**2023 전기 졸업과제 중간보고서**

**목차**

[1. 기존 계획 및 수정사항 1](#_Toc142397881)

[**1.1 기존 목표** 2](#_Toc142397882)

[**1.2 수정 사항** 2](#_Toc142397883)

[2. 데이터 수집 및 전처리 2](#_Toc142397884)

[**2.1 사용된 데이터셋 소개** 2](#_Toc142397885)

[**2.2 데이터 전처리** 2](#_Toc142397886)

[3. 모델 설계 및 구현 5](#_Toc142397887)

[**3.1 모델 설계 상세** 5](#_Toc142397888)

[**3.2 모델 학습** 6](#_Toc142397889)

[4. 갱신된 과제 추진 계획 6](#_Toc142397890)

[**4.1 갱신된 개발 프로세스 일정** 6](#_Toc142397891)

[**4.2 구성원 별 진척도** 7](#_Toc142397892)

[5. 결론 및 추후 계획 7](#_Toc142397893)

[**5.1 중간 결과 요약** 7](#_Toc142397894)

[**5.2 수정된 계획과 추가 연구 방향** 7](#_Toc142397895)

[**5.3 잠재적 문제점과 대응방향** 8](#_Toc142397896)

1. 기존 계획 및 수정사항

**1.1 기존 목표**

실리코 예측을 이용한 새로운 약물-표적 상호작용(drug–target interaction)의 성공적인 식별은 신약 개발에 있어 중요한 과정이다. 본 과제에서는 기존의 화합물의 2D 특징을 사용한 연구에서 벗어나 딥 러닝 모델과 단백질 시퀀스와 화합물 1D 표현만을 사용해 약물 표적 결합 친화도를 예측하는 것을 목표로, CNN, KronRLS, SimBoost과 같은 예측 모델을 생성해 SMILES 표현의 시퀀스를 학습시켜 DeepDTA에 공급해 결합 친화도를 예측하고, 그리드 서치를 통한 최적화된 하이퍼파라미터를 도출하여 Davis, Kiba 데이터셋을 사용해 구축한 모델을 검증할 계획이다.

현재는 단백질 시퀀스와 화합물 SMILES 표현을 전처리 후, CNN을 사용해 SMILES 표현을 학습하는 모델을 구축하는 과정을 진행하고 있다

**1.2 수정 사항**

기존에는 CNN 외로도 추가적으로 KronRLS과 SimBoost 모델을 훈련시켜 세 모델의 예측 결과를 비교할 계획이었으나, 커널 행렬 생성 과정에서 커널 행렬의 크기가 커져 주어진 환경에서 다루기 어려웠다. 토의 끝에 딥 러닝 모델 자체가 KronRLS과 SimBoost을 대체할 수 있어 KronRLS과 SimBoost모델 구축은 불필요하다 판단하고 CNN 모델 하나를 구축하는 것으로 계획을 축소시켰다.

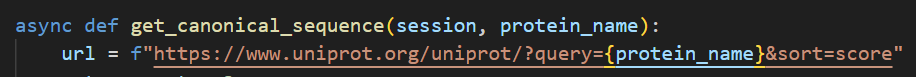
2. 데이터 수집 및 전처리

**2.1 사용된 데이터셋 소개**

본 과제에서는 딥 러닝 모델에 학습시키기 위한 데이터로 단백질 서열과 SMILES 표현 서열, 그리고 결합 친화도 점수, 이 세가지 항목으로 이루어진 데이터셋을 만들었다. 이때 사용된 데이터는 BindingDB의 binding affinity data from PubChem 로, binding affinity value로는 IC50을 사용했다.

**2.2 데이터 전처리**

* 모델을 단순화시키기 위해 필요한 인풋 데이터를 csv(엑셀) 형식으로 저장했다. 데이터는 4가지 항목 (ligand SMILES, 단백질 체인 서열, IC50 값, 그리고 test/train/validation set 세분화)을 가지고 있다.
* BindingDB의 전처리 하기 전 데이터셋은 단백질 체인 서열에 대한 정보가 없고 단백질 이름(Target Name)만 담고 있었기에 단백질 이름을 단백질 체인 서열로 바꿔주는 labeling 작업이 필요했다. 120000개나 되는 데이터를 일일이 labeling하는 것은 불가능했기에 단백질 이름을 해당하는 단백질 체인 서열로 바꿔주는 쿼리를 모든 단백질에 대해 적용하는 코드를 작성했다.



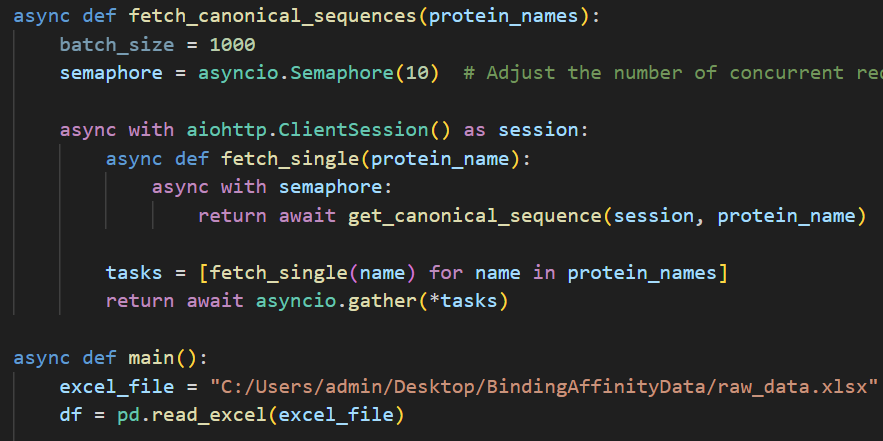


그림 1. data processing code 요약본

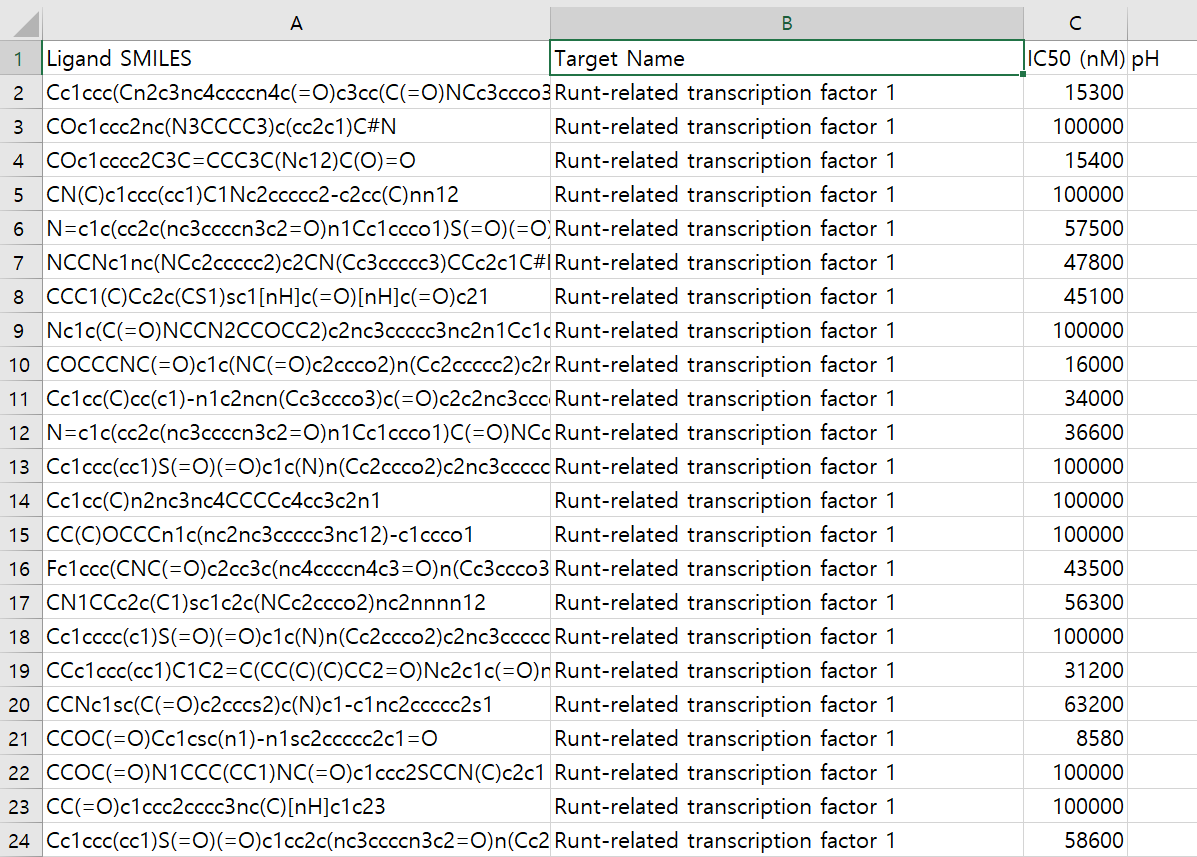


그림 2. labeling 전 dataset

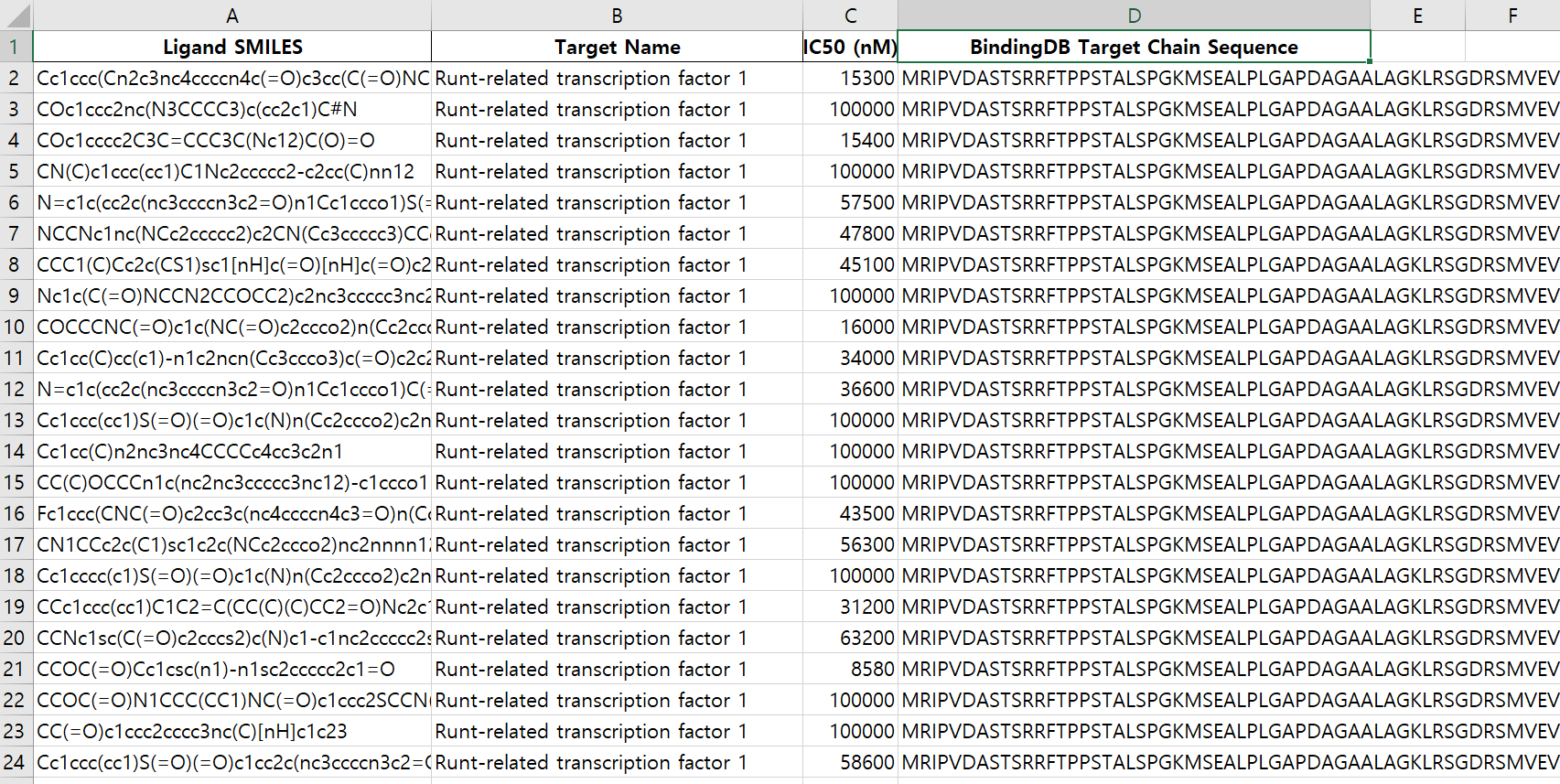


그림 3. labeling 후 dataset

* 주어진 데이터를 훈련, 검증, 테스트 세트로 분리하기 위해 각 행을 행의 ‘set’이 각각 ‘train’, ‘val’, ‘test’ 값을 가지는 데이터로 분리해 train, val, test 데이터 세트에 포함되는 인덱스를 저장한다.

|  |
| --- |
| df = pd.read\_csv(fp)  train\_idx = df[df['set'] == 'train'].index.values  val\_idx = df[df['set'] == 'validation'].index.values  test\_idx = df[df['set'] == 'test'].index.values |

표 1. 각 데이터셋에 해당하는 인덱스를 저장하는 코드

3. 모델 설계 및 구현

**3.1 모델 설계 상세**

주어진 논문을 참고하여 3개의 1D 컨볼루션 레이어를 가지는 CNN 블록을 구현한다. 이때 1D 컨볼루션 레이어가 지닌 필터의 개수는 등차적으로 증가한다. 이렇게 만들어진 두개의 CNN블록 중 하나는 SMILES 서열을 처리하고, 나머지 하나는 단백질 서열을 처리한다.

CNN 블록의 출력값은 ReLU 활성화 함수 및 드롭아웃이 존재하는 4개의 fully connected 레이어를 거쳐 최종적인 결합 친화도 값을 반환한다.

|  |
| --- |
| self.ligand\_conv = Conv1d(lig\_vocab\_size, channel, ligand\_kernel\_size)  self.protein\_conv = Conv1d(pro\_vocab\_size, channel, protein\_kernel\_size)  self.fc1 = nn.Linear(channel\*6, 1024)  self.fc2 = nn.Linear(1024, 1024)  self.fc3 = nn.Linear(1024, 512)  self.fc4 = nn.Linear(512, 1)  self.dropout = nn.Dropout(0.1)  self.relu = nn.ReLU() |

표 2. CNN블록 출력값과 Fully Connected 레이어

**3.2 모델 학습**

학습을 위한 최적화 알고리즘으로는 Adam, 손실 함수로는 MSE를 사용한다. 훈련 데이터로 생성한 DataLoader에서 각 epoch마다 훈련 손실과 검증 손실을 기록하여 가장 낮은 검증 손실을 가진 모델을 구해낸다. 이후 테스트 데이터를 사용해 최종 모델을 평가하고 결과를 도출한다.

|  |
| --- |
| if val\_loss < best\_val\_loss:  best\_val\_loss = val\_loss  best\_weights = self.model.state\_dict()  best\_epoch = epoch  self.logger.info('Best Model So Far in Epoch: {}'.format(epoch+1)) |

표 3. 가장 낮은 검증 손실을 지닌 모델을 도출하는 코드

4. 갱신된 과제 추진 계획

**4.1 갱신된 개발 프로세스 일정**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 5월 | | 6월 | | | | 7월 | | | | | 8월 | | | | 9월 | | | |
| 3주 | 4주 | 1주 | 2주 | 3주 | 4주 | 1주 | 2주 | 3주 | 4주 | 5주 | 1주 | 2주 | 3주 | 4주 | 1주 | 2주 | 3주 |
| 착수보고서 작성 | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 약물-표적  상호작용 이해 | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | 모델 기법 연구 | | | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  | 데이터 수집 및  전처리 | | |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | 모델 구현 | | | | | |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | 중간보고서 작성 | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  | 모델 성능 측정 및 최적화 | | | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 최종보고서 작성 및 발표 준비 | | |

**4.2 구성원 별 진척도**

|  |  |
| --- | --- |
| **이름** | **역할 분담** |
| **박한얼** | * 데이터 전처리 완료 * Hyperparameter 최적화 |
| **김선아** | * input dataset 훈련모델(Trainer) 구현 * 모델 성능 측정 |
| **임연후** | * Pytorch를 이용한 CNN 블록 구현 * FC layer모델 구현 |
| **공통** | * 모델 기법 연구 및 사전 지식 학습 * 보고서 작성 및 발표 |

5. 결론 및 추후 계획

**5.1 중간 결과 요약**

|  |  |
| --- | --- |
| 분류 | 진행률 |
| 데이터 전처리 | 완료 |
| 데이터 인코딩, 임베딩 | 완료 |
| CNN모델(Drug,Target의 representation 추출) | 완료 |
| Training모델(input dataset 훈련모델) | 완료 |
| FC layer모델(binding affinity prediction scoring) | 진행중 |
| Hyperparameter Optimization | 진행중 |

**5.2 수정된 계획과 추가 연구 방향**

계획대로라면 7월 말 ~ 8월 초 중으로 모델에 구현이 완료 됐어야 했기에 현재 prediction scoring(FC layer)모델을 완성시키는 것을 최우선 목표로 하고 있다. 모델 구현이 완료되면 다양한 dataset에 대해서도 비교를 진행할 예정이다.

이후, regularization과 overfitting을 최적화 시킬 수 있는 hyperparameter를 찾기 위해 다양한 메소드들을 실험해 볼 것이다. 우선은 Grid Search를 통해 찾는 것을 염두에 두고 있다.

**5.3 잠재적 문제점과 대응방향**

Overfitting을 막기 위해 K-fold cross validation과 CNN layer의 깊이와 개수 등의 hyperparameter 최적화를 최우선으로 할 것이다. 하지만, 더 정교한 모델을 위해 더 많은 dataset에 대하여 실험을 해봐야 하는데, 구현한 모델이 필요한 data 형식을 만들기 위한 data preprocessing 과정에서의 시간 소모가 크고, 과정이 복잡하기에 쉽지 않을 것으로 예상된다. 훈련 자체의 시간도 오래 걸릴 것으로 예상되는데, 이는 GPU나 클라우드 서버 컴퓨터를 사용해서 해결해보려 한다.