A Hybrid Peer-to-Peer Recommendation System Architecture Based on Locality-Sensitive Hashing

Alexander Smirnov, Andrew Ponomarev

St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences (SPIIRAS)
Motivation

- Most of the modern recommendation system designs are centralized. User data are collected and stored at one central point (server machine of server cluster)

Advantages:
- Broad spectrum of user preference models
- Offline analysis by service providers

Disadvantages:
- Quandary about rights on preference data
- Profile slicing
- Single point of failure
Motivation

- Decentralized (peer-to-peer) recommendation system: no central database.
- Approach to decentralization: «Omnia mea mecum porto»
- Advantages:
  - All user data are on user’s device
  - No profile slicing
  - No single point of failure
  - Improved privacy
- Disadvantages:
  - Severe limitations on prediction models
  - Network traffic and resource balancing needed
  - Likely security issues
- Goal: recommend items without exposing profile details
User-centric recommendation system

Movie recommendation system

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Roger</td>
<td>5</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Locality-sensitive hashing: idea

- Locality-sensitive hashing (LSH) – a widely used technique for probabilistic solution of k Nearest Neighbors problem. The idea is to hash multidimensional objects in such a way that similar objects are likely to have the same hash value.

- Formally, let $d_1 < d_2$ be two distances according to some measure $d$. A family $F$ of functions is said to be $(d_1, d_2, p_1, p_2)$-sensitive if for every $f$ in $F$:
  - If $d(a, b) \leq d_1$, then $\Pr[f(a) = f(b)] \geq p_1$
  - If $d(a, b) \geq d_2$, then $\Pr[f(a) = f(b)] \leq p_2$

- Random projections method for cosine distance ($d$)*

- AND-composition and OR-composition

*) P. Indyk, R. Motwani “Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality”
Locality-sensitive hashing: recommendations

- Collaborative filtering (CF) system – recommends items based on ratings assigned by other users.
- User profile – vector of normalized ratings \( r_{uj} \in [-1, 1], j \in \{0, M\} \), where \( M \) is the number of items.
- Algorithm idea:
  - Preparation: Encode each user \( u \) profile as \( L \) \( b \)-dimensional hash values \( h_i \) of and put each pair \((h_i, u)\) into corresponding hash table \( HT_i \).
  - Recommendations search for user \( v \):
    - Find values of \( L \) hash functions of \( v \)’s profile.
    - Look up each hash value in corresponding table.
    - Use found user identities to calculate exact similarities.
    - Use top-rated items of similar users as recommendations for \( v \).
Distributed locality-sensitive hashing

- Distributed Hash Table (DHT) – a structured Peer-to-Peer architecture allowing to maintain a distributed hash table with fast lookups
  - e.g., Chord: $O(\log n)$

- $L$ hash tables used in LSH nearest neighbor search are transformed into one distributed hash table where key is a tuple $(i, h_i)$

- Search for nearest neighbors is transformed into lookup in DHT of all keys $(i, h_i)$, where $h_i = f_i(\text{Profile}_v)$, $i \in \{1..L\}$
Distributed locality-sensitive hashing

Alice

\[ M \]

\[
\begin{bmatrix}
-1 & -0.5 & \ldots & 0.3
\end{bmatrix}
\]

\[ f_1 \]

\[ b \]

\[
\begin{bmatrix}
1 & 0 & 1 & \ldots
\end{bmatrix}
\]

(546…)

(710…)

(912…)

Roger

\[ M \]

\[
\begin{bmatrix}
-1 & 0 & \ldots & 0.2
\end{bmatrix}
\]

\[ f_1 \]

\[ b \]

\[
\begin{bmatrix}
1 & 0 & 1 & \ldots
\end{bmatrix}
\]

(546…)

DHT

Node 1

(1, 546…): [Alice, Roger]

(L, 912…): [Alice]

Node 2

(2, 710…): [Alice]
Shared state

- Problem! Need to share hash functions between nodes.
- Solution: breaking Peer-to-Peer design by the Master node
  - Not used in recommendation scenarios
  - No private data

The Master node

- Responsible for generating and holding the set of hash functions
Anonymization technique

- Original DHTs have security vulnerabilities:
  - Look up interception
  - Routing corruption

- Secure DHTs:
  - e.g. Octopus*

*) Q. Wang, N. Borisov “Octopus: A Secure and Anonymous DHT Lookup”
Architecture overview

- Similar nodes (neighbors)
- Peer-to-Peer (DHT)
- Physical network
Experiments

- Dataset: MovieLens 100k (943 users on 1682 items)
- Technique: 80/20 split
- Quality indicator: recall at fixed recommendations count
Conclusion

- Main objectives
  - User-centric distributed recommendation system
  - Limited ratings disclosure

- Open questions
  - Shared state in pure peer-to-peer design (epidemic protocols?)
  - Automatic parameters tuning
  - Context-awareness
  - High churn networks
and finally...

Thank you!

Questions are welcome!
Table of Contents

- Motivation
- Locality-sensitive hashing
- Distributed locality-sensitive hashing
- Shared state
- Anonymization technique
- Overall architecture
- Conclusion