목차

[**1.** **AI 모델 구조 설계** 2](#_Toc184889686)

[**1.1.** **구조정의** 2](#_Toc184889687)

[**1.2.** **구조설계** 2](#_Toc184889688)

[**2.** **AI 인터페이스 기능 설계** 5](#_Toc184889689)

[**2.1.** **기능정의** 5](#_Toc184889690)

[**2.2.** **기능절차** 5](#_Toc184889691)

[**2.3.** **인터페이스** 6](#_Toc184889692)

[**3.** **사용자 인터페이스 기능 설계** 7](#_Toc184889693)

[**3.1.** **기능정의** 7](#_Toc184889694)

[**3.2.** **기능절차** 7](#_Toc184889695)

[**3.3.** **인터페이스** 7](#_Toc184889696)

[**3.4.** **프로그램명** 7](#_Toc184889697)

[**3.5.** **프로그램 명세서** 7](#_Toc184889698)

[**4.** **데이터베이스 설계** 7](#_Toc184889699)

[**4.1.** **데이터베이스 구성** 7](#_Toc184889700)

[**4.2.** **데이터페이스 설계** 7](#_Toc184889701)

[**5.** **개발환경** 7](#_Toc184889702)

[**5.1.** **하드웨어 구성** 7](#_Toc184889703)

[**5.2.** **소프트웨어 구성** 8](#_Toc184889704)

1. **AI 모델 구조 설계**
   1. **구조정의**

인공지능 모델 및 데이터 분석 모듈을 활용하기 위한 라이브러리((Pandas, Miniconda, Jupyter, TensorFlow 등)를 적용하여 기술 개발을 진행하고, 돈사 데이터를 분석하여 각 Tier 별 영향인자 데이터를 분석하여 인공 신경망 모델을 통해 학습한다. 생성한 학습 모델을 통해 예측 데이터를 추출하여 웹U를 통해 데이터를 제공하도록 설계한다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

* 1. **구조설계**

인공 신경망 tensorflow 프레임워크를 적용하여 Flask API 와 연동하도록 구성하고, public API 통해 웹 인터페이스를 연동하여 실행 가능한 형태로 구성한다. Flask API 통해 각 모델의 필수 기능만 노출하도록 처리하고, 내부에서 사용하는 private 메소드를 별도로 구성하고, 소스 제어를 쉽게 할 수 있도록 메소드별 처리 기능을 분리하여 설계한다. 인공신경망 라이브러리, 서비스, 컨트롤 등 각 class 단위로 처리하도록 구성한다.

**텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명**

|  |  |  |  |
| --- | --- | --- | --- |
| **기능** | **클래스 및 파일** | **메소드** | **설명** |
| 애플리케이션 추가 | app.py | routes\_list | Flask 애플리케이션에 라우트를 추가 |
| 딥러닝 예측 | controllers\model\_met\_controller.py | PredictMetModel\_DeepL\_1 | 입력된 데이터(13개의 입력 항목)를 기반으로 예측을 수행합니다. 모델을 사용하여 탄소 배출량을 예측하고 결과를 반환( JSON 형식의 데이터를 받아들이고, 해당 데이터를 전처리하여 결과를 반환) |
| 딥러닝 예측 | controllers\model\_met\_controller.py | PredictMetModel\_DeepL\_2 | 입력된 데이터(5개의 입력 항목)를 기반으로 예측을 수행합니다. 모델을 사용하여 탄소 배출량을 예측하고 결과를 반환( JSON 형식의 데이터를 받아들이고, 해당 데이터를 전처리하여 결과를 반환) |
| 딥러닝 예측 | controllers\model\_met\_controller.py | PredictMetModel\_DeepL\_3 | 입력된 데이터(MET 항목)를 기반으로 예측을 수행합니다. 모델을 사용하여 탄소 배출량을 예측하고 결과를 반환( JSON 형식의 데이터를 받아들이고, 해당 데이터를 전처리하여 결과를 반환) |
| 데이터 전처리 | controllers\model\_met\_controller.py | Preprocessing | 돈사 데이터의 전처리를 수행하는 API 엔드포인트. POST 요청을 통해 JSON 형식의 데이터를 받아들이고, 해당 데이터를 전처리하여 결과를 반환 |
| 데이터 전처리 | services\model\_met\_services.py | PreprocessingData | 주어진 데이터의 전처리를 수행하는 기능( 데이터의 인덱스 간의 차이를 계산 및 데이터 전처리 과정에서 발생할 수 있는 오류를 처리하고, 전처리된 데이터를 JSON 형식으로 반환 ) |
| 데이터 예측 | services\model\_met\_services.py | MetPredictPigroomFromJson | JSON 형식으로 전달된 데이터를 기반으로 예측을 수행 |
| 모델 생성 학습 | library\MetGenerater.py | modelTrainningNoFeatures | 주어진 데이터에서 독립 변수를 사용하지 않고 종속 변수(CH4)만을 기반으로 모델을 학습하는 기능을 수행 |
| 모델 생성 학습 | library\MetGenerater.py | modelTrainningWithFeatures | 주어진 데이터로부터 모델을 학습하고 생성하는 기능을 수행 |
| 데이터 예측 | library\MetGenerater.py | modelPredictWithFeatures | 주어진 데이터와 저장된 모델을 사용하여 예측값을 생성하는 기능을 수행 |
| 데이터 예측 | library\MetGenerater.py | modelPredictNoFeatures | 주어진 데이터에서 종속 변수(CH4)를 사용하여 모델의 예측값을 생성하는 기능을 수행 |
| 데이터 예측 | controllers\model\_met\_controller.py | MetModelCreate\_DeepL\_1 | 돈사 데이터와 관련된 메탄(CH4) 배출량 예측 모델을 생성하는 데 사용됩니다. |
| 모델 생성 학습 | services\model\_met\_services.py | MetModelCreate | RNN(순환 신경망) 학습 모델을 생성하는 기능을 수행합니다. 이 메서드는 주어진 파라미터를 기반으로 데이터를 로드하고, 모델을 설정한 후, 학습을 진행 |
| 데이터 예측 | services\model\_met\_services.py | MetPredictPigroom | 여러 개의 입력된 파라미터를 기반으로 돈사의 탄소배출량 예측을 수행 |
| 데이터 예측 | services\model\_met\_services.py | MetPredictPigroomNoFeatures | 입력된 파라미터(Ch4 1개의 입력값)를 기반으로 돈사의 탄소배출량 예측을 수행 |
| 데이터 처리 | library\MetGenerater.py | windowed\_dataset | 주어진 데이터를 윈도우 크기와 배치 크기에 따라 분할하여 TensorFlow 데이터셋을 생성하는 기능을 수행 |
| 데이터 처리 | library\MetGenerater.py | windowed\_dataset2 | 주어진 데이터셋을 윈도우(window) 기반으로 분할하여 배치(batch) 형태로 반환하는 기능을 수행 |
| 모델 생성 | library\MetGenerater.py | setModel | Sequential 모델을 생성하고, Conv1D 레이어와 LSTM 레이어를 포함한 LSTM(장기 단기 기억) 모델을 설정 |
| 데이터 처리 | library\MetGenerater.py | generate\_date\_range | 주어진 시작 날짜와 종료 날짜 사이의 날짜 범위를 생성하는 기능을 수행 |
| 데이터 예측 | library\MetGenerater.py | predict\_future | 주어진 LSTM 모델을 사용하여 미래의 값을 예측하는 기능을 수행 |
| 데이터 처리 | services\resample\_services.py | preprocess\_data | 데이터프레임을 전처리하여 1분 간격으로 리샘플링하고 결측치를 보완하는 기능을 수행 |
| 환기율 계산 | services\model\_met\_services.py | CalcVentilationRate | 주어진 환기량과 섹션 유형에 따라 특정 공간의 환기율을 계산 |
| 환기율 계산 | services\model\_met\_services.py | CalcVentilationRateFromRoomType | 방의 유형에 따라 환기율을 계산하는 기능. 해당 방의 부피를 기반으로 환기율을 계산 |
| 시뮬레이션 타입 | controllers\model\_met\_controller.py | process\_simulation\_types | 시뮬레이션 타입에 따라 다양한 시뮬레이션을 요청하고 그 결과를 수집하는 역할 |
| API 실행 | controllers\model\_met\_controller.py | PigroomTier\_1 | 탄소 배출량을 예측하는 기능을 수행( 회귀식1) |
| API 실행 | controllers\model\_met\_controller.py | PigroomTier\_2 | 탄소 배출량을 예측하는 기능을 수행( 회귀식2) |
| API 실행 | controllers\model\_met\_controller.py | PigroomTier\_3 | 탄소 배출량을 예측하는 기능을 수행( 회귀식3) |
| API 실행 | controllers\model\_met\_controller.py | PigroomIPCC | 탄소 배출량을 예측하는 기능을 수행( IPCC 계산식) |
| 예측 모델 실행 | controllers\model\_met\_controller.py | PredictRegressionJson\_IPCC | 탄소 배출량을 예측하는 회귀 모델을 실행. IPCC 기준에 따라 탄소 배출량을 계산 |
| 예측 모델 실행 | controllers\model\_met\_controller.py | PredictRegressionJson\_Tier1 | 주어진 매개변수를 기반으로 탄소 배출량을 예측하는 회귀 모델을 실행(AVERAGE\_WEIGHT,Temp\_Out,EC\_P,CO2) |
| 예측 모델 실행 | controllers\model\_met\_controller.py | PredictRegressionJson\_Tier2 | 주어진 매개변수를 기반으로 탄소 배출량을 예측하는 회귀 모델을 실행(AVERAGE\_WEIGHT,Temp\_Out,EC\_P,CO2, Ventilation rate,RH\_Out,NH3,Temp) |
| 예측 모델 실행 | controllers\model\_met\_controller.py | PredictRegressionJson\_Tier3 | 주어진 매개변수를 기반으로 탄소 배출량을 예측하는 회귀 모델을 실행(AVERAGE\_WEIGHT,Temp\_Out,EC\_P,CO2, Ventilation rate,RH\_Out, NH3, Temp, RH,Temp\_P, PH\_P) |

1. **AI 인터페이스 기능 설계**
   1. **기능정의**

* 탄소배출 딥러닝 1 모델 : 입력된 돈사 데이터(13개의 입력 항목)를 기반으로 예측을 수행한다. 딥러닝 1 모델을 사용하여 탄소 배출량을 예측하고 결과를 반환한다.
* 탄소배출 딥러닝 2 모델 : 입력된 돈사 데이터(5개의 입력 항목)를 기반으로 예측을 수행한다. 딥러닝 2 모델을 사용하여 탄소 배출량을 예측하고 결과를 반환한다.
* 탄소배출 딥러닝 3 모델 : 입력된 돈사 데이터(MET 항목)를 기반으로 예측을 수행한다. 딥러닝 3 모델을 사용하여 탄소 배출량을 예측하고 결과를 반환한다.
* 탄소배출 전처리 : 돈사 데이터의 빈 값을 확인하여 결측 된 데이터를 보간 방법을 통해 전처리 수행 후 처리 결과 데이터를 반환한다.
  1. **기능절차**

AI 딥러닝 모델은 웹 인터페이스를 통해 사용자에게 실행 기능을 제공하고, 딥러닝 학습 모델 기반으로 메탄 결과와 전처리 기능을 제공한다.

텍스트, 도표, 라인, 스크린샷이(가) 표시된 사진

자동 생성된 설명

* 1단계 : DASHBOARD – 탄소배출 딥러닝 3 모델을 통해 매일 스케쥴러가 생성하는 예측 데이터를 일단위로 시각화하여 표시한 결과를 확인한다.
* 2단계 : SIMULATION – 신규 SIMULATION 데이터를 선택 및 입력한다.

모델 설명 : 생성 SIMULATION 모델 설명 입력

시뮬레이션 유형 : 트윈형

적용농자 : 예산 공주대 축사 선택

분야 선택 : 비에너지

적용모델 : 탄소배출 딥러닝 1 모델, 탄소배출 딥러닝 2 모델

기간 선택 : 예측에 활용할 돈사 데이터 선택

축사 선택 : 비육돈사/자돈사

* 3단계 : SIMULATION – 기간에 해당하는 데이터 가져온다.

데이터 전처리 : 탄소배출 전처리 API 를 통해 적용

* 4단계 : SIMULATION – 실행한다.

선택한 탄소배출 딥러닝 1 모델, 탄소배출 딥러닝 2 모델을 적용하여 예측 데이터를 추출한다.

* 5단계 : 예측 결과 최종 데이터를 확인한다.
  1. **인터페이스**
* 탄소배출 딥러닝 모델 1

입력 : 돈사 데이터 (데이터의 datestamp, 분뇨 온도, 분뇨 EC, 분뇨측정기, 외부 온도, 외부 습도, 환기율, 내부 온도, 내부 습도, 내부 이산화탄소 농도, 내부 암모니아 농도, 돼지 중량, 사육두수, 급이량)

출력 : 메탄 결과 데이터 객체

* 탄소배출 딥러닝 모델 2

입력 : 돈사 데이터 (데이터의 datestamp, 내부 이산화탄소 농도, 내부 암모니아 농도, 돼지 중량, CH4, 급이량)

출력 : 메탄 결과 데이터 객체

* 탄소배출 딥러닝 모델 3

입력 : 돈사 데이터 (데이터의 datestamp, CH4)

출력 : 메탄 결과 데이터 객체

* 탄소배출 전처리

입력 : 돈사 데이터 (데이터의 datestamp, 분뇨 온도, 분뇨 EC, 분뇨측정기, 외부 온도, 외부 습도, 환기율, 내부 온도, 내부 습도, 내부 이산화탄소 농도, 내부 암모니아 농도, 돼지 중량, 급이량, 사육두수)

출력 : 돈사 전처리 결과 객체

1. **사용자 인터페이스 기능 설계**
   1. **기능정의**
   2. **기능절차**
   3. **인터페이스**
   4. **프로그램명**
   5. **프로그램 명세서**
2. **데이터베이스 설계**
   1. **데이터베이스 구성**
   2. **데이터페이스 설계**
      1. **사용자 데이터 정의**
      2. **객체 관계 모델링(ERD) – 논리, 물리**
      3. **테이블 명세서**
3. **개발환경**
   1. **하드웨어 구성**

하드웨어는 아래 서버 정보로 구성하여 AI API 인터페이스와 사용자 인터페이스를 설치하여 서비스 구성함

|  |  |
| --- | --- |
| **구분** | **세부 정보** |
| 프로세서 | 인텔 제온 프로세서 W-2223  (4C 3.6GHz 3.9GHz Turbo HT 8.25MB 120W DDR4- 2666) |
| 프로세서 | CPU 히트싱크 5820 타워 |
| 섀시 옵션 | Precision 5820 타워 950W PCIe FlexBay 섀시 CL |
| 메모리 | 192GB 6x32GB DDR4 2933MHz RDIMM ECC 메모리 |
| 비디오 솔루션 | 듀얼 Nvidia T400 4GB, 4GB, 3 mDP to DP 어댑터 |
| 키보드 | Dell 유선 키보드 KB216 블랙 (한글) |
| 마우스 | Dell MS116 유선 마우스 블랙 |
| Boot 드라이브 옵션 | 인텔 NVMe PCIe SSD (Front PCIe FlexBay) |
| 1st 하드 드라이브 | Dell M.2 캐리어 |
| 1st 하드 드라이브 | M.2 1TB PCIe NVMe Class 40 SSD |
| RAID HDD/SSD | SATA 없음 RAID |
| 전원 코드 | 시스템 전원 코드 220V (한국) |
| 네트워크 카드 | 추가 네트워크 카드 불포함(통합 NIC 포함) |
| 5.24 Flexbay | 옵티컬 불포함 |
| 하드 드라이브 컨트롤러 | 인텔 통합형 컨트롤러  (RST-e) (1-2 Front FlexBay NVMe PCIe 드라이브) |
| OS | Ubuntu 22.04.3 LTS 설치 |
| 비디오 솔루션 | NVIDIA® Tesla™ T4 16GB, 패시브, 싱글 와이드 |
| \*특이사항 | \* NVIDIA® Tesla™ T4: 베이스 내부 장착  \* OS: Ubuntu 22.04.3 LTS 설치 |

* 1. **소프트웨어 구성**
     1. **AI 개발환경 및 설치 정보**

리눅스 커널 버전 : 5.15.0-125-generic

리눅스 정보 : Ubuntu 22.04.5 LTS

리눅스 가상화 미니콘다 버전 : conda 24.9.2

미니콘다 경로 : /home/etri/carbon/miniconda3

미니콘다 가상화 경로 : /home/etri/carbon/miniconda3/envs/etri-carbon-api/

미니콘다 가상화 이름 : etri-carbon-api

가상화 파이썬 버전 정보 : Python 3.9.20

API 구성에 사용된 패키지 버전 정보 : Flask 3.0.3 전체적인 api 구성에 사용된 웹 프레임워크

API 구성에 사용된 패키지 버전 정보 : tensorflow-gpu : 2.10.0

API 구성에 사용된 패키지 버전 정보 : keras : 2.10.0 딥러닝 모델을 구축하고 훈련하는 데 사용

API 구성에 사용된 패키지 버전 정보 : matplotlib : 3.7.1 데이터 분석 결과를 시각적 표현 및 차트

API 구성에 사용된 패키지 버전 정보 : numpy : 1.24.0 탄소 예측 모델에서 데이터를 처리하고, 수학적 계산을 수행하는 데 사용

API 구성에 사용된 패키지 버전 정보 : pandas : 1.5.3 데이터 관리및 분석, 다양한 형식의 데이터를 처리

API 구성에 사용된 패키지 버전 정보 : scikit-learn : 1.5.1 데이터 전처리, 모델 훈련, 평가 및 예측을 위한 도구

API 경로 : /home/etri/carbon/etri-carbon-api/

API 실행 경로 : /home/etri/carbon/etri-carbon-api/app.py

개발 에디터 정보 : visual studio code 1.95.2

DB 데이터 확인 : Dbeaver 22.0.1

원격 접속 클라이언트 : mobaxterm 11.1

* + 1. **사용자 인터페이스**

JDK 정보 : JDK 1.8

JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64

Apache Tomcat 서버의 설치 경로 Tomcat : /apps/tomcat/apache-tomcat-9.0.97

Nginx 웹 서버의 설치 경로 : /etc/nginx

nginx version: nginx/1.18.0 (Ubuntu)

URL: http://192.168.88.166/eoco2/login, (id/password: admin/admin123456#)

애플리케이션의 설치 경로 및 접근 URL /apps/eoco2\_wireframe\_admin