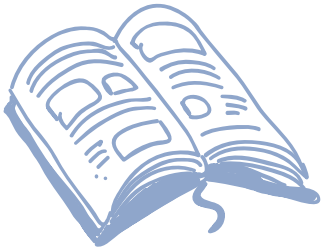
Deep Learning Method for Traffic Flow Prediction

110034535黃于瑞

Dr. Ming-Chuan Chiu

Presenter: 黃于瑞

2022/01/07



Outline

1. Introduction
2. Method
3. Experiment
4. Result
5. Contribution
6. Reflection

Outline

1. Introduction
2. Method
3. Experiment
4. Result
5. Contribution
6. Reflection

國立清華大學 National Tsing Hua University

INNOVATION ENGINEERING LAB

Introduction

Why
Accurate and real-time traffic flow prediction is important

How
Deep Learning Method

When
Make decision

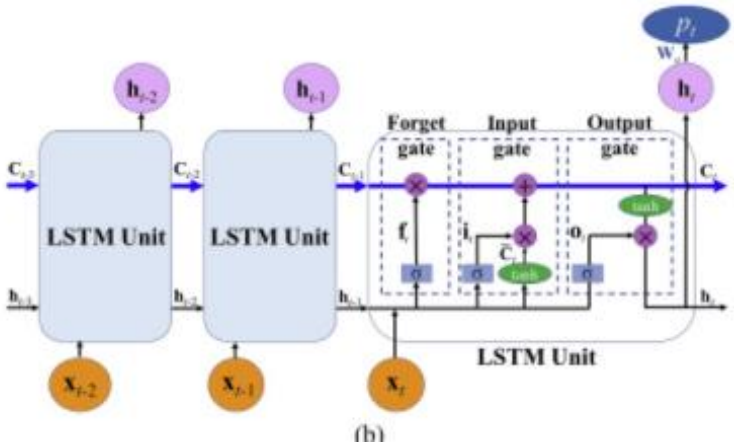
Where
Free way or road

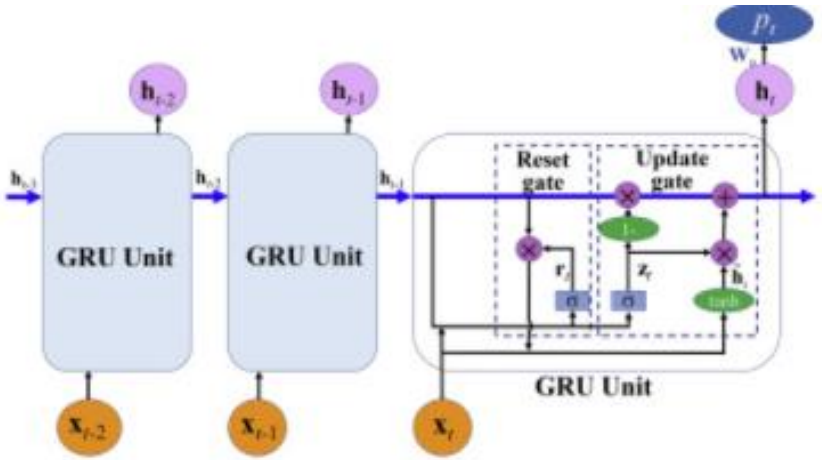
Who
Government

What
Intelligent transportation management and control.

Introduction

- In the past, most of the papers are focus LSTM, the project can show that how variants of RNNs help the traffic prediction.



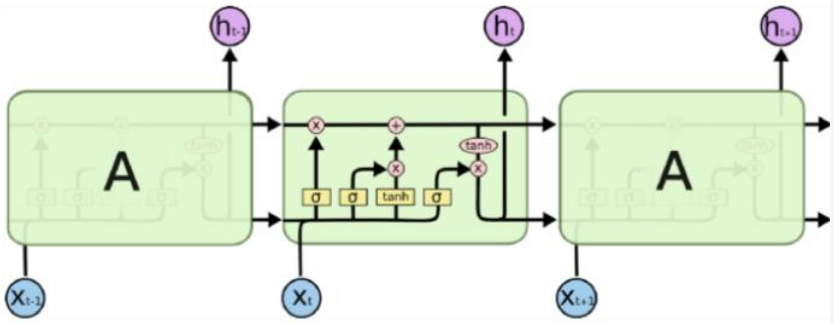


Outline

1. Introduction
2. Method
3. Experiment
4. Result
5. Contribution
6. Reflection

Method

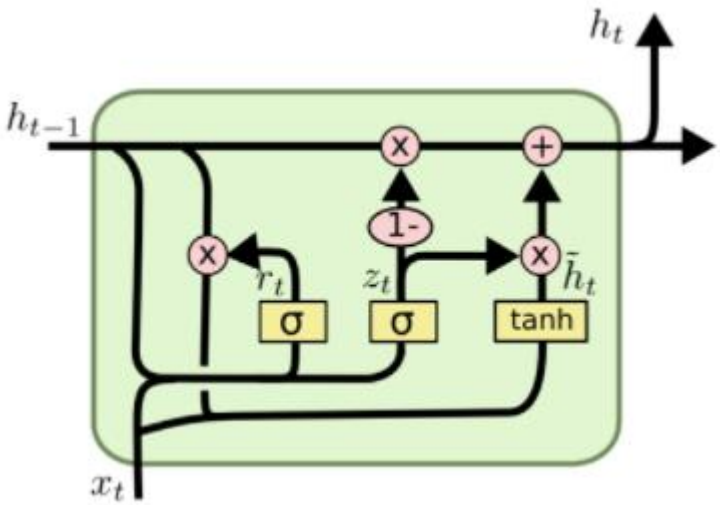
```
from keras.layers import Dense, Dropout, Activation
from keras.layers.recurrent import LSTM, GRU
from keras.models import Sequential
```




```
def get_lstm(units):
    """LSTM(Long Short-Term Memory)
    Build LSTM Model.

    # Arguments
        units: List(int), number of input, output and hidden units.
    # Returns
        model: Model, nn model.
    """

    model = Sequential()
    model.add(LSTM(units[1], input_shape=(units[0], 1), return_sequences=True))
    model.add(LSTM(units[2]))
    model.add(Dropout(0.2))
    model.add(Dense(units[3], activation='sigmoid'))

    return model
```

Method

- GRU



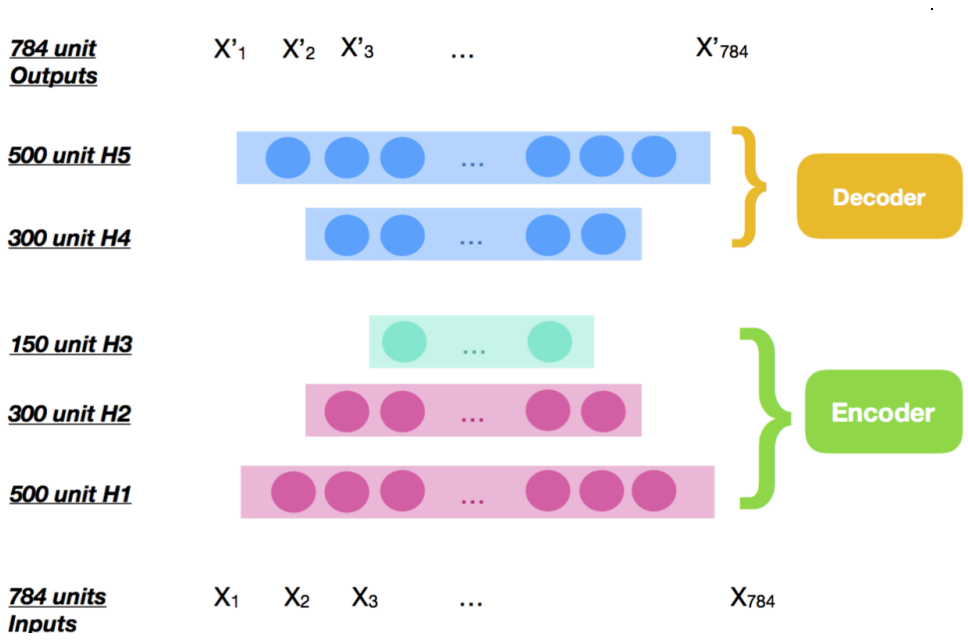

```
from keras.layers import Dense, Dropout, Activation
from keras.layers.recurrent import LSTM, GRU
from keras.models import Sequential
```

```
def get_gru(units):
    """GRU(Gated Recurrent Unit)
    Build GRU Model.

    # Arguments
        units: List(int), number of input, output and hidden units.
    # Returns
        model: Model, nn model.
    """

    model = Sequential()
    model.add(GRU(units[1], input_shape=(units[0], 1), return_sequences=True))
    model.add(GRU(units[2]))
    model.add(Dropout(0.2))
    model.add(Dense(units[3], activation='sigmoid'))

    return model
```

Method

- SAES

SAE model with proposed dropout method is presented, consisting layers of autoencoders with deep learning-based structure.




```python
def get_saes(layers):
    """SAEs(Stacked Auto-Encoders)
    Build SAEs Model.

    # Arguments
        layers: List(int), number of input, output and hidden units.
    # Returns
        models: List(Model), List of SAE and SAEs.
    """
    sae1 = _get_sae(layers[0], layers[1], layers[-1])
    sae2 = _get_sae(layers[1], layers[2], layers[-1])
    sae3 = _get_sae(layers[2], layers[3], layers[-1])

    saes = Sequential()
    saes.add(Dense(layers[1], input_dim=layers[0], name='hidden1'))
    saes.add(Activation('sigmoid'))
    saes.add(Dense(layers[2], name='hidden2'))
    saes.add(Activation('sigmoid'))
    saes.add(Dense(layers[3], name='hidden3'))
    saes.add(Activation('sigmoid'))
    saes.add(Dropout(0.2))
    saes.add(Dense(layers[4], activation='sigmoid'))

    models = [sae1, sae2, sae3, saes]
```

Outline

1. Introduction
2. Method
3. Experiment
4. Result
5. Contribution
6. Reflection

Experiment

- Datasets: Performance Measurement System (PeMS). Data are collected in real-time from individual detectors spanning the freeway system across all major metropolitan areas of the State of California

Experiment

- Train：7777 sample points
- Test：3332 sample points

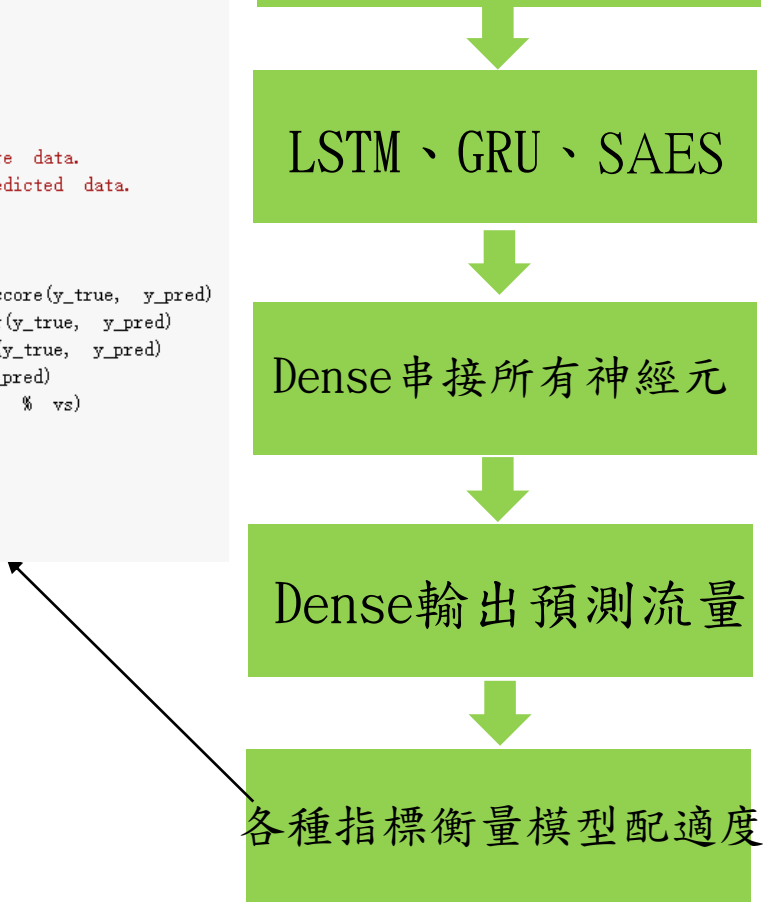
7769	29/02/2016	13	1	100
7770	29/02/2016	19	1	100
7771	29/02/2016	17	1	100
7772	29/02/2016	13	1	100
7773	29/02/2016	19	1	100
7774	29/02/2016	6	1	100
7775	29/02/2016	14	1	100
7776	29/02/2016	11	1	100
7777	29/02/2016	10	1	100

3324	21/03/2016	105	1	100
3325	21/03/2016	107	1	100
3326	21/03/2016	92	1	100
3327	21/03/2016	96	1	100
3328	21/03/2016	90	1	100
3329	21/03/2016	110	1	100
3330	21/03/2016	98	1	100
3331	21/03/2016	90	1	100
3332	21/03/2016	103	1	100
3333				

```
Processing the data
"""
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler, MinMaxScaler


def process_data(train, test, lags):
    """Process data
    Reshape and split train\test data.

    # Arguments
            train: String, name of .csv train file.
            test: String, name of .csv test file.
            lags: integer, time lag.
    # Returns
            X_train: ndarray.
            y_train: ndarray.
            X_test: ndarray.
            y_test: ndarray.
            scaler: StandardScaler.
    """
    attr = 'Lane 1 Flow (Veh/5 Minutes)'
    df1 = pd.read_csv(train, encoding='utf-8').fillna(0)
    df2 = pd.read_csv(test, encoding='utf-8').fillna(0)

    # scaler = StandardScaler().fit(df1[attr].values)
    scaler = MinMaxScaler(feature_range=(0, 1)).fit(df1[attr].values.r
    flow1 = scaler.transform(df1[attr].values.reshape(-1, 1)).reshape(
    flow2 = scaler.transform(df2[attr].values.reshape(-1, 1)).reshape(

    train, test = [], []
    for i in range(lags, len(flow1)):
            train.append(flow1[i - lags: i + 1])
    for i in range(lags, len(flow2)):
            test.append(flow2[i - lags: i + 1])
```

Experiment

```
def eva_regress(y_true, y_pred):
    """Evaluation
    evaluate the predicted resul.

    # Arguments
        y_true: List/ndarray, ture data.
        y_pred: List/ndarray, predicted data.
    """

    mape = MAPE(y_true, y_pred)
    vs = metrics.explained_variance_score(y_true, y_pred)
    mae = metrics.mean_absolute_error(y_true, y_pred)
    mse = metrics.mean_squared_error(y_true, y_pred)
    r2 = metrics.r2_score(y_true, y_pred)
    print('explained_variance_score:%f' % vs)
    print('mape:%f%%' % mape)
    print('mae:%f' % mae)
    print('mse:%f' % mse)
    print('rmse:%f' % math.sqrt(mse))
    print('r2:%f' % r2)
```



Experiment

- Experiment design-LSTM

	Dropout	Batch size	Activate function	Epoch
LEVEL 1	0.2	128	tanh	500
LEVEL 2	0.3	256	Sigmoid	600
LEVEL 3	0.4	512	Relu	700

Experiment

- Orthogonal Array-LSTM

Experiment	Dropout	Batch size	Activate function	Epoch
1	0.2	128	tanh	500
2	0.2	256	Sigmoid	600
3	0.2	512	Relu	700
4	0.3	128	Relu	700
5	0.3	256	Sigmiod	600
6	0.3	512	Tanh	500
7	0.4	128	Sigmoid	600
8	0.4	256	Relu	700
9	0.4	512	Tanh	500

Experiment

Experiment	Dropout	Batch size	Activate function	Epoch	R-square	Mse
1	0.2	128	tanh	500	0.9218	100.562
2	0.2	256	Sigmoid	600	0.9399	98.929
3	0.2	512	Relu	700	0.9152	99.151
4	0.3	128	Relu	700	0.9341	101.256
5	0.3	256	Sigmiod	600	0.9298	99.076
6	0.3	512	Tanh	500	0.9054	102.364
7	0.4	128	Sigmoid	600	0.9263	103.456
8	0.4	256	Relu	700	0.9312	100.421
9	0.4	512	Tanh	500	0.9255	98.994

Experiment

- Experiment design-GRU

	Dropout	Batch size	Activate function	Epoch
LEVEL 1	0.2	128	tanh	500
LEVEL 2	0.3	256	Sigmoid	600
LEVEL 3	0.4	512	Relu	700

Experiment

- Orthogonal Array-GRU

Experiment	Dropout	Batch size	Activate function	Epoch
1	0.2	128	tanh	500
2	0.2	256	Sigmoid	600
3	0.2	512	Relu	700
4	0.3	128	Relu	700
5	0.3	256	Sigmiod	600
6	0.3	512	Tanh	500
7	0.4	128	Sigmoid	600
8	0.4	256	Relu	700
9	0.4	512	Tanh	500

Experiment

Experiment	Dropout	Batch size	Activate function	Epoch	R-square	Mse
1	0.2	128	tanh	500	0.9185	106.738
2	0.2	256	Sigmoid	600	0.9214	105.047
3	0.2	512	Relu	700	0.9296	108.269
4	0.3	128	Relu	700	0.9136	104.902
5	0.3	256	Sigmiod	600	0.9363	104.857
6	0.3	512	Tanh	500	0.8917	106.216
7	0.4	128	Sigmoid	600	0.9343	110.956
8	0.4	256	Relu	700	0.9211	105.238
9	0.4	512	Tanh	500	0.9193	112.483

Experiment

- Experiment design-SAES

	Dropout	Batch size	Activate function	Epoch
LEVEL 1	0.2	128	tanh	500
LEVEL 2	0.3	256	Sigmoid	600
LEVEL 3	0.4	512	Relu	700

Experiment

- Orthogonal Array-SAES

Experiment	Dropout	Batch size	Activate function	Epoch
1	0.2	128	tanh	500
2	0.2	256	Sigmoid	600
3	0.2	512	Relu	700
4	0.3	128	Relu	700
5	0.3	256	Sigmiod	600
6	0.3	512	Tanh	500
7	0.4	128	Sigmoid	600
8	0.4	256	Relu	700
9	0.4	512	Tanh	500

Experiment

Experiment	Dropout	Batch size	Activate function	Epoch	R-square	Mse
1	0.2	128	tanh	500	0.9137	102.46
2	0.2	256	Sigmoid	600	0.9402	100.128
3	0.2	512	Relu	700	0.921	103.574
4	0.3	128	Relu	700	0.8954	96.481
5	0.3	256	Sigmiod	600	0.9264	99.962
6	0.3	512	Tanh	500	0.9357	101.548
7	0.4	128	Sigmoid	600	0.9394	97.146
8	0.4	256	Relu	700	0.9422	95.088
9	0.4	512	Tanh	500	0.9069	98.452

Outline

1. Introduction
2. Method
3. Experiment
4. Result
5. Contribution
6. Reflection

Result

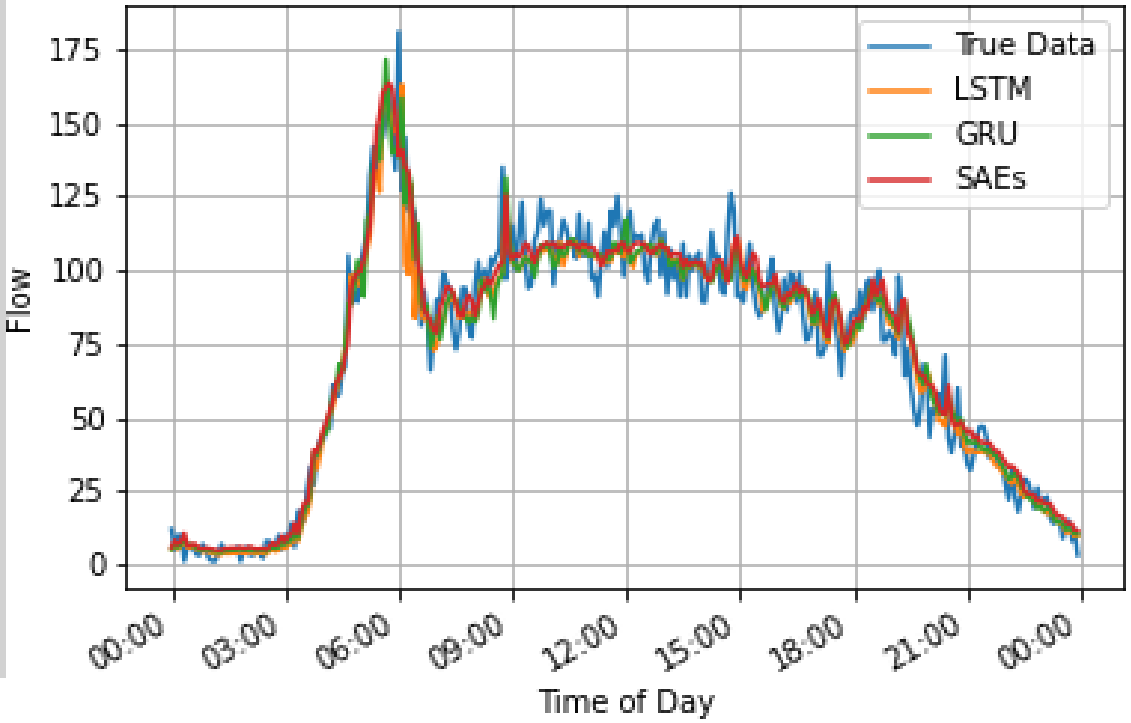
	Dropout	Batch size	Activate function	Epoch
LSTM	0.2	256	Sigmoid	600
GRU	0.3	256	Sigmiod	600
SAES	0.4	256	Relu	700

Metrics	MAE	MSE	RMSE	MAPE	R-squre	Explaind variance score
LSTM	7.17	98.92	9.94	16.91%	0.9399	0.94
GRU	7.31	104.85	10.24	18.05%	0.9363	0.9363
SAEs	7.14	95.08	9.75	20.26%	0.9422	0.9439

Result

```
LSTM
execution time: 1.3946423530578613
explained_variance_score:0.940020
mape:16.913820%
mae:7.172698
mse:98.929231
rmse:9.946317
r2:0.939904
GRU
execution time: 0.8498928546905518
explained_variance_score:0.936315
mape:18.054205%
mae:7.319282
mse:104.857864
rmse:10.240013
r2:0.936303
SAEs
execution time: 1.4183828830718994
explained_variance_score:0.943919
mape:20.260020%
mae:7.147675
mse:95.088501
rmse:9.751333
r2:0.942237
```

```
end = time.time()
sec = end - start
print(name)
print('execution time:',sec)
eva_regress(y_test, predicted)
```



Outline

1. Introduction
2. Method
3. Experiment
4. Result
5. Contribution
6. Reflection

Contribution

- Precise short-term traffic flow prediction is vital for intelligent transportation systems.
- Show that GRU can get similar result in less execution time.

Outline

1. Introduction
2. Method
3. Experiment
4. Result
5. Contribution
6. Reflection

Reflection

- Using traffic flow alone as input variables can yield the acceptable performance of traffic flow prediction, while using traffic flow together with speed as inputs may yield better results than using traffic flow alone.

- Why SAES is better than LSTM and GRU ?

國立清華大學 National Tsing Hua University

INNOVATION ENGINEERING LAB

Reference

- https://ieeexplore.ieee.org/abstract/document/8317872
- https://github.com/xiaochus/TrafficFlowPrediction
- https://ieeexplore.ieee.org/document/6894591
- https://www.twblogs.net/a/5b8e34a32b717718834380e5
- https://ir.nctu.edu.tw/handle/11536/45372?locale=zh_TW
- https://kknews.cc/zh-tw/code/rnja46v.html

國立清華大學
National Tsing Hua University

INNOVATION ENGINEERING LAB

Thank You for Your Listening