Project Proposal

COMS 6156 UNI : NS3886 Navinashok Swaminathan

Project Overview

For my project I am building a browser extension based music recommendation system in the domain of anime, that would be agnostic of the underlying streaming platform with feedback loops.

So for my project I would be building a chrome extension wherein users can login using their Anilist account. For every logged in user , data will be collected about the anime they watched and the ratings they gave for each of them. A user profile is created out of this information and anime songs(through youtube/soundcloud etc) are recommended based on this using multiple recommender models.

An area I'm keen to explore as part of the project is making use of hard/soft feedback and how this could be incorporated into the project. Hard feedback is of the nature - whether or not they like a particular song. Soft feedbacks are taken by tracking - for how long the video was played (say did they listen to the full song or did they just skip through it). These will be used to update recommendations in near-real time

User Interaction flow

So the chrome extension would have an anilist login button, which syncs and creates their user profile. Songs are recommended in sequence that they can skip through(right/left) . This would open up the respective youtube/soundcloud page of the song, There would be a like button next to the song being played for hard feedback. (This is independent of the underlying platform's like button due to api limit issues for each individual platform). ~~Based on their hard/soft feedback, the next sequence of songs to be recommended are updated dynamically in real time.~~

For every request a fetcnextbatch() call would fetch a sample of songs from each of the models based on user history(profile+feedback), and this would be used to fetch the next batch of songs to be played in sequence.
Revision : Instead of real time feedback loop based recommendations that would result in system overhead, I have decided to opt for a simpler solution as above. There would be some wait associated with users having to get the next batch of songs, but sufficient enough songs could be fetched in each call to create a playlist. So instead of a real-time song-to-song based feedback loop I have decided to go for a batch-to-batch based feedback loop.

Innovation and Motivation

This is innovative in the sense that there is currently no browser extension that does real time ani-song recommendation. The motivation for this project is my personal interest as I've long wished for a website/extension that intelligently recommends anisongs based on my taste.

Evaluation

Metrics used will be
1. Like Rate : videos liked (divided by) / videos displayed
2. Percentage of video watched : total duration for which users were engaged (divided by) / total video duration
3. Chrome extension rating

Competitive Model Evaluation
A comparative study on different models including the one presented in the midterm paper will be done by tracking average per-model metric score.

~~The way I plan to do this is by randomly routing recommendation requests to one of the deployed models, and per-model metric will be captured.~~

Every fetchnextbacth() call would get a sample of recommendation from each model. Every recommendation would have an associated id. The user's soft/hard feedback are registered for this id. Aggregation of feedbacks for all recommendation-id associated with each model is taken for per-model feedback.

For maximum utilization of feedback data , instead of per model feedback , The user engagement on items recommended by other models will also be fed as input to any random chosen model for every subsequent request.

Technology Stack ~~(To be Finalized)~~

Postgres, JavaScript, FastAPI with uvicorn, Docker, AWS - ECS , RDS

FallBack Options

If I face issues with anilist login, I will simply provide an interface wherein users can pick their favourite anime (since i already have the CSV of anime titles from MAL).

If I face issues with time constraints, I plan to cut the option of feedback loops and also do this just for one streaming platform - youtube.

~~Potential Enhancements (Under Consideration)~~
Far - Stretch Goal

Playlists :
A playlist like feature with public/private visibility wherein users can create a playlist of similar songs. This could be used to recommend similar playlists, or recommend songs that would be suited for the playlist, rather than recommending songs for the user profile as a whole. However since the app would be at the prototype stage, I expect the data to be sparse , and hence It wouldn't add as much value. Which is why I haven't finalized on this feature.