

# 停車位預約系統流程改善

## 第三組

- 112034553 黃柏欽
- 112034560 劉倫華
- 112034565 黃琬芹
- 112034566 洪庭緣

# CONTENTS

1

背景介紹

2

研究方法

3

模型建立

4

網頁設計與後  
臺管理系統

5

結論

08:00



# 1. 背景介紹



# 背景介紹

“

在清華大學停車場，以往車輛主要依賴人工尋找車位，這種方式效率低且容易導致停車時間延長，在缺乏有效管理系統的情況下，這不僅耗費了大量的時間，也增加了停車場的混亂程度。尤其在高峰時段，車輛難以找到合適位置停放，導致塞車及浪費時間，並影響了校園內的交通流暢度和生活品質。

”

# 5W1H

## WHAT

預測停車場的佔用率



## WHERE

清大校園內4處停車場



## WHY

無法準確確認未來停車空位



## WHEN

平日上班上課期間



## WHO

清大停車場用戶



## HOW

比較3種模型，確認預測準確率



08:00



## 2. 研究方法



# 模型



“

LSTM



“

BiLSTM



“

TSD-GRU



# LSTM

## 模型結構

長短期記憶 ( LSTM ) :

- 特殊的循環神經網絡 ( RNN ) ，解決梯度消失問題。
- 通過引入門控機制，捕捉和保持序列中的長期相關性。

## 工作流程

細胞狀態 ( Cell State ) :

- 記住長期信息，提供穩定的記憶能力。

隱藏狀態 ( Hidden State ) :

- 根據當前輸入和前一隱藏狀態計算預測結果。

訓練過程:

- 根據當前輸入和目標輸出計算預測值。
- 比較預測值和真實值的誤差來更新模型權重。

## 優勢

- 長期記憶：克服RNN的梯度消失問題，保留長期依賴信息。
- 準確預測：通過多層結構和門控機制，提升預測準確性。

# BiLSTM

## 模型結構

### 雙向傳播機制：

- 同時捕獲過去和未來的上下文信息。

## 工作流程

### 前向LSTM層 (Forward LSTM)

- 接收輸入序列，按照時間順序（從前到後）處理。
- 每個時間步產生一個隱藏狀態。

### 後向LSTM層 (Backward LSTM)

- 接收輸入序列，按照時間逆序（從後到前）處理。
- 每個時間步產生一個隱藏狀態。

### 隱藏狀態合併

- 將前向和後向的隱藏狀態進行連接。合併後的隱藏狀態傳遞到輸出層，進行預測或分類任務。

## 優勢

- 上下文理解：結合前向和後向的上下文信息，增強序列數據的理解能力。
- 強大工具：適用於處理複雜序列任務，如自然語言處理和時間序列預測。

# TSD-GRU

## 工作流程

1. 數據分解：使用TSD分解數據。
2. 特徵處理：將分解後的特徵輸入GRU模型。
3. 預測：GRU模型預測未來值。

## 時間序列分解 (TSD)

- ❑ 將停車佔用率數據分解為趨勢、季節性和殘差分量。
- ❑ 幫助理解數據的基本模式和變化。

## 優點

- 準確性高：通過理解數據基礎結構後再進行預測。
- 高效訓練：GRU模型參數少，訓練速度快，資源使用效率高。

## 門控迴圈單元 (GRU)

- ❑ 使用GRU模型處理趨勢、季節性和殘差特徵。
- ❑ GRU透過更新和重置門克服梯度消失問題，比RNN更有效。
- ❑ 與LSTM相比，GRU參數更少，訓練速度更快，資源使用更有效。

08:00



# 3. 模型建立

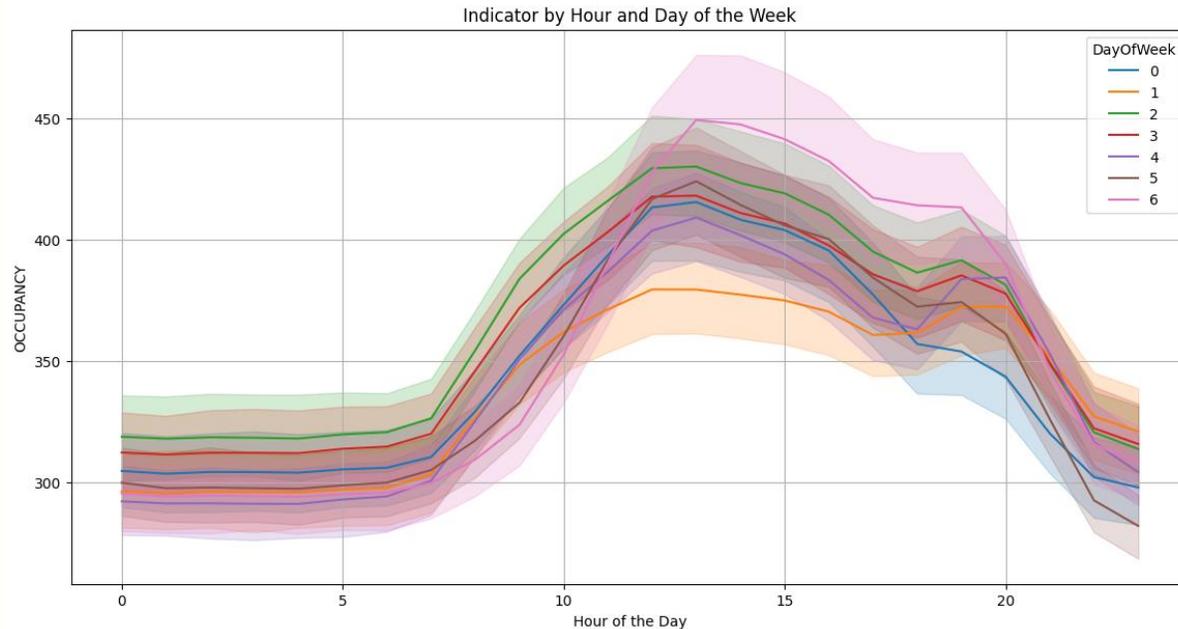


# 訓練數據集介紹

	TIME	OCCUPANCY	RATE	INDICATOR	FS
0	2018/6/1 0:00	200	0.200	0.279856	0.255971
1	2018/6/1 0:05	200	0.200	0.277484	0.255497
2	2018/6/1 0:10	200	0.200	0.276493	0.255299
3	2018/6/1 0:15	202	0.202	0.264126	0.255354
4	2018/6/1 0:20	203	0.203	0.259295	0.255637

1. TIME：時間。
2. OCCUPANCY：停車場的汽車數量。
3. RATE：停車位佔用率。
4. INDICATOR：根據平日假日、天氣狀況、交通流量等特徵計算出的值。
5. FS：利用Fourier Transform將時間域中的數據轉換到頻率域中。

# INDICATOR計算



- 一天中的時間：工作時間內佔用率較高。
- 一周中的日子：工作日的佔用率較高。
- 天氣狀況：不良天氣可能增加佔用率。
- 地點：不同地點的佔用模式不同。
- 事件和季節變化：特定事件和季節變化會影響佔用率。

公式：INDICATOR=基礎值+(時間權重×時間)+(日期權重×一周中的日子)+(天氣權重×天氣狀況)+地點調整+事件調整

# 模型建立-LSTM

```
# Define hyperparameters for the LSTM model
LOOK_BACK = 3
INPUT_FEATURES_NUM = LOOK_BACK
HIDDEN_SIZE = 8
OUTPUT_FEATURES_NUM = 1
NUM_LAYERS = 1
max_epochs = 4000
LEARNING_RATE = 0.02
circle = 10
```

LSTM定義超參數

# 模型建立-LSTM

```
# Define the LSTM Neural Network
class LSTMModel(nn.Module) :

    def __init__(self, input_size, hidden_size, output_size, num_layers):
        super().__init__()
        self.lstm = nn.LSTM(input_size, hidden_size, num_layers) # utilize the GRU model in torch.nn
        self.forwardCalculation = nn.Linear(hidden_size, output_size)

    def forward(self, _x):
        x, _ = self.lstm(_x) # _x is input, size (seq_len, batch, input_size)
        s, b, h = x.shape # x is output, size (seq_len, batch, hidden_size)
        x = x.view(s * b, h)
        x = self.forwardCalculation(x)
        x = x.view(s, b, -1)
        return x

# Initialize the LSTM model
gru_model = LSTMModel(input_size=INPUT_FEATURES_NUM, hidden_size=HIDDEN_SIZE, output_size=OUTPUT_FEATURES_NUM, num_layers=NUM_LAYERS)
loss_function = nn.MSELoss()
optimizer = torch.optim.Adam(gru_model.parameters(), lr=LEARNING_RATE)
```

設置了 LSTM 層和全連接層，LSTM 層接受輸入特徵大小、隱藏單元數量、輸出大小和 LSTM 層數。

# 模型建立-BiLSTM

```
# Define hyperparameters for the model
LOOK_BACK = 9
INPUT_FEATURES_NUM = LOOK_BACK
HIDDEN_SIZE = 8
OUTPUT_FEATURES_NUM = 1
NUM_LAYERS = 1
max_epochs = 500
LEARNING_RATE = 0.02
circle = 5
```

BiLSTM定義超參數

# 模型建立-BiLSTM

```
# Define a Bidirectional LSTM (BiLSTM) model
class BiLSTMModel(nn.Module) :
    def __init__(self, input_size, hidden_size, output_size, num_layers):
        super().__init__()
        self.lstm = nn.LSTM(input_size, hidden_size, num_layers, bidirectional=True) # utilize the GRU model in torch.nn
        self.forwardCalculation = nn.Linear(hidden_size*2, output_size)

    def forward(self, _x):
        x, _ = self.lstm(_x) # _x is input, size (seq_len, batch, input_size)
        s, b, h = x.shape # x is output, size (seq_len, batch, hidden_size)
        x = x.view(s * b, h)
        x = self.forwardCalculation(x)
        x = x.view(s, b, -1)
        return x

# Initialize the BiLSTM model
gru_model = BiLSTMModel(input_size=INPUT_FEATURES_NUM, hidden_size=HIDDEN_SIZE, output_size=OUTPUT_FEATURES_NUM, num_layers=NUM_LAYERS)
for name, param in gru_model.named_parameters():
    if param.requires_grad:
        print(param.shape)

loss_function = nn.MSELoss()
optimizer = torch.optim.Adam(gru_model.parameters()), lr=LEARNING_RATE)
```

設置了雙向 LSTM 層和全連接層。雙向 LSTM 層接受輸入特徵大小、隱藏單元數量、輸出大小和 LSTM 層數。

# 模型建立-TSD-GRU(TSD-preparation )

```
# trend feature
def create_interval_dataset(dataset, look_back):
    dataX, dataY = [], []
    for i in range(len(dataset) - look_back):
        dataX.append(dataset[i])
        dataY.append(dataset[i+look_back])
    return np.asarray(dataX), np.asarray(dataY)

# cycle feature
def create_cycle_dataset(dataset, look_back):
    dataX, dataY = [], []
    for i in range(len(dataset) - look_back * 2):
        dataX.append(dataset[i:i + look_back])
        dataY.append(dataset[i + look_back:i + look_back + look_back])
    return np.asarray(dataX), np.asarray(dataY)

# RNN parameters
def create_RNNs_dataset(dataset, look_back):
    dataX, dataY = [], []
    for i in range(len(dataset) - look_back * 2):
        dataX.append(dataset[i:i + look_back])
        dataY.append(dataset[i + look_back + look_back])
    return np.asarray(dataX), np.asarray(dataY)

# import time-series-decomposition dataset
def import_TSD_data():
    file_names = [
        'Commercial1.csv', 'Commercial2.csv', 'Commercial3.csv', 'Commercial4.csv', 'Commercial5.csv',
        'Commercial6.csv', 'Commercial7.csv', 'Commercial8.csv', 'Commercial9.csv', 'Commercial10.csv',
```

定義了import\_data，建立training sets 和test sets，趨勢特徵、循環特徵以及時間序列的數據集，這些函數主要用於數據的準備工作。

# 模型建立-TSD-GRU(Customize GRU )

```

# input data
Trend, Cycle, Effect, train_size, test_size = import_TSD_data()

# Hyperparameters
LOOK_BACK = 6
INPUT_FEATURES_NUM = 2*LOOK_BACK + 1
HIDDEN_SIZE = 8
OUTPUT_FEATURES_NUM = 1
NUM_LAYERS = 1
max_epochs = 4000
LEARNING_RATE = 0.02
circle = 5

# GRU Neural Networks
class GRU_model(nn.Module):
    def __init__(self, input_size, hidden_size, output_size, num_layers):
        super().__init__()
        self.gru = nn.GRU((input_size - 1), hidden_size, num_layers)
        self.forwardProcessing = nn.Linear((hidden_size + 1), output_size)

    # override initial forward
    def forward(self, x_input):
        # x_input is the input matrix
        # size (seq length, batch, input_size)
        x = x_input[:, :, 0:(INPUT_FEATURES_NUM-1)]
        e = x_input[:, :, (INPUT_FEATURES_NUM-1)].reshape(1, x_input.shape[1], 1)
        x, _ = self.gru(x)
        x = torch.cat([x, e], dim=2)
        s, b, h = x.shape
        x = x.view(s * b, h)
        x = self.forwardProcessing(x)
        x = x.view(s, b, -1)
        return x

```

建立和訓練GRU模型基本框架和參數設置，用於處理時間序列數據。序列的每個元素都包含了一組特徵，首先通過GRU層處理了序列的特徵部分，生成了隱藏狀態。然後，這個隱藏狀態與效應特徵部分進行拼接，形成了新的特徵矩陣，並通過線性層進行處理，最終生成預測結果。

# 模型建立-TSD-GRU(TSD-GRU model creation)

```

from tqdm import tqdm
# TSD-GRU formulation
def TSD_GRU(gru_model, trend, cycle, effect):

    # define training and testing sets after applying Time-Series-Decomposition
    train_trend = trend[:train_size]
    test_trend = trend[train_size:]
    train_cycle = cycle[:train_size]
    test_cycle = cycle[train_size:]
    effect = effect.reshape(-1, 1)
    train_effect = effect[LOOK_BACK+LOOK_BACK:train_size]
    test_effect = effect[LOOK_BACK+LOOK_BACK+train_size:]

    # create training set
    train_trend, train_y = create_RNNs_dataset(train_trend, look_back=LOOK_BACK)
    _, train_cycle = create_cycle_dataset(train_cycle, look_back=LOOK_BACK)

    train_x = np.concatenate((train_trend, train_cycle, train_cycle), axis = 1)
    train_x = torch.tensor(train_x, dtype=torch.float32)
    train_y = torch.tensor(train_y, dtype=torch.float32)

    batch_size = train_x.shape[0]

    input_feature = train_x.shape[1]
    train_x_tensor = train_x.reshape(1, batch_size, input_feature)
    train_x_tensor = torch.tensor(train_x_tensor, dtype=torch.float32)
    train_y_tensor = train_y.reshape(1, batch_size, OUTPUT_FEATURES_NUM)
    train_y_tensor = torch.tensor(train_y_tensor, dtype=torch.float32)

    # create testing sets
    test_trend, test_y = create_RNNs_dataset(test_trend, look_back=LOOK_BACK)

```

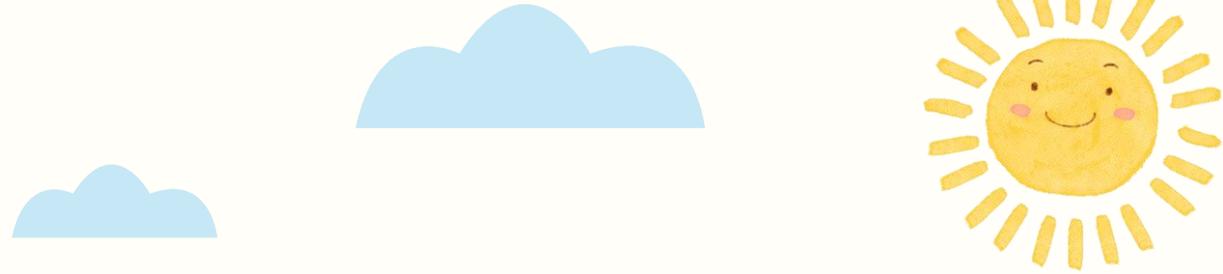
它將數據集劃分為訓練集和測試集，然後對訓練數據進行TSD處理，提取出趨勢、周期和效應特徵，用於GRU模型的訓練。在模型的訓練過程中，它通過多個循環進行訓練，每個循環中都隨機初始化模型的參數。

# 結果比較



Model	MAPE(%)	RMSE(x10 <sup>-2</sup> )	RAE(%)	R <sup>2</sup> (%)
LSTM	3.54	1.69	15.6	93.3
BiLSTM	5.00	3.74	21.4	94.1
★ TSD-GRU	2.33	1.66	12.35	96.4

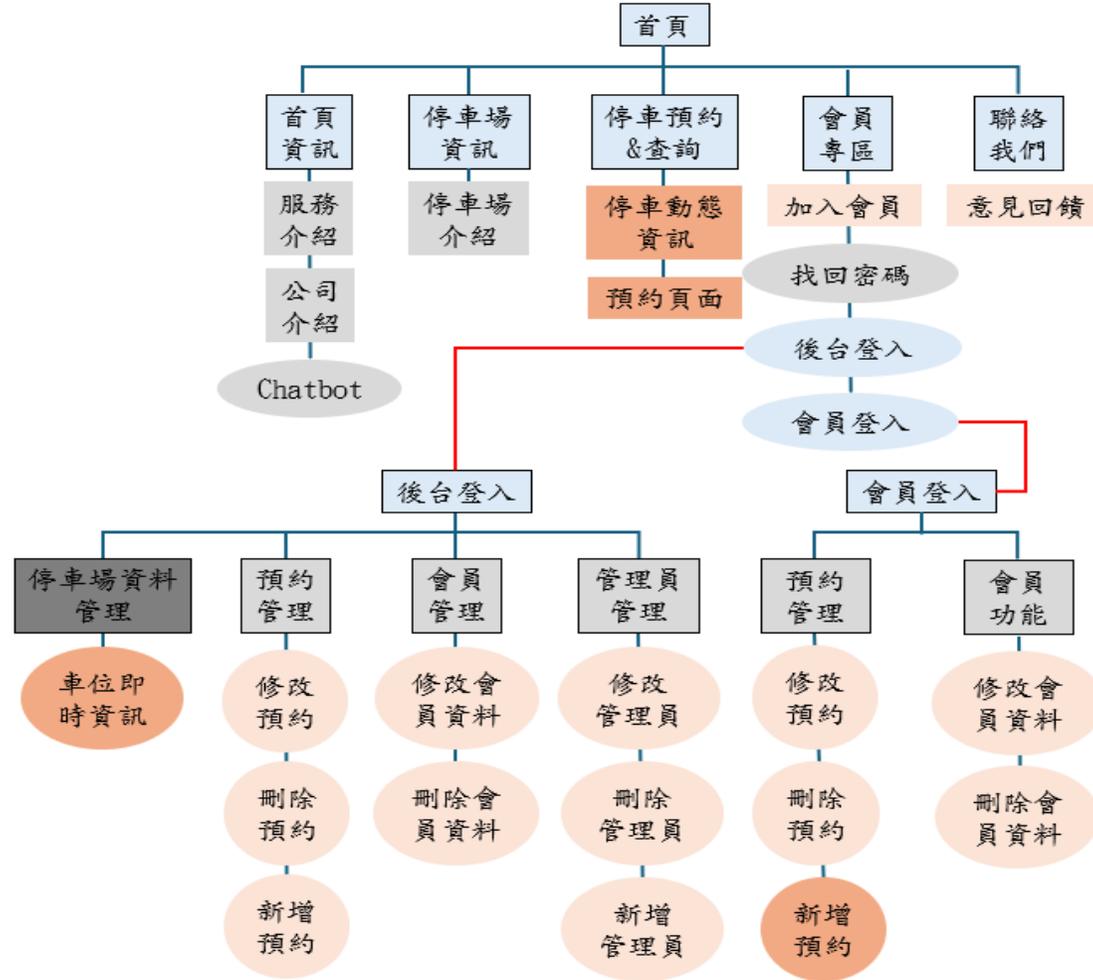
08:00



# 4. 網頁設計與後臺管理系統



# Web架構圖



# ER-Model

BOOKING
<u>Order no</u> 
<u>Time date</u>
Time
<u>Select park</u>
Total 
Hour
ID
License

MEMBER
Account 
Password
Email
ID 
Telephone
Name
Membership category
Category

PARKING LOT
Park 
Num
Booked quantity
Forecasted bookings

MANAGER
Account 
Password
Email
ID 
Telephone
Name

08:00



# 功能介紹

普通預約：[http://140.114.54.94/IIE\\_2024/group3/Final/index.html](http://140.114.54.94/IIE_2024/group3/Final/index.html)

會員預約：[http://140.114.54.94/IIE\\_2024/group3/Final/booking\\_member.html](http://140.114.54.94/IIE_2024/group3/Final/booking_member.html)

後台介面：

[http://140.114.54.94/IIE\\_2024/group3/Final/%E5%BE%8C%E5%8F%B0%E7%B3%BB%E7%B5%B1/tables4.html](http://140.114.54.94/IIE_2024/group3/Final/%E5%BE%8C%E5%8F%B0%E7%B3%BB%E7%B5%B1/tables4.html)



# 功能介紹 - 網路預約及查詢



## 剩餘車位數即時資訊

若停車場車位容量未滿，則可進行停車位之預約活動

<input checked="" type="radio"/> 機車		<input type="radio"/> 汽車
行政大樓周邊停車場	剩餘預約量 20	<a href="#">預約</a>
工程一館旁邊停車場	剩餘預約量 10	<a href="#">預約</a>
風雲樓後方停車場	剩餘預約量 0	暫無空位
大禮堂前面停車場	剩餘預約量 10	<a href="#">預約</a>
排球場旁邊停車場	剩餘預約量 30	<a href="#">預約</a>

即時停車場資訊介面

## 車位預約

請填寫基本資料

### 填寫個人資訊

身份  學生  教職員  一般旅客

姓名  電話  車牌

日期  時間

[訂票確認](#)

車位預約介面

08:00

# 功能介紹 - 會員專區



## 填寫個人資訊

身份  學生  教職員  一般旅客

Lab724

停車系統預約

1小時

停車系統預約

選擇日期:

(+08:00) CST - Taipei

< 六月 >

週一 週二 週三 週四 週五 週六 週日

27 28 29 30 31 1 2

3 4 5 6 7 8 9

10 11 12 13 14 15 16

17 18 19 20 21 22 23

24 25 26 27 28 29 30

1 2 3 4 5 6 7

選擇時間:



8:00 am

8:15 am

8:30 am

8:45 am

9:00 am

9:15 am

9:30 am

9:45 am

10:00 am

10:15 am

10:30 am



08:00

# 功能介紹 - 會員專區

TidyCal

確認預約:

---

停車系統預約 與 Lab724

---

📅 星期六, 六月 22日, 在 9:45 上午

---

🌐 Asia/Taipei

🕒 1小時

---

您的姓名:  您的Email:

車牌

停車場

Park1 : 行政大樓周邊

Park2 : 大禮堂前

Park3 : 風雲樓後方

Park4 : 工程一館旁邊

This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply.

會員預約資料填寫



## Booking Confirmation

Hello, qwe!  
Your booking has been confirmed!

### 停車系統預約

With:  
洪庭緣

Date:  
June 13, 2024

Time:  
9:00 AM

Timezone:  
Asia/Taipei

預約成功確認

# 5. 結論



## “ 結論

- 經過比較LSTM、BiLSTM和TSD-GRU三種模型的預測結果，我們發現TSD-GRU模型在準確性上具有顯著優勢
- 本研究成功地利用TSD-GRU模型改善了清華大學校園停車位的預測和管理系統
- 通過應用該模型，我們可以更精確地預測停車場的佔用率，使得用戶在停車時能夠更高效地找到空位，縮短尋找停車位的時間，從而減少校園內的交通擁堵

## “ 未來展望

未來，我們計劃進一步完善和擴展停車位預約系統，以提供更全面的服務。引入即時停車位影像辨識系統，結合影像辨識技術實時更新停車位狀態，提供更加準確和動態的停車資訊。除了現有的線上繳費系統，我們將增加更多支付方式，如行動支付、電子錢包等，讓用戶在繳費時有更多選擇。透過這些改進措施，我們期望能夠進一步提升停車場管理的智慧化水平，為用戶提供更加便捷、高效的停車服務，最終實現校園交通管理的全面優化。

08:00



THANK YOU!