Code States

AI 부트캠프

Section 4 : Deep Learning Project

AI_06_강지호

프로젝트 목표

- **개인 유저의 미시청 영화 평점 미리 예측하기**

  개인 유저가 평가한 영화의 평점들과 타 유저가 평가한 영화들의 평점, 영화의 줄거리를 활용하여 미시청 영화에 대한 유저의 평점을 미리 예측해보는 것이 목표.

- **데이터 선정**

  kaggle의 'the-movies-dataset'
  https://www.kaggle.com/rounakbanik/the-movies-dataset

  270,896명의 유저가 평가한 평점 데이터 26,024,289개

- **userId**: 유저 ID
- **movieID**: 영화 ID
- **rating**: user가 평가한 movie의 평점
- **overview**: 줄거리

데이터

- **가설**

  영화 줄거리는 영화 내용을 담고 있으므로, 줄거리를 벡터화하여 임베딩하면 영화의 특성을 추출할 수 있을 것이다. 영화 줄거리, 유저의 영화 평점 정보를 사용한다면 아직 시청하지 않은 영화에 대한 평점을 미리 예측할 수 있을 것이다.

- **전처리**

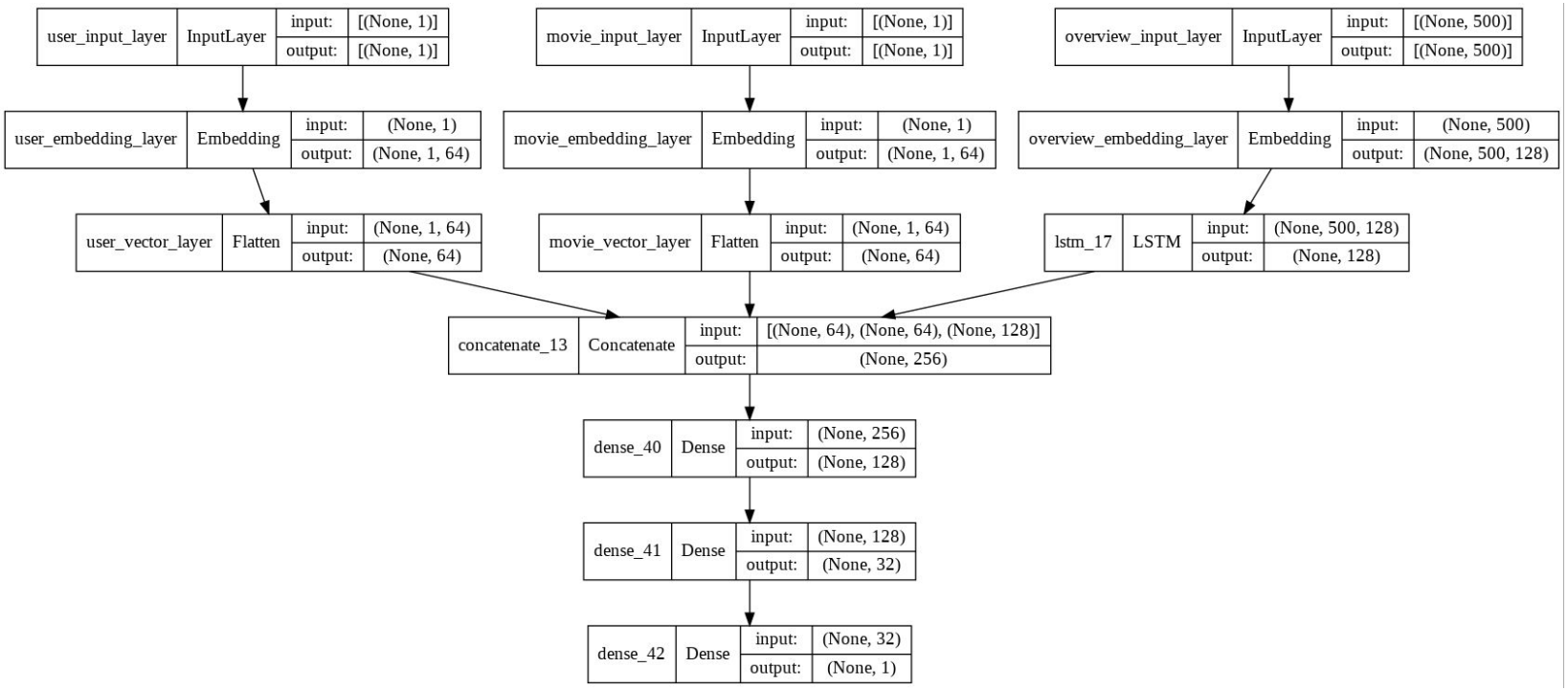
  - 중복, 결측치 제거
  - userId, movieId, rating만 있는 df에 각 movieId별 줄거리 내용 추가
  - NLP: 줄거리 벡터화를 위해 소문자화, 불용어 제거, 토큰화 후 sequences로 표현

- **Data**

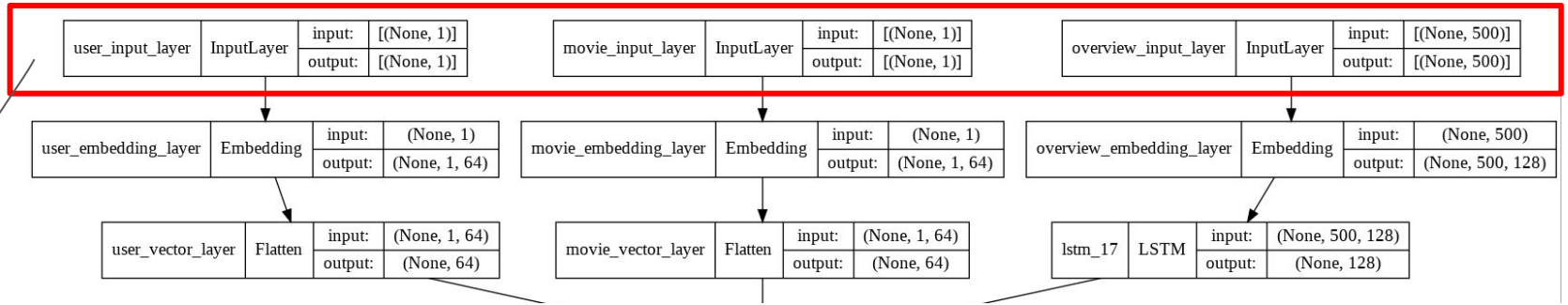
  Input: 한 명의 유저가 평가한 하나의 영화와 그 줄거리 (userID, movieID, movie_overview)

  Output: 한 명의 유저가 평가한 하나의 영화 평점 (Rating)

파이프라인 구축

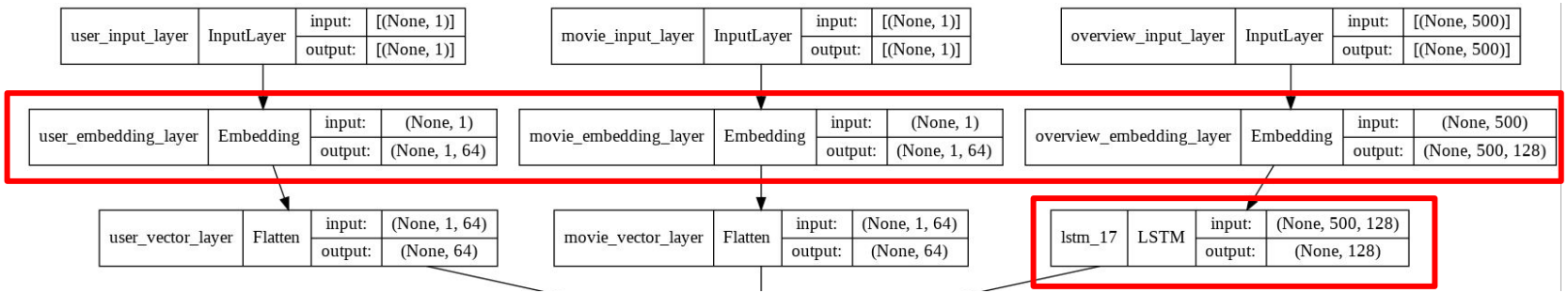

user_input_layer | InputLayer | input: [(None, 1)] | output: [(None, 1)]
movie_input_layer | InputLayer | input: [(None, 1)] | output: [(None, 1)]
overview_input_layer | InputLayer | input: [(None, 500)] | output: [(None, 500)]
user_embedding_layer | Embedding | input: (None, 1) | output: (None, 1, 64)
movie_embedding_layer | Embedding | input: (None, 1) | output: (None, 1, 64)
overview_embedding_layer | Embedding | input: (None, 500) | output: (None, 500, 128)
user_vector_layer | Flatten | input: (None, 1, 64) | output: (None, 64)
movie_vector_layer | Flatten | input: (None, 1, 64) | output: (None, 64)
lstm_17 | LSTM | input: (None, 500, 128) | output: (None, 128)
concatenate_13 | Concatenate | input: [(None, 64), (None, 64), (None, 128)] | output: (None, 256)
dense_40 | Dense | input: (None, 256) | output: (None, 128)
dense_41 | Dense | input: (None, 128) | output: (None, 32)
dense_42 | Dense | input: (None, 32) | output: (None, 1)

파이프라인 구축



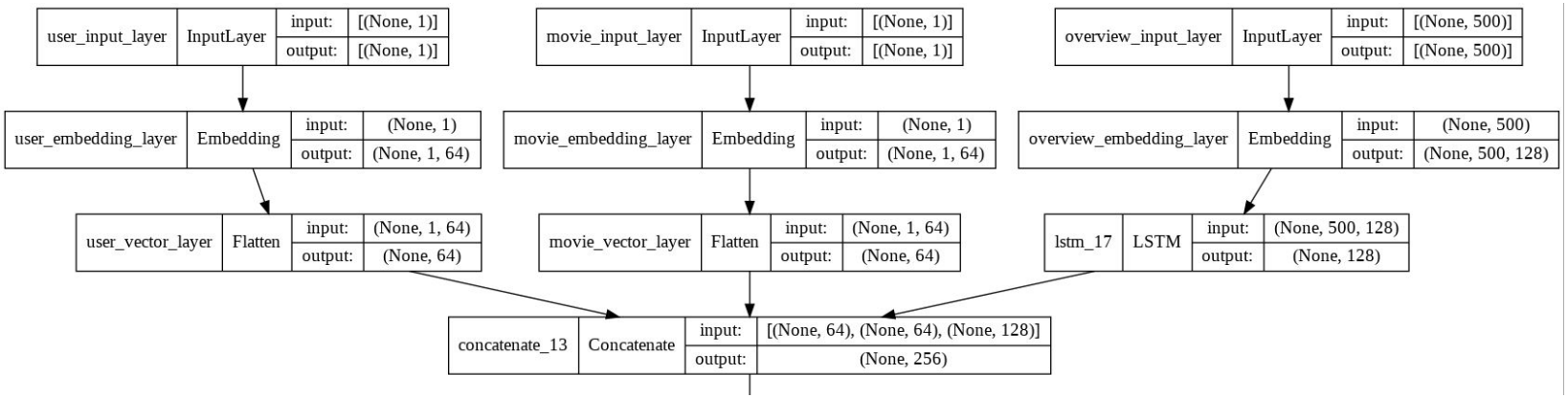
- **user_input_layer**: User Id
- **movie_input_layer**: movie Id
- **overview_input_layer**: movie_id에 해당하는 줄거리를 벡터화

파이프라인 구축



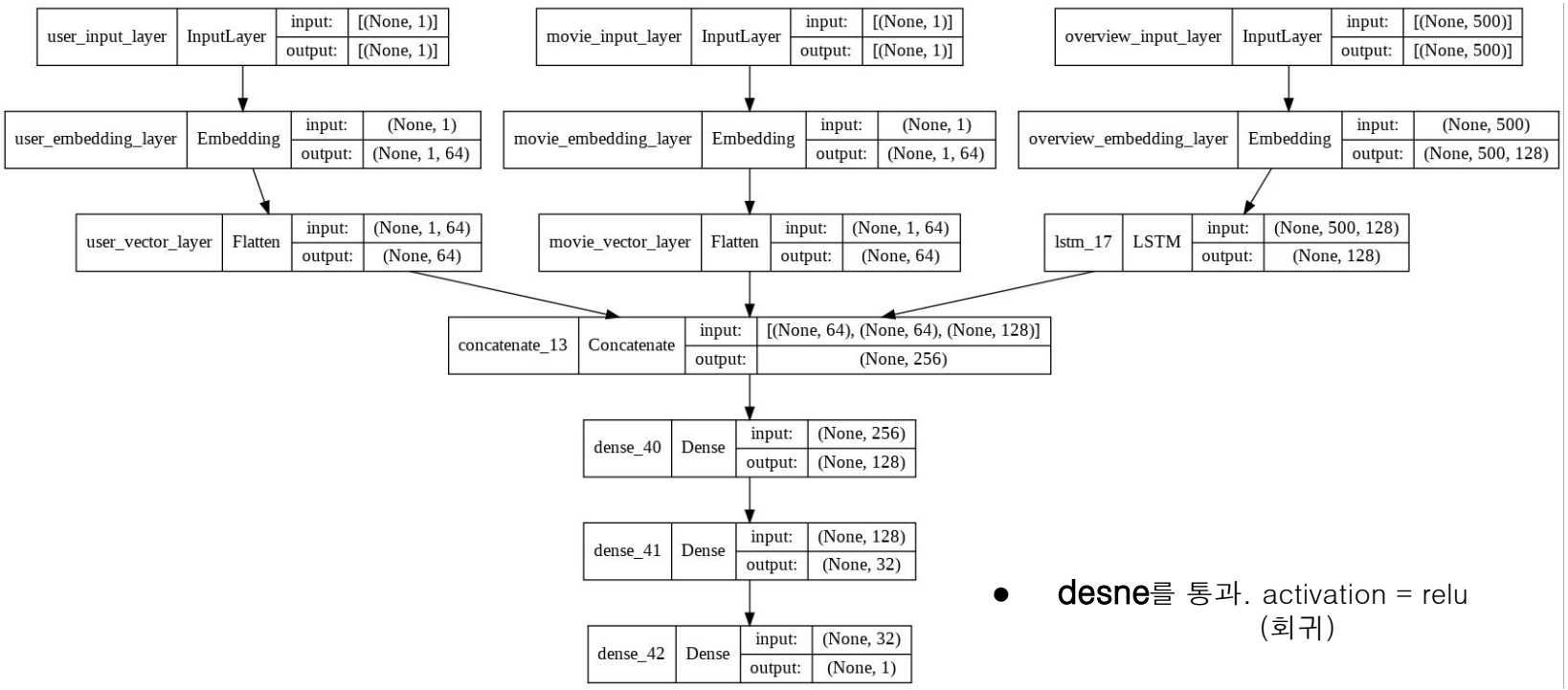
- **user_embedding_layer**: User Id를 64차원으로 임베딩하여 의미를 갖게 만듬
- **movie_embedding_layer**: movie Id를 64차원으로 임베딩하여 의미를 갖게 만듬
- **overview_embedding_layer**: 줄거리 벡터를 128차원으로 임베딩 후**LSTM** 통과하여 줄거리 특성을 파악

파이프라인 구축



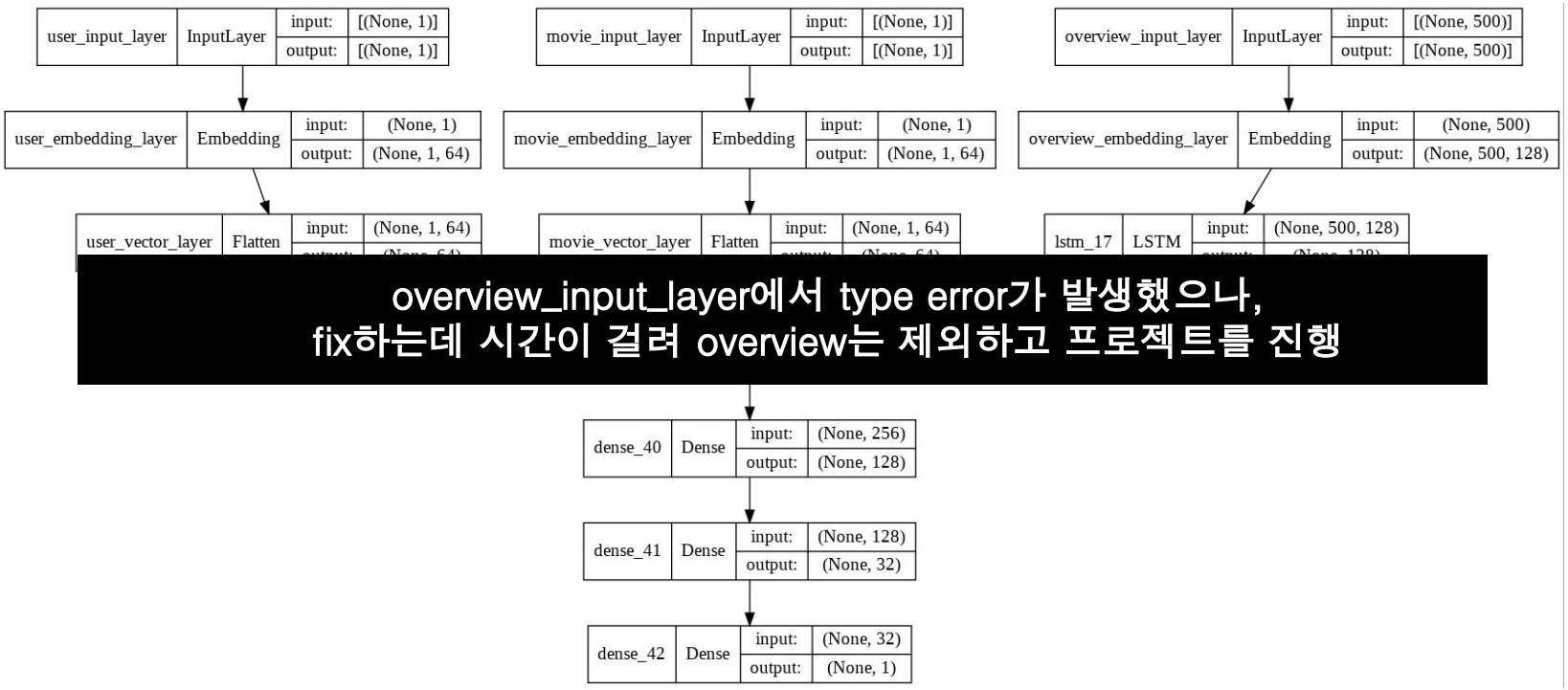
- concatenate: 3개의 input 값 형태를 맞춰준 다음, concat으로 합치기

파이프라인 구축

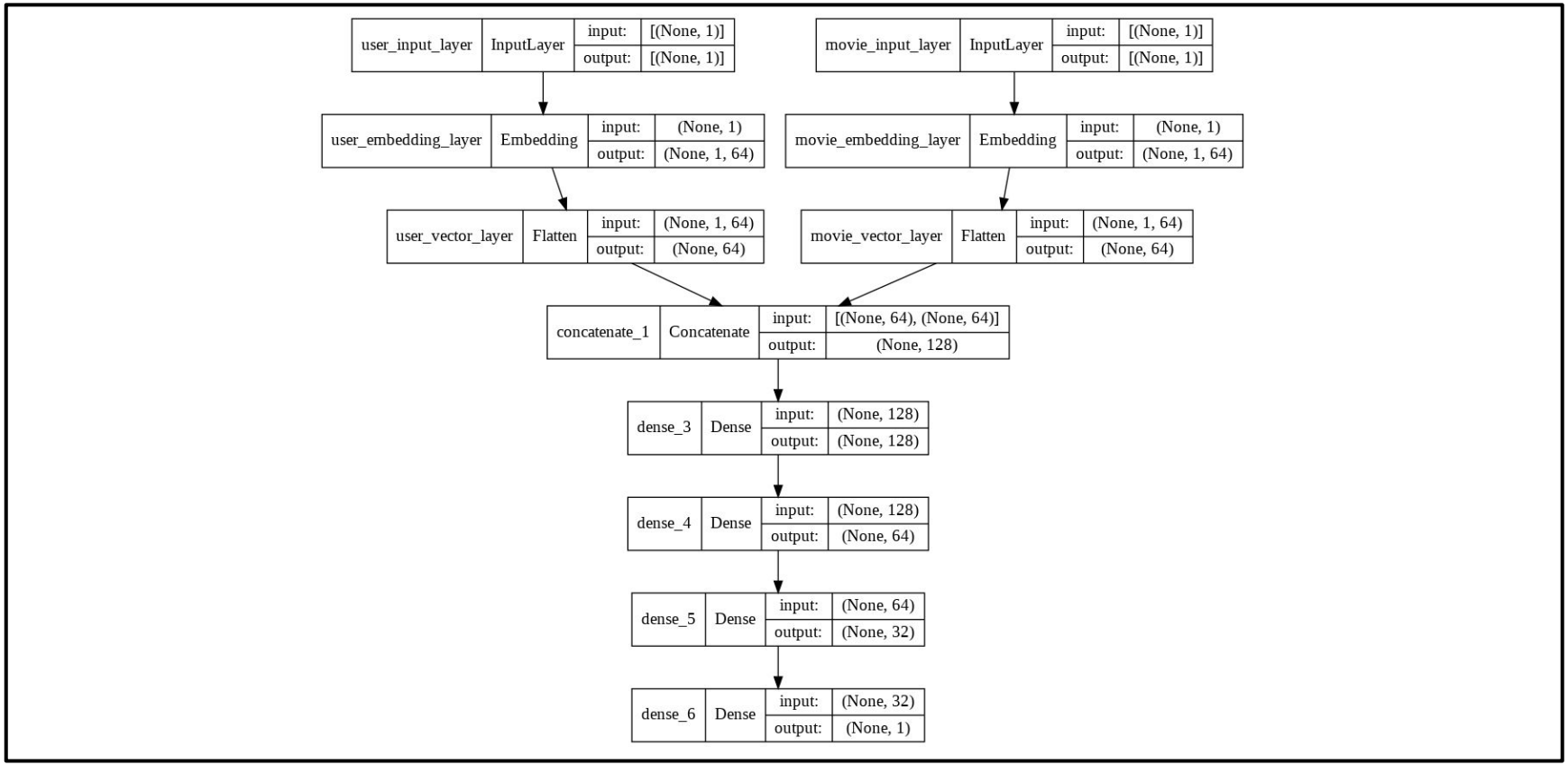


- desne를 통과. activation = relu (회귀)

파이프라인 구축


user_input_layer | InputLayer | input: [(None, 1)] | output: [(None, 1)]
movie_input_layer | InputLayer | input: [(None, 1)] | output: [(None, 1)]
overview_input_layer | InputLayer | input: [(None, 500)] | output: [(None, 500)]
user_embedding_layer | Embedding | input: (None, 1) | output: (None, 1, 64)
movie_embedding_layer | Embedding | input: (None, 1) | output: (None, 1, 64)
overview_embedding_layer | Embedding | input: (None, 500) | output: (None, 500, 128)
user_vector_layer | Flatten | input: (None, 1, 64)
movie_vector_layer | Flatten | input: (None, 1, 64)
lstm_17 | LSTM | input: (None, 500, 128)
overview_input_layer에서 type error가 발생했으나,
fix하는데 시간이 걸려 overview는 제외하고 프로젝트를 진행
dense_40 | Dense | input: (None, 256) | output: (None, 128)
dense_41 | Dense | input: (None, 128) | output: (None, 32)
dense_42 | Dense | input: (None, 32) | output: (None, 1)

파이프라인 구축


user_input_layer
InputLayer
input: [(None, 1)]
output: [(None, 1)]
movie_input_layer
InputLayer
input: [(None, 1)]
output: [(None, 1)]
user_embedding_layer
Embedding
input: (None, 1)
output: (None, 1, 64)
movie_embedding_layer
Embedding
input: (None, 1)
output: (None, 1, 64)
user_vector_layer
Flatten
input: (None, 1, 64)
output: (None, 64)
movie_vector_layer
Flatten
input: (None, 1, 64)
output: (None, 64)
concatenate_1
Concatenate
input: [(None, 64), (None, 64)]
output: (None, 128)
dense_3
Dense
input: (None, 128)
output: (None, 128)
dense_4
Dense
input: (None, 128)
output: (None, 64)
dense_5
Dense
input: (None, 64)
output: (None, 32)
dense_6
Dense
input: (None, 32)
output: (None, 1)

모델 평가

- **성능 1:** loss 감소가 거의 없음
  1 epoch mse: 0.2483
  20 epoch mse: 0.2348

- **성능 2**: R2: 0.07

- **한계점 및 보완 사항**
  – 시간 단축을 위해 2600백만여개의 데이터를 모두 활용하지 못했으므로 학습 데이터를 추가할 필요가 있음.
  – 모든 데이터를 활용한다면 과적합 방지를 위한 장치가 필요함.
  – 줄거리를 벡터화 시킨 데이터를 학습에 사용하지 못한것이 매우 아쉬움. 데이터 타입을 다시 맞추어서 학습하면 성능 향상에 도움이 될 것 같음.
  – 기존 추천시스템은 머신러닝에서 많이 사용되고 있는데, 딥러닝을 사용하여 문제를 해결할 경우 모델의 구조적 보완이 더 필요해 보임.