ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
Δ.Π.Μ.Σ. « Τεχνητή Νοημοσύνη»

ΕΡΓΑΣΙΑ ΣΤΟ ΜΑΘΗΜΑ: Εφαρμογές Τεχνητής Νοημοσύνης

ΑΚΑΔ. ΕΤΟΣ 2024-2025

Artist Recommendation System (HetRec2011 last.fm Dataset)

Αχιλλέας Οικονομόπουλος mtn2411

1. Εισαγωγή

Τα Recommender Systems αποτελούν πλέον κύριο κομμάτι πολλών εφαρμογών και πλατφορμών e-commerce, streaming, social media και άλλων. Πρόκειται για αλγορίθμους σχεδιασμένους να προσφέρουν μια πιο εξατομικευμένη εμπειρία στους χρήστες, «παράγοντας» προτάσεις βασισμένες στο ιστορικό αλληλεπίδρασής του με το σύστημα. Αφορούν λοιπόν ένα πρόβλημα πρόβλεψης βάσει υπαρχόντων δεδομένων, κάτι που τα συνδέει άμεσα με την έννοια της Μηχανικής Μάθησης.

Σε γενικές γραμμές μία προσέγγιση για την υλοποίηση Recommender System έγκειται σε 1 από 3 κατηγορίες:

- **Content-Based Filtering:** Εύρεση κάποιον χαρακτηριστικών που εκφράζουν πληροφορία για τα δεδομένα, αναπαράσταση των users και των items με αυτά τα χαρακτηριστικά, και εύρεση ομοιοτήτων.
- **Collaborative Filtering:** Άμεση χρήση των αλληλεπιδράσεων user-item για την εύρεση σχέσεων μεταξύ των χρηστών και των αντικειμένων.
- **Hybrid:** Συνδυασμός και των δύο τεχνικών.

Η εργασία αυτή αφορά την υλοποίηση ενός Hybrid Recommender System (Content-Based και Memory-Based Collaborative Filtering) για το Last.fm, και συγκεκριμένα για την παραγωγή προτεινόμενων καλλιτεχνών βάσει του hetrec2011-last.fm dataset [1].

2. Ορισμός προβλήματος

Ο στόχος των Recommender Systems είναι πάντα κοινός: Εύρεση patterns στα δεδομένα συμπεριφοράς των χρηστών ώστε να του προταθούν αντικείμενα / προϊόντα που θα τον ενδιαφέρουν [2]. Υπάρχουν πολλές συνιστώσες που συμβάλλουν στην επίτευξη αυτού του στόχου:

- Τι είδους πληροφορία / δεδομένα είναι διαθέσιμα για τον κάθε χρήστη;
- Πώς θα επεξεργαστώ τα δεδομένα ώστε να εκμαιευθεί πολύτιμη πληροφορία στο context της εφαρμογής;
- Ποια θα είναι η προσέγγιση / μοντέλο που θα χρησιμοποιηθεί και ποια είναι τα υπέρ και τα κατά της;
- Πώς θα μπορέσω να αξιολογήσω την προσέγγιση αυτή;

Όσον αφορά τα δεδομένα, το hetrec2011-last.fm παρέχει κυρίως πληροφορία για τις αλληλεπιδράσεις των χρηστών-καλλιτεχνών (user-artist listening counts και artists-associated tags). Πρόκειται για ένα dataset που αφορά ~1900 users, ~17.600 καλλιτέχνες, ~93.000 records για user-artist listening counts, ~186400 artist tags.

Σε αυτό το πλαίσιο το Recommender System καλείται να αντιμετωπίσει τις εξής κύριες δυσκολίες:

- **Χειρισμός implicit feedback:** Τα listening counts κυμαίνονται εντός του $[0, +\infty)$ και συνήθως ακολουθούν λοξή κατανομή, κάνοντας πολύ δύσκολη την ερμηνεία τους. Δεν υπάρχει ξεκάθαρη κλίμακα [Δε μου αρέσει – Μου αρέσει] όπως στις βαθμολογίες.
- **Sparsity:** Με 17.600 καλλιτέχνες και περίπου 50 interactions ανά user, η εύρεση ομοιοτήτων αποτελεί μεγάλο πρόβλημα.
- **Cold starts:** Το σύστημα να μπορεί να δώσει προτάσεις για «νέους» users για τους οποίους δεν έχει πολλά δεδομένα.
- **Πολυπλοκότητα:** Αν και σε σύγκριση με άλλα datasets, το hetrec2011-last.fm θεωρείται μεσαίου μεγέθους, ο αναποτελεσματικός χειρισμός του μπορεί να οδηγήσει σε αύξηση της πολυπλοκότητας του μοντέλου.

3. Παρουσίαση προσέγγισης/ μοντέλου

Όπως αναφέρεται και στην εισαγωγή, η υλοποίηση αφορά ένα Hybrid Recommender System, που κάνει χρήση τόσο τεχνικών Content Based, όσο και Collaborative Filtering. Αν και πρόκειται για την πιο απλή ενδεχομένως δυνατή υλοποίηση αυτής της μορφής και τα αποτελέσματα δεν είναι εντυπωσιακά, είναι μια αρκετά πιστή αναπαραγωγή των βασικών βημάτων της δημιουργίας ενός τέτοιου συστήματος.

Content-Based Filtering

Ως feature set εδώ χρησιμοποιούνται τα tags. Η ιδέα είναι λοιπόν ότι για τη δημιουργία artist profiles, θα αναπαραστήσουμε κάθε artist ως ένα σύνολο από tags που του έχουν αποδοθεί από χρήστες.

Ένα πολύ απλό representation θα ήταν να αναπαραστήσουμε τους artists ως το πλήθος των tags που τους αποδόθηκαν. Βάσει της πληροφορίας στα πλαίσια του μαθήματος και για μια πιο αποτελεσματική προσέγγιση, τα counts μετατρέπονται σε TFIDF, όπου τον ρόλο των terms αναλαμβάνουν τα tags, και αυτό των documents οι artists. Πιο συγκεκριμένα για την εύρεση των **artist profiles**.

i) Υπολογισμός κάθε artist-tag count ξεχωριστά
ii) Για κάθε artist, εύρεση του μέγιστου tag count του και διαίρεσή του από τα υπόλοιπα (Normalized tf)
iii) Για κάθε tag, υπολογισμός του πλήθους των artists που το περιλαμβάνουν
iv) Υπολογισμός του $idf_i = \log\left(num\_of\_artists \ / artists\_that\_contain(i)\right)$
v) Πολλαπλασιασμός των tf με τα εκάστοτε idf για το τελικό dataframe

Στον κώδικα που παραδίδεται η διαδικασία αυτή εκτελείται στο runtime καθώς το dataframe που προκύπτει χρειάζεται ~115MB αποθηκευτικού χώρου. Ιδανικά θα αποθηκευόταν μια συμπιεσμένη μορφή και θα μπορούσε να γίνει load όποτε χρειαστεί το σύστημα.

Τα **user profiles** προκύπτουν με τη βοήθεια των user profiles. Δεδομένων των interactions που είχε ο user με κάποιους artists, το user profile προκύπτει ως το weighted sum των interacted artist profiles με τα αντίστοιχα listening counts ως βάρη.

$$user\_profile_i = \sum_{j \in listened\_artists} \log\left(1 + listening\_count_{ij}\right) * artist\_profile_j$$

Επιλέχθηκε (αν και δοκιμάστηκε) να μη χρησιμοποιηθεί κάποιου είδους averaging ή normalization γενικότερα, για να διατηρείται η ιδέα της «ισχυρής προτίμησης» που μπορεί να έχει ένας user προς κάποιο artist. Εφαρμόστηκε όμως log transformation για να μειωθεί ο αντίκτυπος των πολύ ισχυρών outliers.

Ισχύουν τα ίδια ζητήματα που έχουν και τα artist profiles.

Εν τέλει υλοποιείται ο μηχανισμός recommendation, ο οποίος:

1) Λαμβάνει ως input (userID, k)
2) Ανακτά το user profile του και υπολογίζει τα similarities με το artist_profile_dataframe (παίρνουμε λοιπόν similarities με όλους τους artists)
3) Ταξινομεί σε φθίνουσα σειρά τους artists βάσει των similarities και επιστρέφει τους k πρώτους.

Collaborative Filtering

Ως προσέγγιση για το Collaborative Filtering κομμάτι του συστήματος, χρησιμοποιείται ένας memory-based item-item αλγόριθμος. Τα βήματα έχουν ως εξής:

1) Δημιουργία του artist-user dataframe με:

$$artist\_user\_df_{ij} = \log\left(1 + listening\_count_{ij}\right)$$

Λόγω του πιο αφηρημένου interpretability των listening counts ως user-item interaction metrics, δε γίνεται υπολογισμός απόκλισης από τους μέσους όρους.

2) Χρήση του k-NearestNeighbors της sklearn για τη δημιουργία 2 πινάκων
   a. artist-artist similarities
   b. artist-artist neighbors

   Σαν similarity metric χρησιμοποιείται το cosine similarity, ενώ k=50.

Εδώ και οι δύο πίνακες έχουν μέγεθος ~7MB, που επιτρέπει το save / load τους. Και οι δύο παρατίθενται στο φάκελο "data".

3) Μηχανισμός recommendation:
   a. Λαμβάνει ως input (userID,k)
   b. Υπολογίζει «score» prediction για κάθε artist με τον οποίο δεν έχει κάνει interact ο χρήστης **at runtime** (το γεγονός ότι έχουμε προϋπολογισμένους τους πιο similar γείτονες μας το επιτρέπει).

$$\hat{r}_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

   Όπου το $N(i;x)$ αποτελεί υποσύνολο των 50 πιο κοντινών γειτόνων που δόθηκαν προηγουμένως.

   c. Ταξινομεί σε φθίνουσα σειρά τους artists βάσει των predicted scores και επιστρέφει τους k πρώτους.

Hybrid Implementation

Στην ουσία υλοποιείται μια πολύ απλή μορφή hybrid implementation, όπου CBF και CF εναλλάσσονται, ανάλογα **πάντα** με τα αποτελέσματα που προκύπτουν από το CF. Το τελικό σύστημα λοιπόν:

1) Δέχεται ως input (userID,k)
2) Ανακτά τις προτάσεις και από τα δύο παραπάνω συστήματα
3) Ελέγχει αν ισχύουν κάποιοι περιορισμοί για να αποφασίσει πιο από τα 2 αποτελέσματα θα επιστρέψει στο χρήστη.
    a. Το CF έχει επιστρέψει αποτελέσματα (Υπάρχει περίπτωση ο χρήστης να μην κατάφερε να βρει 2 γείτονες που είχε ακούσει στο παρελθόν -> **Fallback για το sparsity**)
    b. Ο user έχει > 5 interactions (Πολύ δύσκολο ο CF να βρει ποιοτικά αποτελέσματα -> **Αντιμετώπιση Cold Start**)
    c. Το top recommendation του CF έχει score > 3.5 (threshold που υποδεικνύει την ποιότητα των recommendations)

    Αν κάποιος από αυτούς τους περιορισμούς δεν ισχύει, χρησιμοποιείται ο CBF, ειδάλλως επιστρέφονται τα αποτελέσματα του CF.

Σημείωση: Μία επιπλέον τροποποίηση γιαα την αύξηση ποιότητας των αποτελεσμάτων ήταν το CF υποσύστημα να λαμβάνει υπόψη μόνο τα predictions στα οποία έχουν συμβάλλει τουλάχιστον 3 γείτονες, δηλ. $|N(i;x)| > 2$ **. Η συνθήκη αυτή ενδεχομένως να περιόριζε τα αποτελέσματά του CF αν το χρησιμοποιούσαμε αυτούσιο, έχουμε όμως πλέον σαν fallback το CBF.**

4. Πειραματική μελέτη

A. Δεδομένα που χρησιμοποιήθηκαν

Τα κύρια δεδομένα που χρησιμοποιεί το σύστημα είναι:

users_artists.dat: Περιέχει όλα τα interactions (listening counts) μεταξύ users και artists.
users_taggedartists.dat: Περιέχει όλα τα tags που έχουν δοθεί στον κάθε artist.
artists.dat / tag.dat: Metadata και χρήση τους για ID consistency checks

Πιο συγκεκριμένα από τα παραπάνω αρχεία εξάγουμε:

- ID Sets για την αναγνώριση artists, users, tags
- User -> Artist Listening counts
- Artists -> Tags που τους έχουν προσάψει οι users
- Artist Metadata για καλύτερο visualization των αποτελεσμάτων

B. Preprocessing Δεδομένων

Εξασφάλιση data consistency στο dataset

Αφαίρεση κυρίως των artist IDs (και των αντίστοιχων εγγραφών) που εμπεριέχονται στο Artists – Tags Matrix αλλά για τους οποίους δεν είτε δεν έχει καταγραφεί κάποιο interaction, είτε δεν υπάρχουν διαθέσιμα metadata (δεν έχει νόημα να προταθεί ένα artist ID αν δεν ξέρουμε σε ποιον καλλιτέχνη αντιστοιχεί). Το ίδιο φιλτράρισμα εφαρμόζεται και για users και tags, για τα οποία όμως δεν υπήρχαν inconsistencies (καθαρά για λόγους scalability).

Train / Test split

Προτιμήθηκε μία μέθοδος split που εξασφαλίζει και σταθερό μέγεθος του test set και ότι κάθε user θα έχει ένα sample σε αυτό (εκτός αυτών με <5 interactions που τοποθετούνται εξ ολοκλήρου στο train set)

Λόγω του πολύ μεγάλου sparsity του artists-users dataframe επιλέχθηκε να λαμβάνεται **τυχαία** ως test data 1 sample ανά user, που αντιστοιχεί στο 2% του dataset.

Full dataset size: 92834
Train set size: 90957 -> 0.98
Train set size: 1877 -> 0.02

C. Μετρικές Αξιολόγησης

RMSE

Παρόλο που για τη συγκεκριμένη εφαρμογή το RMSE ίσως δεν είναι η καλύτερη μετρική αξιολόγησης του συστήματος, υλοποιήθηκε ενδεικτικά μία συνάρτηση που υπολογίζει το RMSE στο test set **μόνο για το CF κομμάτι του μοντέλου**. Συγκεκριμένα:

$$\mathrm{R}MSE_{CF} \approx 3.9802$$

Από τα 1877 δείγματα του test set, ο αλγόριθμος δεν κατάφερε να κάνει πρόβλεψη (ο user δεν είχε interaction με κανέναν από τους 50-NN) σε 279 από αυτά (~15% του test set).

Precision @ 10

Μία καλύτερη μετρική ειδικά για τα implicit data ενδεχομένως να είναι το Precision@10 (η οποία και πάλι ίσως καταλήγει να είναι περιττή λόγω της επιλογής του test set.) Το P@10 στην ουσία παίρνει τα 10 πρώτα recommendations του μοντέλου και τα συγκρίνει με τα περιεχόμενα του test set. Αν ένα recommendation βρίσκεται στο test set, θεωρείται ότι έχουμε 1/10 hits.

Εδώ τον ίδιο σκοπό εξυπηρετεί και το Hit Rate @ 10 και στην ουσία αυτό υπολογίζεται (αφού έχουμε 1 sample ανά user). Στην ουσία το μέγιστο δυνατό precision για το συγκεκριμένο test set είναι:

$$MAX\ P@10 = \frac{num\_of\_users * 1}{num\_of\_users * 10} = 1/10$$

Δεδομένου του MAX P@10 λοιπόν, εξάγονται οι εξής μετρικές:

Μοντέλο	P@10	P@10 / MAX_P@10
CBF	0.01742	0.1742
CF	0.01832	0.1832
Hybrid	**0.01896**	**0.1896**

Όπως ήταν ίσως αναμενόμενο, το CF αποδίδει καλύτερα του CBF, ενώ ο συνδυασμός τους αποδίδει ακόμη καλύτερα αποτελέσματα.

D. Σχολιασμός

Γενικότερα δεν υπάρχει πολλή πληροφορία για τη χρήση του συγκεκριμένου dataset ώστε να δοθούν συγκεκριμένα splits και baselines. Οι πιο σχετικές υλοποιήσεις [3][4] που αφορούν δεδομένα του last.fm χρησιμοποιούν πολύ μεγαλύτερα datasets, όπως το lastfm-360k (360k users / 18m total lines), και το lastfm-2B (2 billion events).

Τα κοντινότερα αποτελέσματα έρχονται από το [5], όπου για το ίδιο πλήθος users, με περίπου το 1/5 των artists και listening counts, καταγράφονται HR@10 μεταξύ 0.0605 και 0.2483.

E. Screenshots από την εφαρμογή

```
(env) (env) localadmin@test-achilleas:~/MSc-Applications$ python recommender.py
Loading datasets and creating the structures...
Done.

Loading tag counts...
100%|
Done.

Calculating TF for each tag-artist...
100%|
Done.

Finally calculating TFIDF for each tag-artist...
9718it [00:01, 9243.82it/s]
Done.

Collectiong all user-artist interactions...
92834it [00:02, 36494.45it/s]
Done.

Building the artists-users-interactions matrix...
92834it [00:02, 35111.88it/s]
Done.

Loading similarities and neighbors for the artists...
Done.

The recommendation system is ready to use!

Type in the user ID you want recommendations for:
```

Εικόνα 1: Αρχικοποίηση των δομών δεδομένων

```
Type in the user ID you want recommendations for: 1600

Here are your Top 5 Artists:
1. Kylie Minogue        15849              http://www.last.fm/music/Kylie+Minogue
2. Lady Gaga            6260               http://www.last.fm/music/Lady+Gaga
3. Christina Aguilera   3325               http://www.last.fm/music/Christina+Aguilera
4. Britney Spears       2787               http://www.last.fm/music/Britney+Spears
5. Avril Lavigne        2493               http://www.last.fm/music/Avril+Lavigne

Type in the amount of recommendations you'd like to get: 15

Recommendations:
1. Miley Cyrus          http://www.last.fm/music/Miley+Cyrus
2. Kelly Clarkson       http://www.last.fm/music/Kelly+Clarkson
3. Black Eyed Peas      http://www.last.fm/music/Black+Eyed+Peas
4. Jennifer Lopez       http://www.last.fm/music/Jennifer+Lopez
5. Glee Cast            http://www.last.fm/music/Glee+Cast
6. Taylor Swift         http://www.last.fm/music/Taylor+Swift
7. Selena Gomez & the Scene http://www.last.fm/music/Selena%2BGomez%2B%2526%2Bthe%2BScene
8. Leona Lewis          http://www.last.fm/music/Leona+Lewis
9. Nelly Furtado        http://www.last.fm/music/Nelly+Furtado
10. Cheryl Cole          http://www.last.fm/music/Cheryl+Cole
11. Paramore             http://www.last.fm/music/Paramore
12. Nicole Scherzinger   http://www.last.fm/music/Nicole+Scherzinger
13. Gwen Stefani         http://www.last.fm/music/Gwen+Stefani
14. Ciara                http://www.last.fm/music/Ciara
15. Fergie               http://www.last.fm/music/Fergie
```

Εικόνα 2: Χρήση του συστήματος και αποτελέσματα

5. Συμπεράσματα

Όπως αναφέρεται και προηγουμένως στην αναφορά, η εργασία κατέληξε να είναι μια υλοποίηση ενός Hybrid Recommender με αρκετά μικρό βαθμό πολυπλοκότητας σε σχέση με τις διαθέσιμες επιλογές στο πεδίο των Recommender Systems. Μέσα όμως από τη κατασκευή ενός τέτοιου συστήματος προκύπτουν κύρια συμπεράσματα:

Ένα πολύ μεγάλο κομμάτι της διαδικασίας είναι ο χειρισμός του dataset.
Ερωτήματα όπως:

- Ποια δεδομένα θα χρησιμοποιήσω από τα διαθέσιμα.
- Πώς θα τα προεπεξεργαστώ για να βελτιωθεί το αποτέλεσμα.
- Σε τι δομές θα τα αποθηκεύσω ώστε να είναι η πρόσβαση σε αυτά να είναι εύκολη, traceable, αλλά και αποδοτική ταυτόχρονα.
- Πώς θα χειριστώ τόσο sparse data.

κατέληξαν να καταλαμβάνουν το μεγαλύτερο κομμάτι χρόνου που χρειάστηκε.

Είναι πολύ σημαντική (ειδικά όσο τα μεγέθη των datasets λαμβάνουν τέτοιες διαστάσεις) η εξοικείωση με τις κύριες δομές δεδομένων που χρησιμοποιεί το Machine Learning.

Το implicit feedback χρειάζεται διαφορετική προσέγγιση.
Ένα άλλο κομμάτι που απασχόλησε πολύ είναι το πώς μπορούν να ερμηνευθεί το implicit feedback. Σε κάποιες περιπτώσεις χαμηλό implicit feedback μπορεί να σημαίνει έλλειψη ενδιαφέροντος, σε κάποιες άλλες όμως αυτός ο ισχυρισμός να πέφτει τελείως έξω. Σε συνδυασμό με τις μη φραγμένες τιμές που μπορούσαν να πάρουν τα listening counts και τις λοξές κατανομές τους, το implicit feedback προκάλεσε μεγάλο προβληματισμό στο πώς μπορούν να χρησιμοποιηθούν αποτελεσματικά οι τεχνικές που έχουμε γνωρίσει ως τώρα.

Πολλές εναλλακτικές και δυνατότητες τροποποίησης
Αν και οι μηχανισμοί συστάσεων μπορεί να φαίνονται αρκετά ευθείς στον τρόπο με τον οποίο λειτουργούν, στην ουσία αποτελούνται από πολλές επιμέρους διαφορετικές διαδικασίες οι οποίες επιδέχονται ξεχωριστή τροποποίηση.

- Τι στατιστικές μεθόδους θα χρησιμοποιήσω στις μετρήσεις μου και πώς η καθεμία ευθυγραμμίζεται με το στόχο του συστήματος (transformations, normalizations κ.τ.λ.)
- Ποια μετρική ομοιότητας εκφράζει καλύτερα τα δεδομένα μου.
- Ποιους αλγορίθμους μπορώ να χρησιμοποιήσω για να επιταχύνω τη διαδικασία
    - Π.χ: χρήση kNN
    - Εξετάστηκαν και dimensionality reduction, χρήση sparse matrices χωρίς κάποιο αποτέλεσμα
- Hyperparameter Tuning
- Επιλογή αρχιτεκτονικής Collaborative Filtering
- Επιλογή μεθόδου συνδυασμού CF και CBF

6. Βελτιώσεις – Περεταίρω Έρευνα

Διαφορετική προσέγγιση στην ερμηνεία των listening counts.

- Εξέταση binary μοντέλου όπου ύπαρξη interaction = με ενδιαφέρει, έλλειψη interaction (ή ύπαρξή της κάτω από ένα threshold) = δε με ενδιαφέρει.
- Ενίσχυση του binary μοντέλου με ένα μηχανισμό confidence, δηλαδή πόσο confident είμαι ότι στον χρήστη αρέσει το αντικείμενο

Χρήση διαφορετικού μηχανισμού για Collaborative Filtering.

- Η τρέχουσα υλοποίηση φαίνεται αδύναμη δεδομένου του sparsity του dataset.
- Χρήση κάποιου trainable Latent Factor μοντέλου και σύγκριση των αποτελεσμάτων.
- Διερεύνηση του Alternating Least Squares αλγορίθμου

Διερεύνηση εναλλακτικών για το συνδυασμό CBF, CF σε Hybrid

- Υλοποίηση μηχανισμού συγχώνευσης των καλύτερων αποτελεσμάτων του CF με των καλύτερων αποτελεσμάτων του CBF.
- Χρήση κάποιου μοντέλου για weighted aggregation των 2.

Validation – Better Evaluation

- Εισαγωγή διαδικασίας για validation στη διαδικασία κατασκευής για έγκυρο hyperparameter tuning.
- Καλύτερο train / test splitting.
- Διερεύνηση του ποιο evaluation metric ταιριάζει στη συγκεκριμένη υλοποίηση και γιατί (nDCG@10 ;).

Δημιουργία web-based UI για το recommender system

7. Αναφορές

[1] grouplens hetRec2011 last.fm dataset: https://grouplens.org/datasets/hetrec-2011/

[2] Rina Diane Caballar, C. S. (2024, Ιούνιος 6). What is a recommendation engine? Ανάκτηση από IBM: https://www.ibm.com/think/topics/recommendation-engine

[3] Recommender Systems Implementations (Tim Kartawijaya, Charlene Luo, Fernando Troeman): https://github.com/timjaya/lastfm

[4] Pavan Seshadri, Shahrzad Shashaani, Peter Knees (2024, 11 Sept.). Enhancing Sequential Music Recommendation with Negative Feedback-informed Contrastive Learning: https://arxiv.org/abs/2409.07367v1

[5] Weiping Song, Zhijian Duan, Ziqing Yang, Hao Zhu, Ming Zhang, Jian Tang (2019, 22 Jun.). Ekar: An Explainable Method for Knowledge Aware Recommendation: https://arxiv.org/abs/1906.09506