world problems.
- ☛ to have improved job prospects.
- ☛ to have Improved understanding of our field and its applications.
- ☛ to have Personal growth like better communication and problem solving.

2.5 Reference

[1] Firstly , from Website called javapoint.

[2] David Opeoluwa Oyewola research paper

2.6 Glossary

Terms	Acronym
Data Visualization	
Machine Learning	ML
Gated Recurrent Unit	GRU

3 Problem Statement

Develop a machine learning model to predict traffic patterns in a smart city, enabling more effective traffic management and congestion reduction.

The urban landscape is rapidly evolving, with increasing population density and a growing number of vehicles contributing to traffic congestion in smart cities. Efficient traffic management is crucial for

ensuring smooth mobility, reducing pollution, and enhancing the overall quality of urban life. However, traditional traffic management methods often fall short in dealing with the dynamic nature of traffic patterns. As a result, there is a need for accurate predictions of traffic patterns that can empower city planners and transportation authorities to make informed decisions for optimized traffic flow and congestion reduction.

The problem at hand revolves around the challenge of predicting traffic patterns within a smart city context. Urban congestion not only leads to wasted time and increased fuel consumption but also negatively impacts air quality and the overall well-being of residents. To address these issues, accurate and reliable traffic pattern predictions are required.

By harnessing the power of data collection technologies, such as traffic cameras, GPS devices, and sensors embedded in road infrastructure, we can amass a wealth of information about traffic behavior. This data, when analyzed using machine learning algorithms, can provide valuable insights into the factors influencing traffic patterns, such as time of day, day of the week, weather conditions, special events, and road layout.

The primary goal of this report is to develop predictive models that can effectively forecast traffic patterns in a smart city environment. These predictions can serve as a foundation for evidence-based decision-making, enabling city planners to implement targeted interventions to mitigate congestion, optimize traffic signal timings, and allocate resources efficiently.

4 Existing and Proposed solution

Provide summary of existing solutions provided by others, what are their limitations?

Predictive Analytics Platforms

Commercial predictive analytics platforms offer pre-built models for traffic prediction using machine learning and historical data.

Limitations: These platforms might lack customization to specific city contexts. They can also have high costs associated with licensing and integration.

Machine Learning Ensembles (e.g. Random Forest):

Ensemble methods combine multiple models to improve prediction accuracy. Random Forest, a popular ensemble, has been applied to capture complex relationships in traffic data.

Limitations: Random Forest can overfit with too many trees or deep trees, reducing generalization. It might struggle to capture the dynamic nature of traffic patterns and sudden disruptions.

What is your proposed solution?

Deep Learning Model (GRU) and Time Series Analysis

What value addition are you planning?

Time series models have been used to forecast traffic patterns based on historical data. They capture temporal trends and seasonality in traffic.

Deep learning models like LSTM,GRU are capable of capturing long-range dependencies and temporal patterns in sequences, making them suitable for traffic prediction.

4.1 Code submission (Github link)

https://github.com/meet272002/upskillcampus/blob/main/SmartCityTrafficPatternPrediction.ipynb

4.2 Report submission (Github link)

https://github.com/meet272002/upskillcampus/blob/main/SmartCityTrafficPatternPrediction.ipynb

5 Proposed Design/ Model

The proposed model is a Gated Recurrent Unit (GRU) based architecture designed for time series prediction tasks. The GRU is a type of recurrent neural network (RNN) that is well-suited for sequential data analysis due to its ability to capture long-range dependencies and mitigate the vanishing gradient problem. This model aims to predict future values in a time series given historical data.

Model Architecture

The architecture of the proposed GRU-based model is as follows:

Input Layer: The input shape is set to (X_Train.shape[1], 1), where X_Train.shape[1] represents the number of time steps in the input sequence. This choice is made to allow the model to learn temporal patterns within the data.

Gated Recurrent Unit (GRU) Layers: The model consists of three consecutive GRU layers. Each GRU layer is configured to return sequences, meaning that it produces outputs at each time step rather than just the final time step. This helps the model capture sequential dependencies throughout the input sequence.

Activation Function ('tanh'): The hyperbolic tangent activation function ('tanh') is chosen for the GRU units. This nonlinearity introduces a squashing effect that helps the model capture both positive and negative patterns in the data.

Dropout Layers: After each GRU layer, a dropout layer is applied with a dropout rate of 0.2. Dropout serves as a regularization technique by randomly "dropping out" a fraction of units during training, which helps prevent overfitting and enhances the model's generalization ability.

Dense Output Layer: The final GRU layer is followed by a dense output layer with a single unit. This layer aims to produce the predicted value for the time series at the next time step.

Learning Rate Schedule

To optimize the model's weights during training, a learning rate schedule is employed. The learning rate is initialized at 0.1 and undergoes exponential decay. The decay rate is set to 1e-7, and the learning rate is adjusted every 1000 steps. This schedule aids in controlling the rate at which the model learns and can lead to more stable convergence during training.

Training Process

During training, the model is compiled using the mean squared error (MSE) loss function. The Stochastic Gradient Descent (SGD) optimizer is utilized with the aforementioned learning rate schedule. Additionally, early stopping with a minimum change (min_delta) of 0.001 and a patience of 10 epochs is implemented. Early stopping monitors the validation loss and restores the best model weights if the loss does not improve within the specified patience period.

The model is trained for 50 epochs with a batch size of 150 samples. Training progress is monitored via the loss convergence over epochs.

Prediction

Following training, the model is utilized to predict future values in a given test set (X_Test) using the model.predict() function. The resulting predictions are returned as pred_GRU.

This proposed GRU-based model presents a comprehensive approach to time series prediction, capitalizing on the capabilities of recurrent neural networks while incorporating dropout and a learning rate schedule to enhance generalization and convergence properties.

```
def GRU_model(X_Train, y_Train, X_Test):
    early_stopping = callbacks.EarlyStopping(min_delta=0.001,patience=10, restore_best_weights=True)

    #The GRU model
    model = Sequential()
    model.add(GRU(units=150, return_sequences=True, input_shape=(X_Train.shape[1],1), activation='tanh'))
    model.add(Dropout(0.2))
    model.add(GRU(units=50, return_sequences=True, input_shape=(X_Train.shape[1],1), activation='tanh'))
    model.add(Dropout(0.2))
    model.add(GRU(units=50, return_sequences=True, input_shape=(X_Train.shape[1],1), activation='tanh'))
    model.add(Dropout(0.2))
    model.add(Dense(units=1))

    learning_rate_schedule = keras.optimizers.schedules.ExponentialDecay(
    initial_learning_rate=0.1,
    decay_steps=1000,
    decay_rate=1e-7
    )

    callback = [early_stopping]
    model.compile(optimizer=keras.optimizers.SGD(learning_rate=learning_rate_schedule),loss='mean_squared_error')
    model.fit(X_Train,y_Train, epochs=50,batch_size=150)
    pred_GRU= model.predict(X_Test)
    return pred_GRU
```

5.1 High Level Diagram (if applicable)

Input
|
v

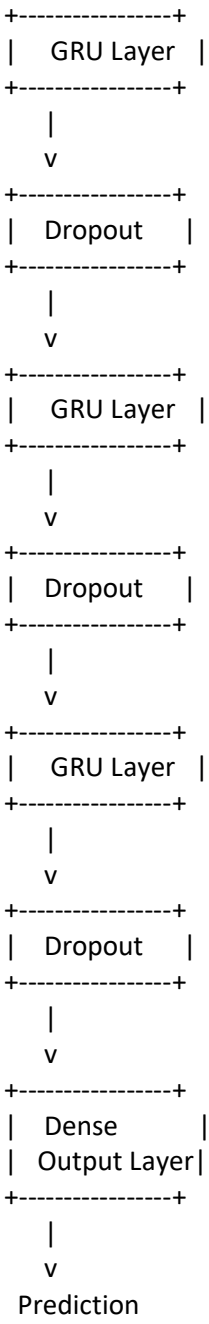


Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

6 Performance Test

In My Case there were no specific test constraints ,but the accuracy was the most important thing which I was suppose to achieve. And to check whether I have gain the accuracy or not I used Root Mean Square Error(RMSE).

Its Checks by taking the square root of the mean of the square of difference of the predicted value and value for the test.

```
def RMSE_Value(test,pred):
    sum = 0
    for i in range(0,len(test),1):
        sum += (pred[i] - test[i])**2
    sum = sum/len(test)

    return math.sqrt(sum)
```

These above code shows the code for calculating the root – mean square error of the predicted value.

6.1 Test Plan/ Test Cases

No any test case.

6.2 Test Procedure

There is not much in the test Procedure , it just RMSE that we are finding in the end to check if the output are perfect or not.

6.3 Performance Outcome

```
Epoch 39/50
48/48 [==============================] - 22s 468ms/step - loss: 2.2832
Epoch 40/50
48/48 [==============================] - 24s 486ms/step - loss: 2.2832
Epoch 41/50
48/48 [==============================] - 22s 448ms/step - loss: 2.2832
Epoch 42/50
48/48 [==============================] - 23s 486ms/step - loss: 2.2832
Epoch 43/50
48/48 [==============================] - 25s 514ms/step - loss: 2.2832
Epoch 44/50
48/48 [==============================] - 31s 650ms/step - loss: 2.2832
Epoch 45/50
48/48 [==============================] - 23s 476ms/step - loss: 2.2832
Epoch 46/50
48/48 [==============================] - 23s 479ms/step - loss: 2.2833
Epoch 47/50
48/48 [==============================] - 23s 471ms/step - loss: 2.2832
Epoch 48/50
48/48 [==============================] - 21s 437ms/step - loss: 2.2833
Epoch 49/50
48/48 [==============================] - 27s 564ms/step - loss: 2.2832
Epoch 50/50
48/48 [==============================] - 26s 551ms/step - loss: 2.2832
225/225 [==============================] - 10s 36ms/step
Root-Mean Square Error: 1.4902841302207581
```

7 My learnings

These was the great experience working with in these intership. As these was my first project I was pretty exited . Firstly, I had to do lot of research in these project as Data provided was time series data , I have very less experience working with time-series data and working with the predictive model. These was a challenge as well as intership program for me.

With these I learn many implemented which I had only studied in the theory. With these intership program , I came aware of proper time management skills which are mandatory in real life project.

I was able to study further in Time-Series Analysis and Neural Network and also do practical implementation which is a good thing for me and my career.

8 Future work scope

As these was my first time in a project like these , these project has not met all the needs which it was suppose to met.
Still there are some problem which I am facing while predicting the value are inappropriate value which I have added in the future task as I have to do more research in these area and find the optimal solution to solve these problem.
Secondly, I am also thinking to apply new model and do comparision with these model . Next , model probabily will be KNN which I think is also appropriate in these type of problems.

Concluding my report here on my learning , I would like to thank all the member of Upskill Campus and Uni-Convergence Technology Who had been great help to me during these program.