개발 결과물 사용 설명서

딥러닝 기반 다변량 스팀 사용 이상 감지 및 영향변수의 원인 분석 기능 제작

2023. 11. 28.

서울과학기술대학교 데이터사이언스학과

연구책임자 심재웅

목차

1. 컴퓨팅 사양

- 운영체제 : Ubuntu 20.04.5 LTS
- CPU : AMD Ryzen 9 7950X 16-Core Processor * 32
- GPU : NVIDIA GeForce RTX 3090

2. 파일 구조

I. Random_forest.ipynb: Random forest 모델 구축 및 예측에서 주요한 변수 분석

II. 1D-CNN.ipynb: 1D-CNN 모델 구축 및 예측에서 주요한 변수 분석

III. LSTM_Attention.ipynb: LSTM&Attention 모델 구축 및 예측에서 주요한 변수 분석

IV. Dual_stage_attention_RNN.ipynb: DARNN 모델 구축 및 예측에서 주요한 변수 분석

V. requirements_linux.txt: 프로젝트 공유나 환경 복원 시 필요한 외부 패키지 목록을 담은 파일(Linux)

VI. requirements_window.txt: 프로젝트 공유나 환경 복원 시 필요한 외부 패키지 목록을 담은 파일(Window)

VII. df_ext(2023-04-01~2023-08-31,51250385)_2023-10-17 10-58-30 -seoultec.xlsx: 학습 및 검증을 위한 데이터셋

3. 개발 환경

3.1. Ubuntu (Linux)

- NVIDIA 드라이버 설치

1. 그래픽 카드에 맞는 NVIDIA 드라이버 버전 확인

```
bash : ubuntu-drivers devices
```

2. 그래픽카드에 호환되는 적절한 nvidia 드라이버 설치

```
bash : sudo apt-get install nvidia-driver-xxx
```

3. nvidia 드라이버를 커널에 로드해주는 nvidia-modprobe 도 같이 설치

```
bash : sudo apt-get install dkms nvidia-modprobe
```

4. 설치한 nvidia 드라이버 작동을 위한 업데이트 및 재부팅

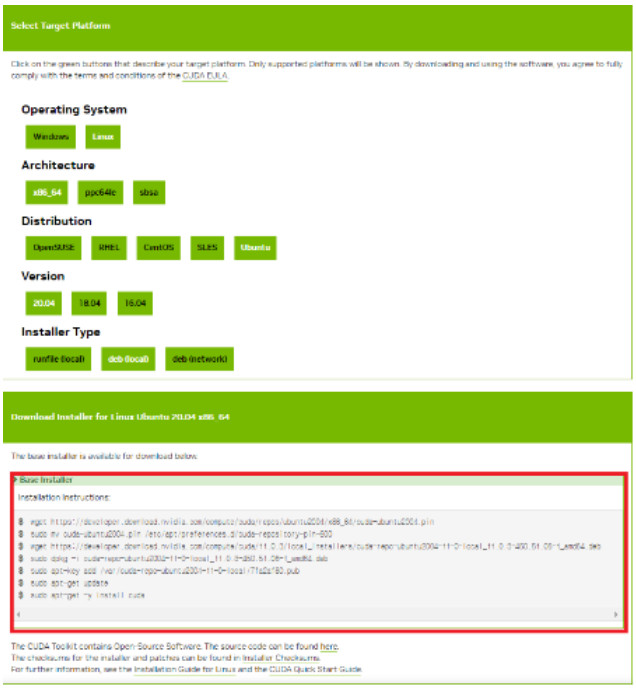
```
bash : sudo apt-get update
bash : sudo apt-get upgrade
bash : sudo reboot
```

- CUDA 11.2 설치

1. CUDA 설치 웹사이트 접속

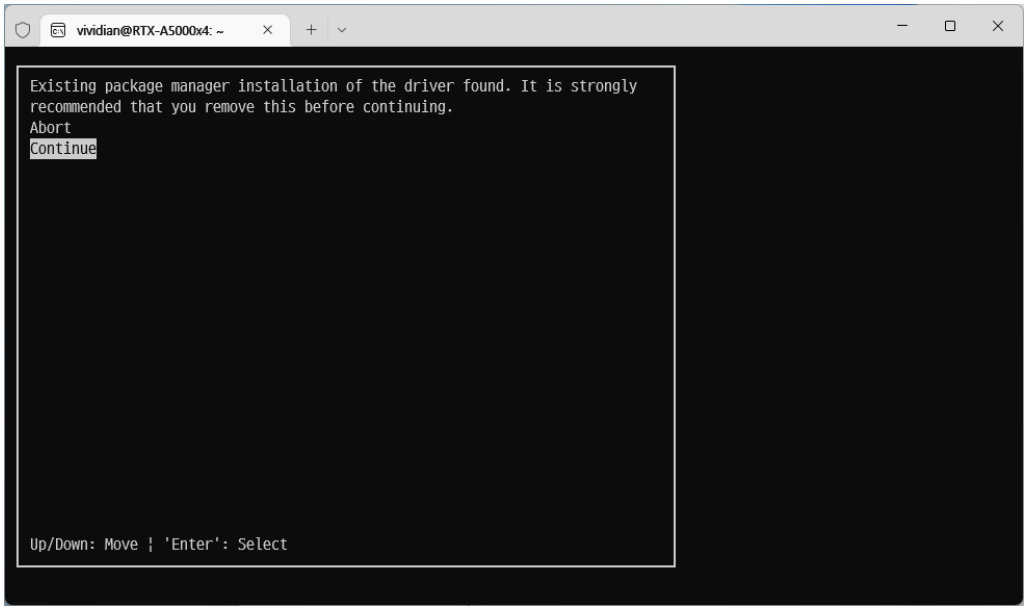
   https://developer.nvidia.com/cuda-11.2.0-download-archive?target_os=Linux&target_arch=x86_64&target_distro=Ubuntu&target_version=2004&target_type=deblocal

2. NVIDIA 에서 OS 조건에 맞는 CUDA11.2 선택



3. 위 이미지의 빨간 상자와 같은 파일 다운로드 명령어를 terminal 에 입력

4. 다운로드 후 실행이 완료되면 설치 진행



5. 설치된 Driver 를 제외하고 CUDA Toolkit 11.2, CUDA Demo Suite 11.2, CUDA Documentation 11.2 설치

6. 설치가 완료되면 PATH 설정을 위해 .bashrc 를 열여서 가장 마지막 부분에 export 부분을 입력해 설정하도록 하며, source ~/.bashrc 를 통해 업데이트

```
bash : sudo vi ~/.bashrc # open file
export PATH=/usr/local/cuda-11.2/bin${PATH:+:${PATH}}
export LD_LIBRARY_PATH=/usr/local/cuda-11.2/lib64:${LD_LIBRARY_PATH:+:${LD_LIBRARY_PATH}}
bash : source ~/.bashrc # apply after save file
```

7. PATH 설정 후 nvcc -V 를 통해 설치 버전 확인

- CUDNN 8.1 설치

1. CUDNN 다운로드 웹사이트 접속

   https://developer.nvidia.com/rdp/cudnn-archive

2. Linux (X85_64)용 cuDNN Libary 를 다운로드

3. 다운로드 받은 cuDNN 압축을 풀고, /usr/local/cuda 디렉토리로 복사

```
bash : sudo cp cuda/include/cudnn*.h /usr/local/cuda-11.2/include
bash : sudo cp cuda/lib64/libcudnn* /usr/local/cuda-11.2/lib64
bash : sudo chmod a+r /usr/local/cuda-11.2/include/cudnn*.h /usr/local/cuda-11.2/lib64/libcudnn*
```

4. 설치 후 버전 확인

```
bash : cat /usr/local/cuda/include/cudnn_version.h | grep CUDNN_MAJOR -A 2
```

- Python 설치

- 터미널 실행 후 아래의 명령어를 통해 Python 3.8 설치

```
bash : sudo apt install python3.8
```

- 가상환경 생성

- 실행 시킬 파일이 존재하는 폴더로 진입 후, 아래의 명령어를 통해 가상환경 생성

```
bash : python -m venv 가상환경이름
```

- 가상환경 실행

- 아래의 명령어를 통해 가상환경 실행

```
bash : source 가상환경이름/bin/activate
```

- 활용 라이브러리 설치

1. pytorch 설치: 아래의 명령어를 통해 가상환경 내 pytorch 설치

```
bash : pip install torch==1.7.1+cu110 torchvision==0.8.2+cu110 torchaudio==0.7.2 -f https://download.pytorch.org/whl/torch_stable.html
```

2. tensorflow 설치: 아래의 명령어를 통해 가상환경 내 tensorflow 설치

```
bash : pip install tensorflow-gpu==2.10
```

3. 나머지 package 설치: 아래의 명령어를 통해, 나머지 필요 package 들을 설치

```
bash : pip install -r requirements_linux.txt
```

3.2. Window

- NVIDIA 드라이버 설치

- Driver 설치 웹사이트 접속 https://www.nvidia.co.kr/Download/index.aspx?lang=kr
- 그래픽 카드에 맞는 NVIDIA 드라이버를 다운 및 설치

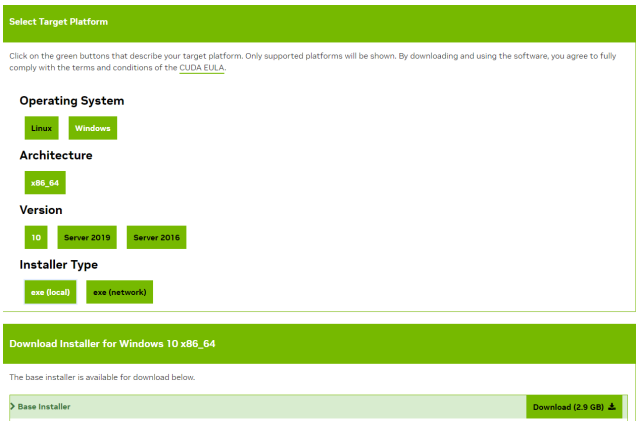
- CUDA 11.2 설치

  다운로드 링크

  1. CUDA 설치 웹사이트 접속

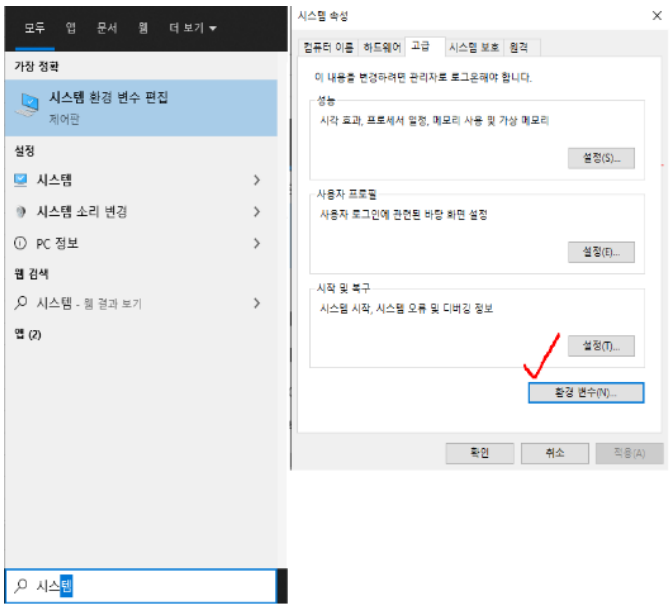
     https://developer.nvidia.com/cuda-11.2.0-download-archive?target_os=Windows&target_arch=x86_64

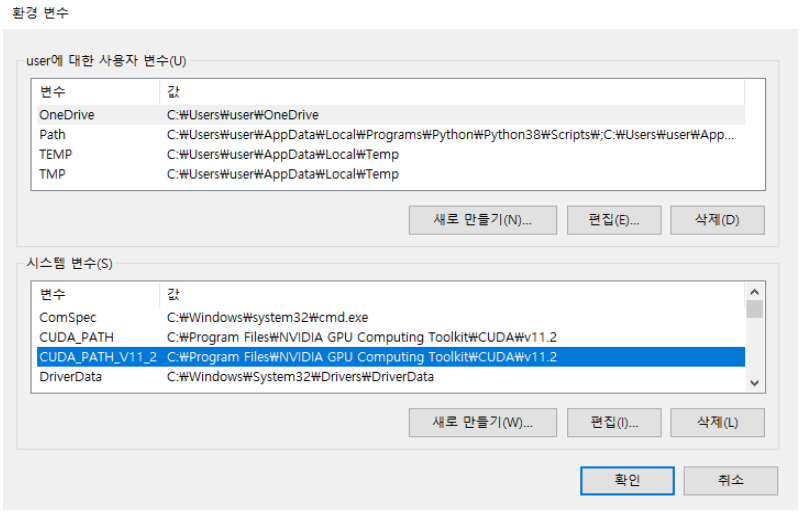
  2. NVIDIA 에서 CUDA11.2 설치 파일 다운로드



  3. CUDA 11.2 설치

  4. 설치확인





- CUDNN 8.1 설치

  1. CUDNN 다운로드 웹사이트 접속

     https://developer.nvidia.com/rdp/cudnn-archive

  2. NVIDIA 에서 CUDNN 8.1 파일 다운로드
  3. 다운로드한 압축 파일 해제
  4. 압축 해체한 파일을 설치한 CUDA 폴더에 덮어씌우기
  5. CUDA 경로 : C:₩Program Files₩NVIDIA GPU Computing Toolkit₩CUDA₩v11.2

- Python 설치

  1. Python 공식 웹사이트에 접속하여 Python 3.8 설치 파일 다운로드

     (https://www.python.org/downloads/)

  2. 다운로드한 설치 프로그램을 실행
  3. 설치 과정을 시작하기 전에 "Add Python 3.8 to PATH" 옵션을 선택
  4. "Install Now"을 선택하여 Python 3.8 을 설치

- 가상환경 생성

- 실행시킬 파일이 존재하는 폴더로 진입 후, 아래의 명령어를 통해 가상환경 생성

```
bash : python -m venv 가상환경이름
```

- 가상환경 실행

- 아래의 명령어를 통해 가상환경 실행

```
bash : 가상환경이름₩Scripts₩activate
```

- 활용 라이브러리 설치

1. pytorch 설치: 아래의 명령어를 통해 가상환경 내 pytorch 설치

```
bash : pip install torch==1.7.1+cu110 torchvision==0.8.2+cu110 torchaudio==0.7.2 -f https://download.pytorch.org/whl/torch_stable.html
```

2. ensorflow 설치: 아래의 명령어를 통해 가상환경 내 tensorflow 설치

```
bash : pip install tensorflow-gpu==2.10
```

3. 나머지 package 설치: 아래의 명령어를 통해, 나머지 필요 package 들을 설치

```
bash : pip install -r requirements_window.txt
```

4. 주요 라이브러리

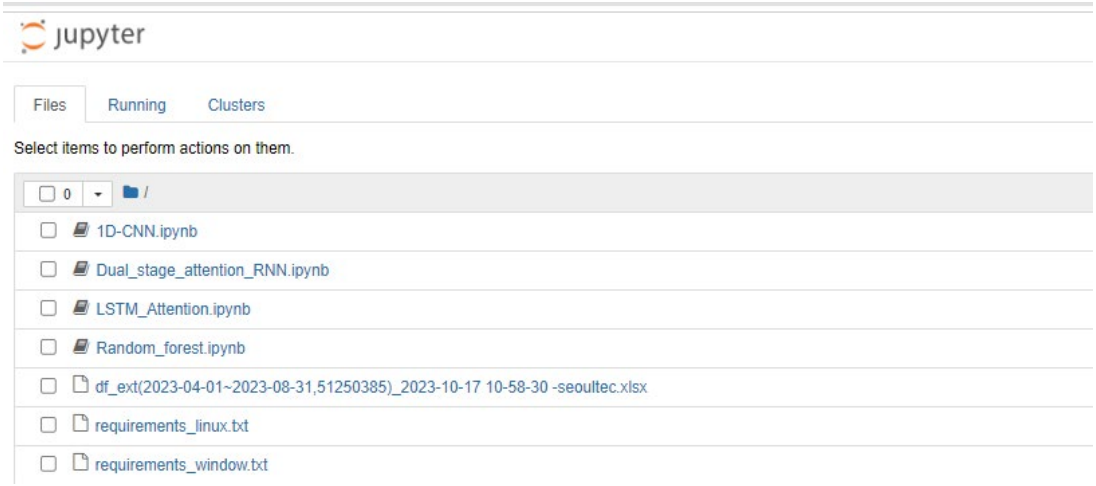
	라이브러리	버전	용도
Pytorch	torch	1.8.0+cu111	딥러닝 라이브러리
	torchaudio	0.8.0	PyTorch 의 오디오 데이터를 처리하기 위한 확장
	torchmetrics	0.5.0	PyTorch 모델을 위한 머신러닝 메트릭 모음
	torchvision	0.9.0+cu111	이미지 처리를 위한 유용한 데이터셋, 모델, 변환 제공 라이브러리
	pytorch-lightning	1.3.8	PyTorch 의 래퍼로, 더 쉽고 빠른 실험을 위한 라이브러리
Tensorflow	tensorflow-gpu	2.10.0	GPU 지원 TensorFlow, 머신러닝 및 신경망 라이브러리

	tensorflow-estimator	2.10.0	TensorFlow 를 사용하여 학습, 평가, 예측을 위한 모델 정의 라이브러리
	tensorboard	2.10.1	TensorFlow 실행 시각화 도구
	tensorboard-data-server	0.6.1	TensorBoard 에 데이터를 제공하여 모델 학습 진행 상황 및 성능을 시각적으로 모니터링
	tensorboard-plugin-wit	1.8.1	TensorBoard 의 시각화를 확장하여 데이터의 시각적 분석을 개선하는 플러그인
	tensorflow-io-gcs-filesystem	0.34.0	TensorFlow와 Google Cloud Storage(GCS) 파일 시스템 간 상호 작용을 위한 입출력 모듈
	numpy	1.24.4	수학 연산을 위한 기본 python 패키지
	pandas	2.0.3	데이터 조작 및 분석을 위한 고수준 데이터 구조 및 조작 도구
	shap	0.39.0	모델의 예측을 설명하기 위한 라이브러리
Scikit-learn	scikit-learn	1.3.1	머신러닝 알고리즘 및 도구를 제공하는 라이브러리
	scikit-image	0.21.0	이미지 분석과 조작을 지원하는 과학적 이미지 처리를 위한 라이브러리
Math	sympy	1.12	수학적 기호 계산을 위한 라이브러리
	mpmath	1.3.0	고도의 수학적 기호 및 수치 계산을 위한 라이브러리
matplotlib	matplotlib	3.7.3	python 에서 2D 도표를 생성을 위한 라이브러리
	matplotlib-inline	0.1.6	Matplotlib 그래프를 직접 출력하기 위한 도구를 제공하는 라이브러리
	seaborn	0.13.0	matplotlib 기반의 통계적 그래픽 라이브러리

5. 실행 방법 및 예시

- 주피터 노트북 실행 후 각 파일 실행

```
bash : jupyter notebook
```



- 모델 실행 결과 모델명의 폴더 생성이 된 후, 다음과 같은 파일들 생성

모델명	파일명	파일 설명
Random Forest	model_performance.txt	학습한 모델을 테스트한 결과가 저장된 파일
	Time series prediction plot.png	시간에 따른 학습 모델의 예측값과 실제값 비교 그래프
	Scatter plot.png	학습된 모델의 예측값 대 실제값 비교 산점도
	Random Forest feature plot.png	모델의 주요 특성 중요도 10개를 추출하고 이를 시각화
1D-CNN	best_model.pth	Validation loss가 가장 낮은 epoch 에서의 모델 weight를 저장해둔 파일
	last_model.pth	가장 마지막에 학습한 모델 weight를 저장해둔 파일
	model_performance.txt	시간에 따른 학습 모델의 예측값과 실제값 비교 그래프
	loss.png	모든 epoch에 대해 train loss와 validation loss 그래프
	Time Series Prediction.png	시간에 따른 학습 모델의 예측값과 실제값 비교 그래프
	scatter plot.png	학습된 모델의 예측값 대 실제값 비교 산점도
	shap summary plot.png	모델의 각 특성이 예측에 미치는 영향력과 방향을 색상과 SHAP 값으로 나타낸 그래프
	cumulative shap plot.png	테스트 데이터셋에 대해 각 특성별로 평균 절대 SHAP 값의 크기를 시각화 한 그래프
LSTM-Attention	best_model.pth	Validation loss가 가장 낮은 epoch 에서의 모델 weight를 저장해둔 파일
	last_model.pth	가장 마지막에 학습한 모델 weight를 저장해둔 파일
	model_performance.txt	시간에 따른 학습 모델의 예측값과 실제값 비교 그래프
	loss.png	모든 epoch에 대해 train loss와 validation loss 그래프
	Time Series Prediction.png	시간에 따른 학습 모델의 예측값과 실제값 비교 그래프
	scatter plot.png	학습된 모델의 예측값 대 실제값 비교 산점도
	Average Attention Map.png	Time stamp 관련 attention weights 시각화
	shap summary plot.png	모델의 각 특성이 예측에 미치는 영향력과 방향을 색상

		과 SHAP 값으로 나타낸 그래프
	cumulative shap plot.png	테스트 데이터셋에 대해 각 특성별로 평균 절대 SHAP 값의 크기를 시각화 한 그래프
DARNN	best_model.ckpt.index	텐서플로우/케라스 모델의 변수 이름과 그들의 저장 위치를 매핑하는 인덱스 정보
	best_model.ckpt.data-00000-of-00001	모델의 실제 가중치와 매개변수를 저장하는 바이너리 데이터 파일
	loss.png	모든 epoch에 대해 train loss와 validation loss 그래프
	Time Series Prediction.png	시간에 따른 학습 모델의 예측값과 실제값 비교 그래프
	scatter plot.png	학습된 모델의 예측값 대 실제값 비교 산점도
	Feature Attention Map.png	Feature 관련 attention weights 시각화
	Time Stamp Attention Map.png	Time stamp 관련 attention weights 시각화