Semantic Search in Cafebazaar

PageRank revisited in a different way

Yasmin Sarcheshmehpour
Cafe bazaar Search
Cafebazaar
Cafebazaar Search
What do we do?

List of apps
What do we know about apps?
How do we use this data?

Extract Keywords!
Search Example #1
Search Example #2
Search Example #3
Search Example #4

Users decide which apps are new!
There are two types of queries

1. Non-semantic queries
2. Semantic queries
Description problems

• Incomplete and not precise

Description benefits

• Support non-semantic queries
Besides description keywords, how can we find semantic relations between apps and queries?
The journey
Manual ways
Manual ways

• Precise

• Incomplete
Manual ways

- Precise
- Incomplete

Automatic solutions
Automatic solutions

For finding semantic relations between apps and queries
Automatic solutions
For finding semantic relations between apps and queries
Query-Apps Semantic relations

Which one is better? 🤔
Query-Apps Semantic relations

Which one is better? Users decide!
User feedbacks

Let’s see an example...
User feedbacks

Action: Search
User feedbacks

Action: View
User feedbacks

Action: Move back
User feedbacks

Action: Install
User feedbacks

Action: Move back
User feedbacks

Action: Install
How to use this data

View/Install
How to use this data

View/Install
How to use this data

View/Install
How to use this data

View/Install
How to use this data

Install
User feedbacks graph
User feedbacks graph

Normalize for queries
User feedbacks graph

Normalize for apps
User feedbacks graph

Number of apps: 150000
Number of queries: 800000
Number of edges: 2500000
Similarity Facts for apps
Similarity Facts

for queries
Predict missing edges
Predict missing edges
Predict missing edges
Semantic relation score
Semantic relation score

درخواست خودرو  تاکسي آنلاین

0.5 × 0.4 × 0.6
How can we infer indirect semantic relations from user feedbacks graph?
How can we find closest semantic queries to a single app?
Solution
Dijkstra
Dijkstra problems
Dijkstra problems

Multiple Paths
Random Walker
Random Walker

• Cover multiple paths
random walker slow
How do we use random walkers

1. Put 1000 random walkers on the starting app
2. Stop them after a number of steps
3. Count the number of random walkers on each query
Random Walker Animation
What do we want?

Same Result
What do we want?

Same Result

Too many steps!
Markov chain

- Unique Stationary Distribution
- Independent of starting vertex
Markov chain

- Unique Stationary Distribution
- Independent of starting vertex

We don't want this!
Random Walker Animation

Draft
Teleportation
teleportation slow
teleportation animation
PageRank
Difference with Classic PageRank

• Classic PageRank
  • Goal: Finding a measure for importance of websites
  • Teleport:
    • Guaranteeing the existence of answer
    • Avoid localization
Final Solution

**Problem:** Finding semantic queries for an app

**Solution:** Random walkers teleporting to starting point
How can we implement Random Walkers?
Implementation
Simulating Random Walkers

\[ M = \begin{pmatrix}
5 & 0 & 0 \\
2 & 2 & 0 \\
2 & 3 & 3 \\
1 & 1 & 0 \\
0 & 2 & 3 \\
\end{pmatrix} \]
Simulating Random Walkers

\[ M_a = \begin{bmatrix} 0.5 & 0 & 0 \\ 0.2 & 0.25 & 0 \\ 0.2 & 0.375 & 0.5 \\ 0.1 & 0.125 & 0 \\ 0 & 0.25 & 0.5 \end{bmatrix} \]

\[ M_q = \begin{bmatrix} 1 & 0.5 & 0.25 & 0.5 & 0 \\ 0 & 0.5 & 0.375 & 0.5 & 0.4 \\ 0 & 0 & 0.375 & 0 & 0.6 \end{bmatrix} \]
Simulating Random Walkers

\[ a = \begin{array}{c} 1000 \\ 0 \\ 0 \end{array} \]

\[ q = \begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \end{array} \]
Simulating Random Walkers

\[ a = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \]

\[ q = \begin{bmatrix} 521 \\ 213 \\ 197 \\ 69 \\ 0 \end{bmatrix} \]
Simulating Random Walkers

\[
\begin{array}{c}
a = \\
\begin{array}{c}
683 \\
221 \\
96 \\
\end{array}
\end{array}
\]

\[
\begin{array}{c}
q = \\
\begin{array}{c}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
\end{array}
\end{array}
\]
Simulating Random Walkers

- The result is still random and thus not precise.
- The process is inefficient
Simulating Random Walkers

• The result is still random and thus not precise.
• The process is inefficient

Infinite random walkers!
More Precise Solution

\[ a = \begin{array}{c}
1 \\
0 \\
0 \\
\end{array} \]

\[ q = \begin{array}{c}
0 \\
0 \\
0 \\
0 \\
0 \\
\end{array} \]
More Precise Solution

\[ a = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \]

\[ q = \begin{bmatrix} 0.5 \\ 0.2 \\ 0.2 \\ 0.1 \\ 0 \end{bmatrix} \]
More Precise Solution

\[ a = \begin{array}{c}
0.7 \\
0.225 \\
0.075
\end{array} \]

\[ q = \begin{array}{c}
0 \\
0 \\
0 \\
0 \\
0
\end{array} \]
Dynamic Programming

\[ a_0 = \langle 0, \ldots, 0, 1, 0, \ldots, 0 \rangle \]

\[ q_i = a_i \times M_a \]

\[ a_{i+1} = q_i \times M_q \]
Dynamic Programming

with teleportation

\[ a_0 = \langle 0, \ldots, 0, 1, 0, \ldots, 0 \rangle \]

\[ q_i = a_i \times M_a \]

\[ a_{i+1} = (1 - \alpha) \times q_i \times M_q + \alpha \times a_0 \]
Error

- Repeat the procedure $k$ times
- Error is proportional to $(1 - \alpha)^k$

$\alpha = 0.15 \quad \Rightarrow \quad k = 10$
Complexity

• Total order: $O(n_a \times k \times n_a \times n_q)$
Complexity

• Total order: $O(n_a \times k \times \mathcal{L})$

Sparse Matrix
Distributed processing

• Calculating similar queries is independent for each app
Is this sufficient?
The Full Framework
Why this is not sufficient?

• Not prefect outcome
What can we do?

Do it every day
Self-correction

User feedbacks decrease weights which are increased incorrectly.
Trends

• Special Occasions
• Nowrouz
• New apps
Conclusion
Our Result

Changes In Bounce

38.00% 40.00% 42.00% 44.00% 46.00%

24-10-2017 21-10-2017
Future Plan

1. Manipulate teleportation vector
   - Current similarity measures

2. Use other metrics
   - Search result index
   - View / Buy
Thank you!

Any questions? 🤔