Implementing a Recommendation System

2018008331 박민경

summary

tensorflow를 활용한 딥러닝 모델로 Recommendation을 위한 알고리즘을 구현하여, 사용자의 movie ratings이 포함된 train data를 기반으로 하여 사용자의 ratings가 포함되지 않은 test data의 ratings를 예측한다.

Compilation method and environment

Windows(cmd): Python==3.8, tensorflow==2.2, numpy==1.18

execute

```
$ python recommender.py u1.base u1.test
```

argv[1] = training data file

argv[2] = test data file

check RMSE

```
$ PA4.exe u1
```

argv[1] = input file name

recommender.py

class Recommender

`__init__`

```
def __init__(self, train_path, test_path):
    self.train_path = train_path
    self.test_path = test_path
    self.train_data = None
    self.val_data = None
    self.test_data = None
    self.model = None
```

init 함수는 Recommender 객체 생성 시 초기화하는 함수이다. argument로 전달받은 train data와 test data의 path를 초기화한다.

set_data

```
def set_data(self):
    header = ['user_id', 'item_id', 'rating', 'timestamp']
    train_data = pd.read_csv(self.train_path, sep='\t', names = header)
    train_data.drop('timestamp', axis=1, inplace=True)
    train_data, val_data = train_test_split(train_data, test_size = 0.01)
    test_data = pd.read_csv(self.test_path, sep='\t', names = header)
    test_data.drop('timestamp', axis=1, inplace = True)

    self.train_data = train_data
    self.val_data = val_data
    self.test_data = test_data
```

set_data 함수는 argument로 전달받은 train data와 test data를 dataframe 형태로 변수에 저장한다. train data의 각 행은 `user_id`, `item_id`, `rating`, `timestamp`로 구성되어 있고, `timestamp`는 현재 model에서 고려하지 않을 것이므로 삭제하여 `train_data`에 저장한다. 또한 training 성능을 위해 `train_data`를 split하여 일부를 validation set으로 이용하기 위해 `val_data`에 저장한다. 마찬가지로 test data도 파일에서 읽어, train data와 같은 형태로 저장하는데 `test_data`는 추후 어떤 user의 어떤 item에 대한 rating을 예측할 것인지에 대한 정보를 제공하며, RMSE 계산에 사용된다.

make_model

```
def make_model(self):
    user_input = Input(shape=[1])
    user_embedding = Embedding(np.maximum(self.train_data.user_id.max() +
self.val_data.user_id.max(), self.test_data.user_id.max())+1
    , 10, input_length=1)(user_input)
    user_vec = Flatten()(user_embedding)

    item_input = Input(shape=[1])
    item_embedding = Embedding(np.maximum(self.train_data.item_id.max() +
self.val_data.item_id.max(), self.test_data.item_id.max())+1
    , 10, input_length=1)(item_input)
    item_vec = Flatten()(item_embedding)

    conc = Concatenate()([item_vec, user_vec])

    ly1 = Dense(64, activation = 'relu')(conc)
    ly2 = Dense(64, activation = 'relu')(ly1)
    out = Dense(1, activation = 'relu')(ly2)

    model = Model([user_input, item_input], out)

    self.model = model
```

make_model 함수는 rating prediction을 위한 model을 만든다. user_id와 item_id를 input으로 하여 rating을 prediction하는 것이다. network는 기본적으로 2개의 hidden-layer와 1개의 output layer를 가지는데, activation 함수로 가장 빠르고 간단한 relu를 사용한다.

clustering

```
def train_test(self):
    self.model.compile(optimizer=optimizers.Adam(1e-4)
    , loss=losses.mean_squared_error
    , metrics=[metrics.RootMeanSquaredError()])

    self.model.fit([self.train_data.user_id, self.train_data.item_id],
self.train_data.rating
    , batch_size=32, validation_data=([self.val_data.user_id, self.val_data.item_id],
self.val_data.rating
    , callbacks=[callbacks.EarlyStopping(patience=3, restore_best_weights=True)], epochs =
100, verbose = 1)

    output = Lambda(lambda x: K.round(x))(self.model.output)
    self.model = Model(self.model.input, output)
    self.model.compile(optimizer = optimizers.Adam(1e-4)
    , loss=losses.mean_squared_error
    , metrics=[metrics.RootMeanSquaredError()])

    self.model.evaluate([self.test_data.user_id, self.test_data.item_id],
self.test_data.rating)
    predictions = self.model.predict([self.test_data.user_id, self.test_data.item_id])
    predictions = predictions.astype(np.int64)
    self.test_data.rating = predictions
```

train_test 함수는 만들어 놓은 model로 ratings prediction을 수행하는 함수이다. train 시 user & item vector embedding이 업데이트되는데 이 때 validation_data로 `val_data`를 사용하여 earlystopping을 적용함으로써 overfitting을 방지한다.

test를 위해 output layer로 새로운 layer를 추가하는데, 이는 예측한 값을 [1, 2, 3, 4, 5] 값 중 하나로 맞춰주기 위함이다. 새로운 model은 이 전 model의 output으로 나온 결과에 round를 적용한 결과가 나오도록 하며, `test_data`의 각 행에서 `user_id`와 `item_id`를 통해 rating을 예측하고 그 결과를 `test_data`에 덮어쓴다.

`main`

```
if __name__ == '__main__':
    train_path = sys.argv[1]
    test_path = sys.argv[2]

    rcm = Recommender(train_path, test_path)
    rcm.set_data()
    rcm.make_model()
    rcm.train_test()
    rcm.test_data.to_csv(train_path+'_prediction.txt', sep='\t', index=False, header =
False)
```

argument로 train file path와 test file path를 입력받아, `Recommender`객체를 생성한다. 그리고 `set_data()`, `make_model()`, `train_test()`를 차례로 호출해 prediction 결과를 `test_data`에 저장하고, 결과 파일에 출력한다.

result (ex)

```
C:₩Users₩박민경님₩Desktop₩한양대학교₩4-1₩데이터사이언스₩과제₩LongTerm_RecommenderSystem>PA4.exe u1
the number of ratings that didn't be predicted: 0
the number of ratings that were unproperly predicted [ex. >=10, <0, NaN, or format errors]: 0
If the counted number is large, please check your codes again.

The bigger value means that the ratings are predicted more incorrectly
RMSE: 0.9943842
```

```
C:₩Users₩박민경님₩Desktop₩한양대학교₩4-1₩데이터사이언스₩과제₩LongTerm_RecommenderSystem>PA4.exe u2
the number of ratings that didn't be predicted: 0
the number of ratings that were unproperly predicted [ex. >=10, <0, NaN, or format errors]: 0
If the counted number is large, please check your codes again.

The bigger value means that the ratings are predicted more incorrectly
RMSE: 0.987269
```

```
C:₩Users₩박민경님₩Desktop₩한양대학교₩4-1₩데이터사이언스₩과제₩LongTerm_RecommenderSystem>PA4.exe u3
the number of ratings that didn't be predicted: 0
the number of ratings that were unproperly predicted [ex. >=10, <0, NaN, or format errors]: 0
If the counted number is large, please check your codes again.

The bigger value means that the ratings are predicted more incorrectly
RMSE: 0.9828021
```

```
C:₩Users₩박민경님₩Desktop₩한양대학교₩4-1₩데이터사이언스₩과제₩LongTerm_RecommenderSystem>PA4.exe u4
the number of ratings that didn't be predicted: 0
the number of ratings that were unproperly predicted [ex. >=10, <0, NaN, or format errors]: 0
If the counted number is large, please check your codes again.

The bigger value means that the ratings are predicted more incorrectly
RMSE: 0.9739097
```

```
C:₩Users₩박민경님₩Desktop₩한양대학교₩4-1₩데이터사이언스₩과제₩2021_ite4005_2018008331₩assignment4>PA4.exe u5
the number of ratings that didn't be predicted: 0
the number of ratings that were unproperly predicted [ex. >=10, <0, NaN, or format errors]: 0
If the counted number is large, please check your codes again.

The bigger value means that the ratings are predicted more incorrectly
RMSE: 0.9786981
```