Network and Deep Learning Analysis
Applied in
Movie Recommendation System

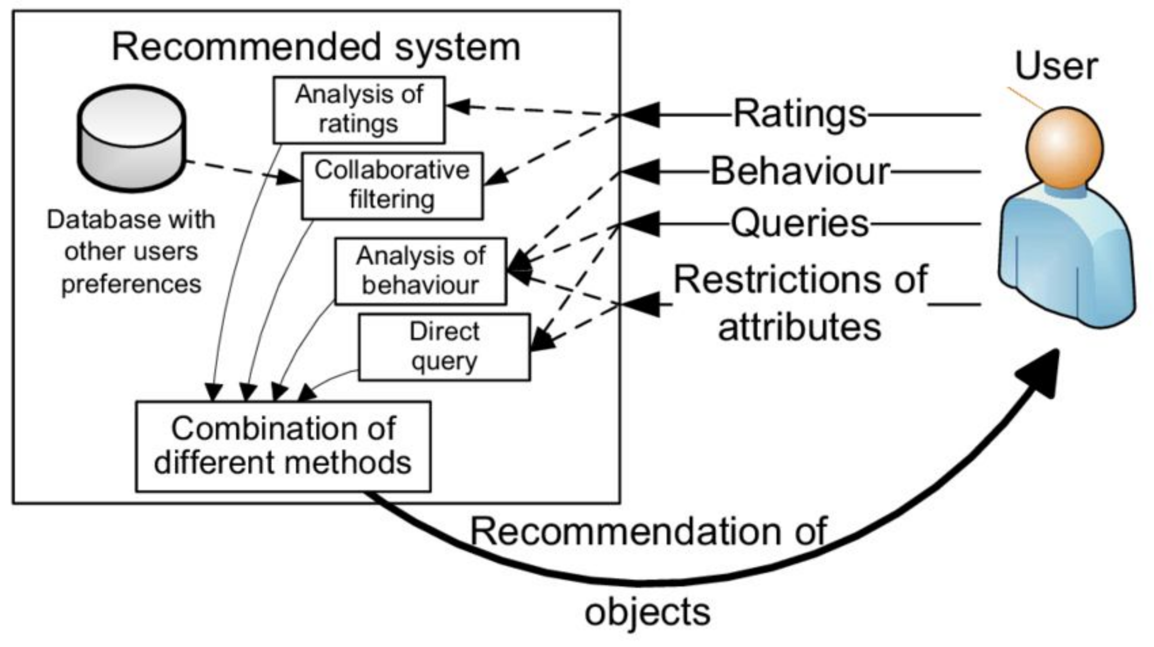
ANLY 590

Qian Leng, Xiaojing Ni,
Yihui Liu, Yutong Du,
Hanjun Wang

TABLE OF CONTENTS

01 Recommendation System

What is a recommendation system?



Model Selected

Model Fitting/Improvement

Results Comparison

Data Source

EDA

02

Data Source

movies.csv

- movieid
- title
- genres

tags.csv

- userid
- movieid
- tag
- timestamp

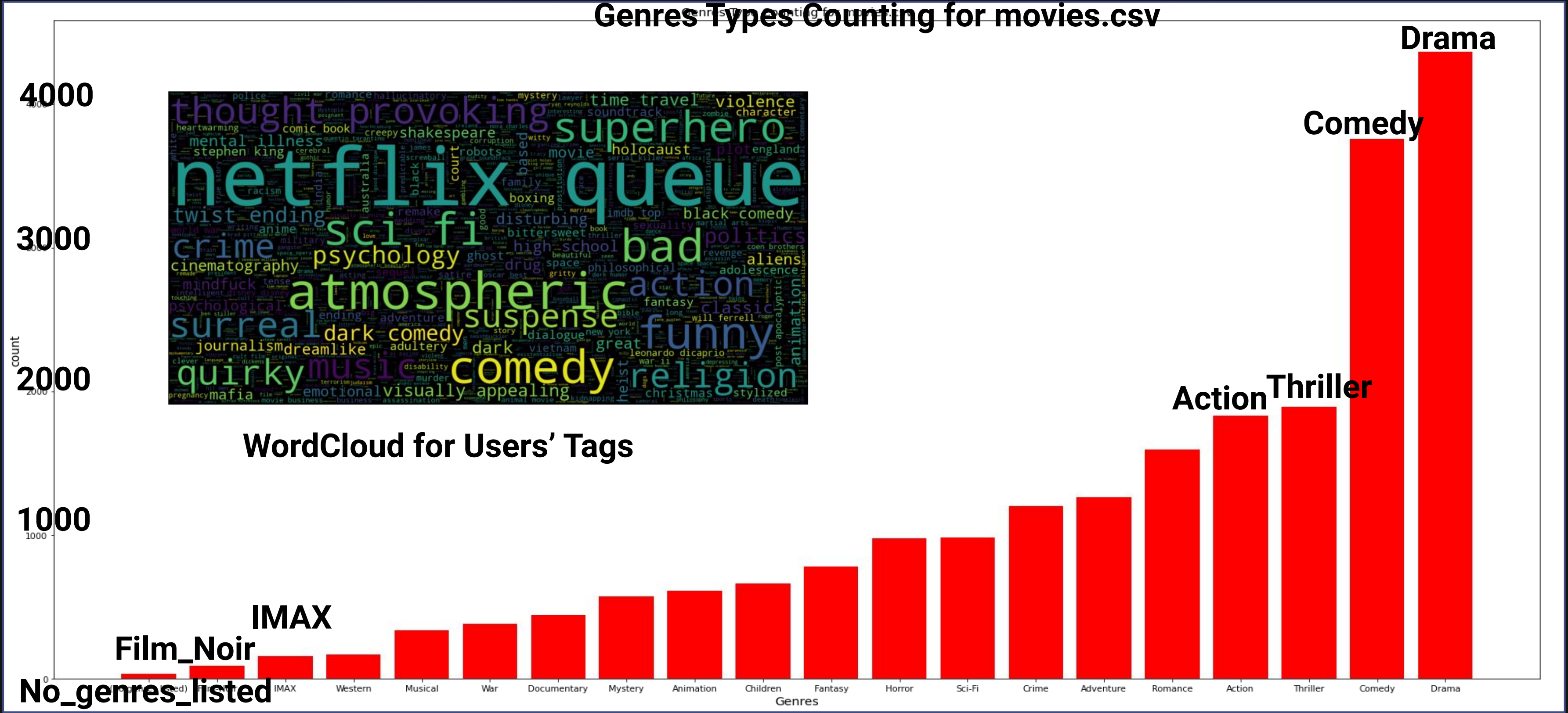
ratings.csv

- userid
- movieid
- rating
- timestamp

Link: https://grouplens.org/datasets/movielens/

Visualizations

Genres Types Counting for movies.csv



WordCloud for Users' Tags

Tables

Top 10 Productive Years

publish_year	count_by_year
2002	311
2006	295
2001	294
2000	283
2009	282
2007	282
2004	279
2003	279
2014	277
1996	276

movie.csv

Top 10 Movie Popular Years

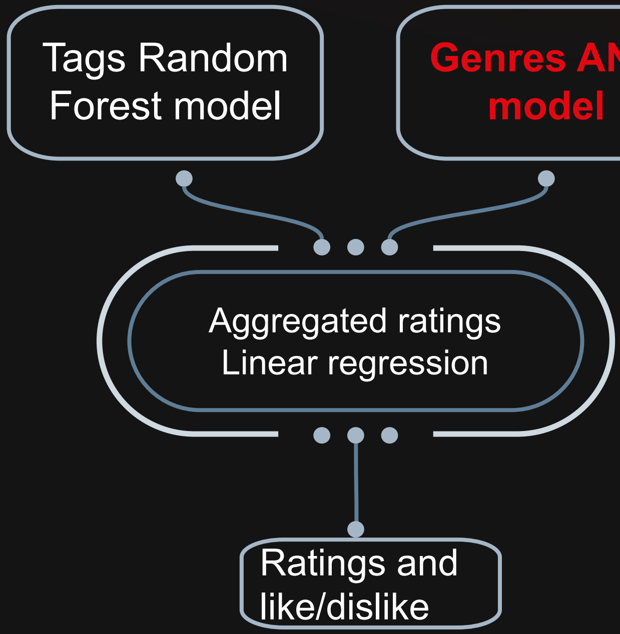
publish_year	count_by_year
1995	6143
1994	5296
1999	4537
1996	4509
2000	4268
2001	3914
1993	3741
1997	3643
2002	3642
1998	3556

Movie.csv + tags.csv

Content-based Model

03

| Would a user like a movie?




```python
## The goal of the model is to predict the rating the person would give to each movie
### There will be three inputs: user liked genres, user disliked genres, and movie genres
### The label will be the actual rating for the movie that the user gave it
user_liked_genres = keras.Input(shape= (20,))
user_disliked_genres = keras.Input(shape= (20,))
movie_genres = keras.Input(shape= (20,))

## Liked genres Input:
liked_input = keras.layers.Dense(20, activation= 'relu')(user_liked_genres)
liked_hidden_1 = keras.layers.Dense(50, activation= 'relu')(liked_input)
liked_hidden_2 = keras.layers.Dense(50, activation= 'relu')(liked_hidden_1)

## Disliked genres Input:
disliked_input = keras.layers.Dense(20, activation= 'relu')(user_disliked_genres)
disliked_hidden_1 = keras.layers.Dense(50, activation= 'relu')(disliked_input)
disliked_hidden_2 = keras.layers.Dense(50, activation= 'relu')(disliked_hidden_1)

## Movie genres Input:
movie_input = keras.layers.Dense(20, activation= 'relu')(movie_genres)
movie_hidden_1 = keras.layers.Dense(50, activation= 'relu')(movie_input)
movie_hidden_2 = keras.layers.Dense(50, activation= 'relu')(movie_hidden_1)

## Merging:
merged_model = keras.layers.concatenate([liked_hidden_2, disliked_hidden_2, movie_hidden_2])
merged_model_hidden_1 = keras.layers.Dense(150, activation= 'relu')(merged_model)
merged_model_hidden_2 = keras.layers.Dense(75, activation= 'relu')(merged_model_hidden_1)
merged_model_hidden_3 = keras.layers.Dense(50, activation= 'relu')(merged_model_hidden_2)

## Output Layer:
output_rating = keras.layers.Dense(1, activation= 'sigmoid')(merged_model_hidden_3)
```

[Dao Vang Github Repository](https://github.com/dao-v/Movie_Recommendation_System)

Content-based Model

Training loss (MSE) 0.039, validation loss 0.040.

PROS

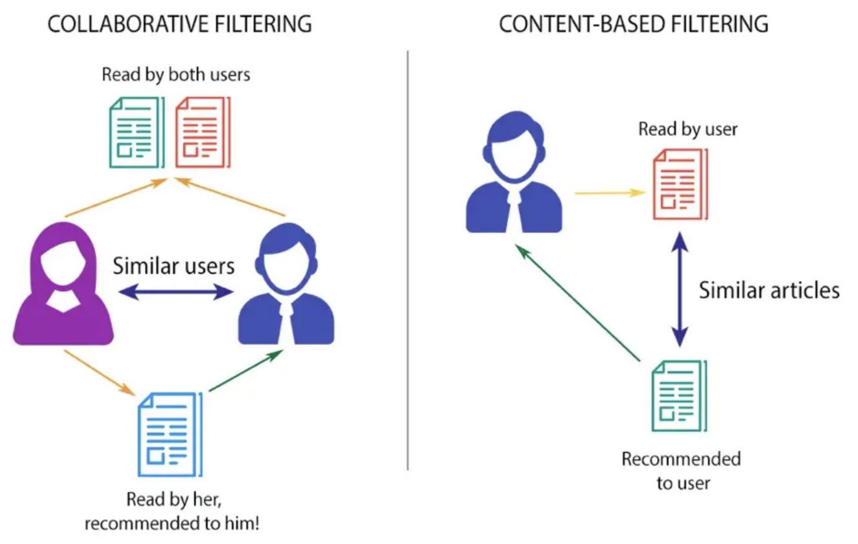
- No need for data on other users
- Can recommend to users with unique tastes
- Can recommend new & unpopular items
- Can provide explanations for recommended items by listing content-features that caused an item to be recommended (in this case, movie genres)

CONS

- Not support if there is a new user
- Does not recommend items outside a user's content profile.
- Unable to exploit quality judgments of other users
- Label definition (this model)

04 Collaborative Filtering Model

How Collaborative Filtering Recommender works?



Collaborative Filtering vs Content-Based Filtering

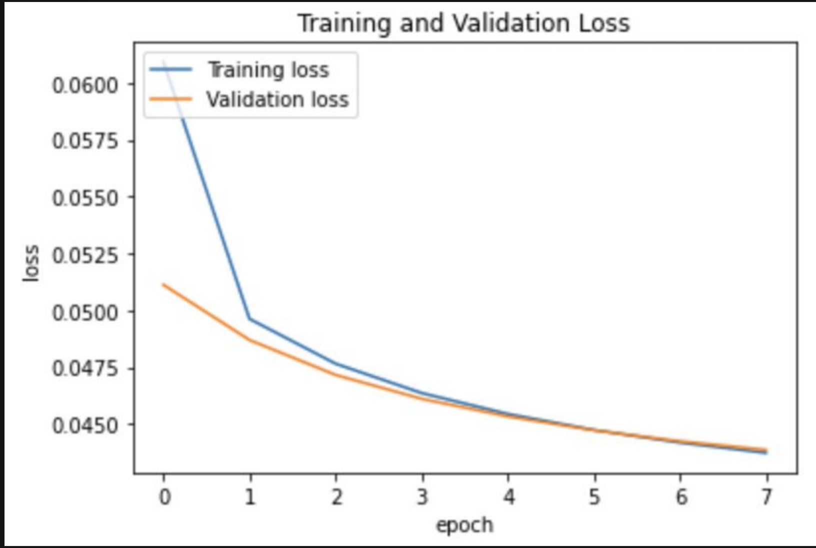
- Based on similarity in preferences, tastes and choices of two users
- Analyses how similar the tastes of one user is to another and makes recommendations on the basis of that
- Being able to do feature learning on its own

Ref: https://towardsdatascience.com/the-4-recommendation-engines-that-can-predict-your-movie-tastes-109dc4e10c52

About the model

```
Number of users: 138493, Number of Movies: 26744, Min rating: 0.5, Max rating: 5.0
```

- Test_size = 0.1
- Embedding_size = 100
- Loss: MSE
- Optimizer: RMSprop(learning_rate=0.0001)
- Training loss: 0.0437
- Validation loss: 0.0439



Ref: https://dl.acm.org/doi/pdf/10.1145/371920.372071

| Demo – Recommendation for a random user

Showing recommendations for user: 67064 – random user_id

Movies with high ratings from user

Twelve Monkeys (a.k.a. 12 Monkeys) (1995) : Mystery|Sci-Fi|Thriller
Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1964) : Comedy|War
Good, the Bad and the Ugly, The (Buono, il brutto, il cattivo, Il) (1966) : Action|Adventure|Western
Good Will Hunting (1997) : Drama|Romance
Moulin Rouge (2001) : Drama|Musical|Romance

→

Top 10 movie recommendations

Usual Suspects, The (1995) : Crime|Mystery|Thriller
Schindler's List (1993) : Drama|War
Godfather, The (1972) : Crime|Drama
Casablanca (1942) : Drama|Romance
One Flew Over the Cuckoo's Nest (1975) : Drama
Godfather: Part II, The (1974) : Crime|Drama
Fight Club (1999) : Action|Crime|Drama|Thriller
Amelie (Fabuleux destin d'Amélie Poulain, Le) (2001) : Comedy|Romance
City of God (Cidade de Deus) (2002) : Action|Adventure|Crime|Drama|Thriller
Dark Knight, The (2008) : Action|Crime|Drama|IMAX

Behavior Sequence Transformer Model

05

A Transformer-based recommendation system

Designed by Qiwei Chen in Behavior Sequence Transformer for E-commerce Recommendation in Alibaba

The BST model leverages the sequential behaviour of the users in watching and rating movies, as well as user profile and movie features, to predict the rating of the user to a target movie.

BST model aims to predict the rating of a target movie by accepting the following inputs:

- A fixed-length sequence of movie_ids watched by a user.
- A fixed-length sequence of the ratings for the movies watched by a user.
- A set of user features, including user_id, sex, occupation, and age_group.
- A set of genres for each movie in the input sequence and the target movie.
- A target_movie_id for which to predict the rating.

Content-based Model

BST model aims to predict the rating of a target movie by accepting the following inputs:

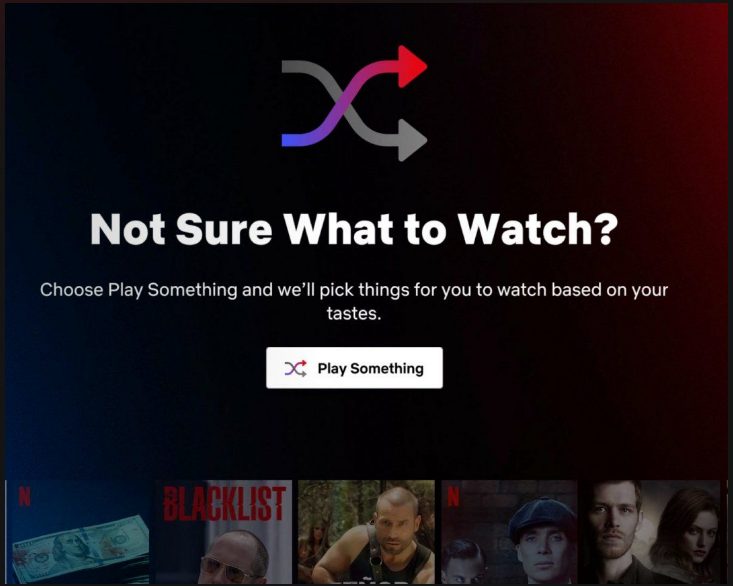
- Incorporate the movie features into the processing of the embedding to each movie of the input sequence and the target movie, rather than treating them as "other features" outside the transformer layer.
- Utilize the ratings of movies in the input sequence, along with the their positions in the sequence, to update them before feeding them into the self-attention layer.

Achieve a MSE at or around 0.7 on the test data.

06 Summary

| Model Comparison

Content Based Model
vs
Collaborative Filtering Model
vs
Behavior Sequence Transformer Model



Thanks