LSTM 전처리 및 학습을 위한 LSTM_pipeline 사용설명서

경남대학교 컴퓨터공학부 융합인공지능 연구실

지도교수: 박미영

작성자: 곽서은

A. Environment setting

1. Anaconda 설치

LSTM_pipeline은 사용하기 앞서 pc에 anaconda가 설치되어 있는지 확인하여 주십시오.

설치가 되어있지 않다면

공식 홈페이지(https://www.anaconda.com/download) 를 참조하여 anaconda를 설치하여 주시기 바랍니다.

2. LSTM_pipeline 경로 설정

본 문서는 document 문서 안에서의 사용법에 대해 안내드리고 있습니다.

가급적 LSTM_pipeline은 document(문서)폴더 안에 위치시켜주십시오.

필독사항.

-본 프로그램은 Anaconda Powershell Prompt에 맞춰서 개발되었기에 Prompt에서 실행하는 것을 권장 드립니다.

-편의상, 뒷 내용부터는 Anaconda Powershell Prompt를 Prompt라 줄여말하겠습니다.

-Anaconda 설치가 성공적으로 완료되었다면, 윈도우 기준 작업표시줄의 검색창을 통하여 Anaconda Powershell Prompt을 검색하면 쉽게 실행시킬 수 있습니다.

-Prompt에서 코드를 실행하려면 아래에 나오는 코드들을 그대로 사용하시면 됩니다. 명령어 맨 앞에 "!" 문자가 있다면 제거한 뒤 사용하십시오

---

3. 작업위치를 LSTM_pipeline으로 변경

Anaconda Powersheel Prompt를 실행하고 아래의 코드를 통해 현재 위치를 파악해줍니다.

jupyter에서 실행하고자 하면 Guideline.ipynb 파일을 C:\Users\User와 같은 홈디렉터리에 위치 시킨뒤 진행하십시오. 다만, 코드가 제대로 동작하지 않을 수 있습니다.

```
pwd
```

```
'C:\WINDOWS\system32'
```

cd ~ 명령어를 통해 사용자의 홈 디렉터리로 이동합니다

```
cd ~
```

```
C:\Users\User
```

cd 명령어 + 폴더이름을 입력하여 작업위치를 변경해줍니다.

```
cd Documents/
```

```
C:\Users\User\Documents
```

```
cd  LSTM_pipeline/
```

```
C:\Users\User\Documents\LSTM_pipeline
```

만약, cd LSTM_pipeline에서 오류가 발생한다면 document 폴더 내에 LSTM_pipeline이 없다는 뜻이니 다운받은 폴더(LSTM_pipeline)를 document 내에 위치시켜 주십시오.

3. LSTM_pipeline을 위한 가상환경(lstm_env)설치

코드 실행에 필요한 모든 패키지들을 다운로드합니다.

```
conda env create -f lstm_env.yaml
```

가상환경 lstm_env를 활성화 합니다.

LSTM_pipeline을 사용할때는 항상 이 가상환경을 활성화 해주시길 바랍니다.

```
conda activate lstm_env
```

실행시킬 수 있는 LSTM_pipeline 코드들은 총 3개로

1_Preprocessing_tool.py

2_Learning_LSTM.py

3_Prediction.py

1,2,3번 순서대로 실행하시길 바랍니다.

1번과 2번이 선행되었으면 3번만 실행하셔도 됩니다.

C. 1_Preprocessing_tool.py

인자들은 -f, -t, -n, -m1, -m2, -l, -w, -p 등이 있습니다.

python 1_Preprocessing_tool.py -h 를 실행하면 각 인자들에 대한 설명을 출력합니다.

실행시 인자의 값으로 데이터 전처리를 수행합니다.

데이터 전처리를 마치면 전처리한 데이터와 통계표를 저장합니다.

경로: './Output/Data_preprocessing/

-f는 filename의 약자로 전처리할 파일이름입니다. 필수 인자입니다.

파일 이름은 역슬래쉬()가 아닌 슬래쉬(/)로 구성되어야 합니다.

파일은 time과 values로 구성된 csv파일이어야 합니다.

```
!python 1_Preprocessing_tool.py -f
C:/Users/User/Documents/LSTM_pipeline/Input/DDp_1.csv
```

-l는 level의 약자로 이상치 보정시, 정상범위에서의 이상치 허용 기준입니다. 기본값은 10%입니다.

```
!python 1_Preprocessing_tool.py -l 20
```

-t는 time의 약자로 시간 간격을 조정합니다.

'hourly', 'daily' 중 선택할 수 있습니다.

```
!python 1_Preprocessing_tool.py -t hourly
```

-n은 normal_range의 약자로 정상범위를 정합니다. -n만 입력할 시 정상범위는 데이터의 UIF(Upper Inner Fence) 또는 LIF(Lower Inner Fence)입니다. -m1이나 -m2를 통해 정상범위를 지정할 수 있습니다.

```
!python 1_Preprocessing_tool.py -n
```

-m1은 min의 약자로 정상범위의 최솟값입니다.

예를 들어 -m1 5를 입력하면 정상범위는 5 이상입니다.

```
!python 1_Preprocessing_tool.py -n -m1 5
```

-m2은 max의 약자로 정상범위의 최댓값입니다.

예를 들어 -m2 10을 입력하면 정상범위는 10 이하입니다.

```
!python 1_Preprocessing_tool.py -n -m2 10
```

더하여 -m1과 -m2를 입력하면 -m1보다 작거나 -m2보다 큰 값을 이상치로 정합니다.

예를 들어 -m1 5 -m2 10이면 정상범위는 5 이상이고 10 이하입니다.

```
!python 1_Preprocessing_tool.py -n -m1 5 -m2 10
```

-w은 week의 약자로 주말 또는 평일 데이터를 저장합니다.

'all', 'weekday', 'weekend' 중에서 선택할 수 있으며 기본은 all입니다.

```
!python 1_Preprocessing_tool.py -w weekday
```

-p은 preprocess의 약자로 특정한 인자의 값으로 데이터 전처리를 진행합니다. -f는 입력하셔야 합니다.

시간 간격(-t)은 daily, 정상범위(-n)는 UIF 또는 LIF, 이상치 허용 범위(-l)은 10%, 저장할 데이터(-w)는 전체 데이터(all)로 전처리를 실행합니다.

D. 2_Learning_LSTM.py

인자들은 -w, -p, -hp, -e, -b, -l 입니다.

python 2_Learning_LSTM.py -h 를 실행하면 각 인자들에 대한 설명을 출력합니다.

실행시 LSTM 모델을 학습합니다.

LSTM 모델 학습이 끝나면 성능 평가 그래프와 모델, 스케일러를 저장합니다.

경로: ./Output/Learning_lstm

-w는 window_size의 약자로 윈도우 사이즈를 정합니다. 기본값은 7이며 정수로 입력하셔야 합니다.

```
!python 2_Learning_lstm.py -w 7
```

-p는 periods의 약자로 예측 윈도우 사이즈를 정합니다. 기본값은 1이며 정수로 입력하셔야 합니다.

```
!python 2_Learning_lstm.py -p 1
```

-hp는 hyper_parameters의 약자로 하이퍼 파라미터를 변경합니다.

-e는 epochs의 약자로 에포크를 지정합니다. 기본값은 700입니다.

-b는 batch_size의 약자로 배치 사이즈를 지정합니다. 기본값은 32입니다.

배치 사이즈는 주로 2의 거듭제곱을 사용합니다.

```
!python 2_Learning_lstm.py -hp -e 500 -b 32
```

E. 3_Prediction

인자들은 -f, -p, -d, -c, -dp, -a 입니다.

python 3_Prediction.py -h 를 실행하면 각 인자들에 대한 설명을 출력합니다.

실행시 이상치 또는 예측 데이터를 저장합니다.

경로: ./Output/Prediction

-f는 filename의 약자로 예측 또는 이상탐지할 파일 이름입니다. 필수 인자입니다.

```
!python 3_Prediction.py -f ./Input/Raw_data.csv
```

-dp는 data_preprocessing의 약자로 데이터 전처리를 실행합니다. 기본적으로 결측치를 보정합니다.

-p를 선택하면 시간 간격을 자동적으로 병합합니다.

-t(--time)와 -w(--week)를 통해 지정할 수 있습니다.

```
!python 3_Prediction.py -dp
```

-t는 time의 약자로 시간 간격을 지정합니다.

'hourly', 'daily' 중에서 선택하셔야 합니다.

```
!python 3_Prediction.py -dp -t hourly
```

-w은 week의 약자로 주말 또는 평일 데이터를 구합니다.

'all', 'weekday', 'weekend' 중에서 선택할 수 있으며 기본은 all입니다.

```
!python 3_Prediction.py -dp -w weekday
```

-p는 prediction의 약자로 입력 데이터의 다음 시간을 예측하고 이상치 레벨을 기준으로 이상 탐지를 진행합니다.

예측 개수는 2_Learning_LSTM의 예측 윈도우 사이즈(-p)입니다.

```
!python 3_Prediction.py -p
```

-d는 detection의 약자로 이상치 레벨을 기준으로 이상치를 검출하고 저장합니다.

```
!python 3_Prediction.py -d
```

-a는 all의 약자로 이상 탐지와 예측을 수행합니다.

이상치와 예측 데이터를 저장합니다.

```
!python 3_Prediction.py -a
```

END

LSTM_pipeline에 대한 설명이 모두 끝이 났습니다.