Graph-based Approaches to Recommendation Systems

Unchitta Kanjanasaratool
unchitta@ucla.edu

UCLA Networks Journal Club

February 10, 2020

Outline

- Motivation
- Filtering-based recommenders
- Random walk-based recommenders
  - PageRank
  - Personalized PageRank (PPR)
  - SALSA
- Twitter: Cassovary, MagicRecs, and GraphJet
- Are recommendation systems polarizing?

Filtering-based recommenders

Content-based Filtering: Use information about the content–e.g. important words or indicator of features–to "filter" for products with high similarities to the query user (typically represented in the same feature space). Requires a similarity metric.

Collaborative Filtering: Filter for important information about the preferences of users or similarities of products using a "collaboration" of data on multiple agents and/or from multiple sources. (E.g. data on agents $1, ..., N$ help fill missing information about a random agent i.)

Filtering-based recommenders

Common approach: Gather user-rating in a matrix and perform matrix factorization (e.g. NMF) to find the latent space.

Filtering-based recommenders

Example (NMF): Suppose have a non-negative matrix $V \in \mathbb{R}^{m \times n}$, where columns are users and rows correspond to items. The (i, j)-th entry is user j's rating on item i. Can factorize into non-negative matrices $V \approx W \times H$.

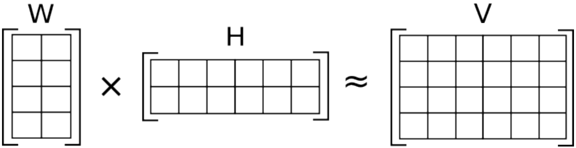

Figure 1: Figure taken from NMF Wikipedia page.

Filtering-based recommenders

For $W \in \mathbb{R}^{m \times k}$, we determine k, the number of "user segments."

Then $H \in \mathbb{R}^{k \times n}$ corresponds to how much each user "belongs to" each of the k segments.

Filtering-based recommenders

Suppose u is the query user.

Can either:

- Use W, H to pick the segment I that A belongs to (largest entry in H_u) and suggest items that "contribute" the most to this segment (largest entries in W_I), or
- Reconstruct the user-item matrix $\tilde{V} = WH$ and recommend the item with the highest (predicted) rating by u.

Note: Prune off items that u already bought and recommend the next highest-ranked items.

Filtering-based recommenders

In practice, MF approaches could run into problems of sparsity and cold start.

Many extensions have been proposed to take into account social trust data e.g. SoRec (Ma 2008) and NPL_Rec (Wang 2019).

Filtering-based recommenders

SoRec: Combine a directed social graph (weighted adjacency matrix) C with the user-item matrix R through the shared user latent feature space.

C and U are factorized simulataneously using PMF to give $U^T Z$ and $U^T V$, where U denotes the user latent feature space, Z factor matrix in the social graph, V item latent feature space.

Random walk-based recommenders

Random Walks

One of the most common algorithms for traditional search engines and recommenders is based on random walks on graphs.

Define $G = (V, E)$, where V is the set of vertices and E is the set of edges (i, j) between the vertices.

A random walker starts at a random node i in the graph and walks to a random neighbor j with probability $\frac{1}{k^{out}(i)}$ where $k^{out}(i)$ is the out-degree of node i. The process is repeated m times so that the walk is of length m (a parameter) steps, inducing a Markov chain.

Random Walks

Letting the walk goes on for long enough time, we can derive the probability distribution by corresponding that to the eigenvalue problem of the transition matrix P.

After derivations, we see that $\mathbf{1}$ is an eigenvalue of the transition matrix, and by the Perron–Frobenius theorem, that eigenvalue corresponds to the largest eigenvalue of P.

Random Walks & Eigenvector Centrality

This eigenvector corresponding to the largest eigenvalue of the transition matrix in the random walker problem is the same as the *eigenvector centralities* of nodes in a graph.

In this (recursive) view, it matters how many friends you have, but also how important your friends are (i.e. if they are also well-connected to other important nodes).

One can calculate the eigenvector centrality for every node through computing the eigenvector of the adjacency matrix corresponding to the largest eigenvalue.

PageRank

PageRank : Random walk + teleportation (to prevent a random surfer from getting stuck in a loop or on a dangling node).

- If the walker steps on a node with no outlink, it teleports to a random node in the graph with uniform probability.
- Otherwise, at any other node, invoke teloportation with probability $1 - \alpha$. ($\alpha = 0.85$ in the original PageRank paper).

PageRank - Important Properties

The Markov chain induced by the random walk with teleportation in PageRank is irreducible, so the stationary state distribution is unique (by the Ergodic Theorem).

I.e., the walker visits each node i of the graph a fixed fraction of the time $\pi(i)$ which depends on the graph structure and the value of α.

From PageRank to Personalized PageRank

Mathematically, we can define the Markov chain as

$$A = [\alpha P + (1 - \alpha)E]^T$$

where $P_{ij} = 1/deg(i)$ and $E = \mathbf{e}\mathbf{v}^T \in \mathbb{R}^{n \times n}$; **e** is a n-vector with all elements $e_i = 1$ and **v** is an n-vector whose elements are non-negative and sum to one.

Intuitively, we can take v to be uniform such that it corresponds to teleportation with uniform probability.

When **v** has a localized dependence on a node, we have "Personalized PageRank."

Personalized PageRank

If instead of teleporting with a uniform probability distribution over all nodes, we teleport to the source node or a seed set instead, we get what is referred to as Personalized PageRank (PPR).

Idea: It is personalized because the walker won't just stray away from the source node on the graph, so that the content stays somewhat relevant to the source node.

I.e., with probability $1 - \alpha$, reset to the source node.

Personalized PageRank

Using the same definition of A as before, we can take $\mathbf{v} = e_i$ where e_i is the i^{th} elementary vector. ($\mathbf{v}$ is called the *personalization vector*.) Then, we can compute the PPR by solving the eigenvector problem

$$\mathbf{x} = A\mathbf{x}.$$

$$\implies \mathbf{x} = (1-\alpha)(\mathbb{I} - \alpha P^T)^{-1}\mathbf{v}.$$

Letting $Q = (1-\alpha)(\mathbb{I} - \alpha P^T)^{-1}$, given any personalization vector, the corresponding personalized PageRank vector is $Q\mathbf{v}$.

In practice, computing both Q and $\mathbf{x}(\mathbf{v})$ can be computationally infeasible, but exists work on efficient algorithms.

Stochastic Approach for Link-Structure Analysis

Another algorithm in the random-walk family is the **Stochastic Approach for Link-Structure Analysis (SALSA)**.

SALSA (Lempel 2001) is an algorithm for identifying important "hubs" (e.g. that points to other authoritative nodes) and "authorities" (e.g. nodes that produce content).

Inspired by "mutually reinforcing relationships" (Jon Kleinberg) between hubs and authorities: good hubs point to many good authorities, and good authorities are pointed to by many good hubs.

SALSA

Consider a bipartite graph $G(V_1, V_2, E)$ whose two parts are hubs and authorities. To construct such a graph, suppose a collection $\mathcal{C}$ of sites built around a topic t is collected. Then,

1. $V_h = s_h \in \mathcal{C} | s \in \mathcal{C}$ and $deg^{out}(s) > 0$.
2. $V_a = s_a \in \mathcal{C} | s \in \mathcal{C}$ and $deg^{in}(s) > 0$.
3. $E = (s_h, s_a) | s_h \rightarrow s_a$.

Note that the link between hubs and authorities are represented as undirected.

SALSA

Idea: Authoritative sites in topic t should be linked to by many sites in the subgraph induced by $\mathcal{C}$, and so they should be visited more often in a random walk.

With mutually reinforcing relationships, hubs and authorities should form a prominent communities which would correspond to the dense portions of the bipartite graph.

SALSA

SALSA performs two distinct random walks on the 2 sides of G by walking length-2 paths in G, so that the walker crosses one side of the graph and back.

Nodes visited on each side correspond to a Markov chain, so that we have two separate chains to analyze and find good hubs and good authorities.

SALSA

The transition matrices for the two chains, one on the hub side (H) and the other on the authority side (A), are defined as follows:

$$H_{ij} = \sum_{k|(i_h,k_a),(j_h,k_a)\in G} \frac{1}{deg(i_h)} \times \frac{1}{deg(k_a)}.$$

$$A_{ij} = \sum_{k|(k_h,i_a),(k_h,j_a)\in G} \frac{1}{deg(i_a)} \times \frac{1}{deg(k_h)}.$$

SALSA

Like in PageRank, aperiodicity, irreducibility (induced by the assumption that G is connected), and symmetry of the chains allow us to use the ergodic theorem to find the principal communities of hubs and authorities:

$\Longrightarrow$ sites whose entries in the principal eigenvector of H (A) are the highest!

Twitter: Cassovary, MagicRecs, and GraphJet

PPR + SALSA : backbone of many services of Twitter that require relevance, ranking, and recommendation (see Gupta et al. 2010, Sharma et al. 2016).

2010 – Cassovary & "Who to Follow" service (for contact recommendations).

2013 – MagicRecs: Proof of the power of real-time recommendations.

2016 – GraphJet: Real-time, scalable platform-wide, recommendation services.

Twitter: "Who to Follow" (WTF)

Goal: To keep an active user base, need to be able to discover and engage with new connections.

How to do this? Similarities are a good measure but a user could be *"interested in"* following some other user, but she may not be *"similar to"* that user. (In fact similarities are used to build the "Similar to You" feature when visiting one's profile.)

Twitter: "Who to Follow" (WTF)

Suppose we are making a recommendation for user A on who to follow. The categorization of users that A might be "interested in" or "similar to" should suggest that we can partition the Twitter graph into two groups, thus inducing a bipartite subgraph.

How to partition and induce this bipartite subgraph? PPR!

Twitter: "Who to Follow" (WTF)

Cassovary first performs egocentric random walks (PPR) with dynamic parameters to populate A's **circle of trust** ($\sim$ 500 top-ranked nodes) "as a service."

The circle of trust forms the "hub" side of the bipartite graph in the same manner discussed early as in SALSA.

The "authority" side of the bipartite graph is populated with users that hubs follow.

Twitter: "Who to Follow" (WTF)

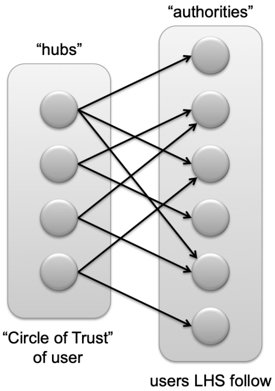

Figure 2: Illustration of our SALSA-based algorithm for generating user recommendations.

Figure 2: Illustration of the bipartite graph from the Twitter paper (2010).

Twitter: "Who to Follow" (WTF)

Now have the components to run a SALSA-type algorithm.

Motivation for SALSA-like algorithims for Twitter: ink structure of the Twitter network is directed and was different from much of the literature at the time, save for SALSA which studied similar structure (the WWW).

Twitter: "Who to Follow" (WTF)

Multiple iterations of SALSA are run and scores are assigned to nodes on both sides of the graph.

The nodes with the highest scores on the authority ("interested in") side are then suggested to A.

There is also a useful interpretation of the scores for the nodes on the hub side: nodes with high scores are likely to be similar to A. Using the principle of homophily, Twitter uses this ranking as a basis for their "similar to you" feature.

Twitter: GraphJet

After successive generations of graph processing engines (Cassovary, MagicRecs), Twitter arrives at **GraphJet**.

- Generalization of WTF to other social interactions.
- GraphJet stores and processes the Twitter network $G(U, T, E)$; U is the set of users, T is the set of tweets, and E encodes numerically certain kinds of relationships between the users and the tweets (e.g. liked, retweeted, etc.) over a temporal window.

Twitter: GraphJet

Using G, Twitter builds three recommendation algorithms to

1. Suggest personalized content in real time (e.g. on the home timeline of a user)
2. Suggest related tweets and promoted products
3. Respond to similarity queries

These algorithms are building blocks of the actual, deployed algorithms. Some outputs are also used further as inputs to machine-learning models.

Twitter: GraphJet (Personalized SALSA)

To recommend personalized content on a user's home timeline, Twitter performs a personalized variant of SALSA which utilizes real-time signals.

Starting with $G(U, T, E)$, SALSA algorithm is run, but with probability α the random walk resets to the query user (same idea as PPR).

The algorithm returns a ranked list of tweets which is then used to populate the query user's timeline.

Are recommendation systems polarizing?

Are recommendation systems polarizing?

Dandekar et al. (2013): bias assimilation and polarization in the context of the DeGroot model and illustration of how recommendation algorithms can be polarizing.

Assume that recommenders take in a bipartite graph $G = (V_1, V_2, E)$ where V_1 are users and V_2 items, and output $j \in V_2$.

Are recommendation systems polarizing?

Consider two books, RED and BLUE. For each individual $i \in V_1$, let $x_i \in [0, 1]$ be the fraction of RED books owned by i and $1 - x_i$ be that of BLUE books.

Definition. Suppose an individual $i \in V_1$ accepts a recommendation made by an algorithm. *The algorithm is polarizing with respect to i* if

1. when $x_i > \frac{1}{2}$, the probability that the recommended book was RED is greater than x_i, and
2. when $x_i < \frac{1}{2}$, the probability that the recommended book was BLUE is less than x_i.

Are recommendation systems polarizing?

Definition. We say that an individual $i \in V_1$ is *unbiased* if i accepts the recommendation [given by a recommender algorithm] with the same probability independent of whether the book is RED or BLUE. i is *biased* if

1. i accepts the recommendation of a RED book with probability x_i and rejects it with probability $1 - x_i$ and
2. i accepts the recommendation of a BLUE book with probability $1 - x_i$ and rejects it with probability x_i.

Are recommendation systems polarizing?

Consider simplified versions of PPR and SALSA:

Algorithm. SimplePPR

Input: $G(V_1, V_2, E)$, node $i \in V_1$
Parameter: large positive integer T.

1. Perform T 3-step random walks on G starting at i.
2. For each node $j \in V_2$, let count(j) be the number of random walks that end at node j.
3. Let $j^* := argmax_j$ count(j)

Output: j^*

Are recommendation systems polarizing?

Consider simplified versions of PPR and SALSA:

Algorithm. SimpleSALSA

Input: $G(V_1, V_2, E)$, node $i \in V_1$

1. Perform a 3-step random walk on G starting at i
2. Let the random end at node $j \in V_2$

Output: j

Are recommendation systems polarizing?

Theorem 1. Fix a user $i \in V_1$. In the limit as $n \to \infty$ and as $T \to \infty$, SimplePPR is polarizing with respect to i.

Theorem 2. Fix a user $i \in V_1$. In the limit as $n \to \infty$, SimpleSALSA is polarizing with respect to i if and only if i is biased.

Are recommendation systems polarizing?

Theorem 2. Fix a user $i \in V_1$. In the limit as $n \to \infty$, SimpleSALSA is polarizing with respect to i if and only if i is biased.

Proof (sketch). Assume $x_i > \frac{1}{2}$. Let p_r be the probability that SimpleSALSA recommends RED. First show $p_r > \frac{1}{2} and p_r \leq x_i$, and if $p_r \leq x_i$ then SimpleSALSA is polarizing iff i is biased.

Are recommendation systems polarizing?

Due to the reduced complexity of the algorithm,

$$
\begin{aligned}
p_r &= \sum_{j \in V_2, j_2 isRED} \mathbb{P}[i \xrightarrow{3} j] \\
&= \sum_{j_1 \in N(i), j_1 isRED} \mathbb{P}[i \xrightarrow{1} j_1] \sum_{j \in V_2, j isRED} \mathbb{P}[j_1 \xrightarrow{2} j] \\
&+ \sum_{j_2 \in N(i), j_2 isBLUE} \mathbb{P}[i \xrightarrow{1} j_2] \sum_{j \in V_2, j isRED} \mathbb{P}[j_2 \xrightarrow{2} j] \\
&= \ldots
\end{aligned}
$$

Are recommendation systems polarizing?

Invoke the Strong Law of Large Numbers (and lemmas proved in Dandekar et al 2013):

$$p_r \to x_i \left(\frac{1}{2} + 2Var(x_1) \right) + (1 - x_i) \left(\frac{1}{2} - 2Var(x_1) \right).$$

Since $x_1 > \frac{1}{2}$ and $Var(x_1) > 0$ by assumption, get

$$p_r > \frac{1}{2} \text{ and } p_r \leq x_i.$$

Are recommendation systems polarizing?

First suppose i is unbiased. Let p be the probability that i accepts the recommendation. Then the probability that the recommended book was RED given i accepted the recommendation is given by

$$\frac{p_r p}{p_r p + (1 - p_r)p} = p_r \leq x_i$$

so SimpleSALSA is not polarizing if i is not biased.

Are recommendation systems polarizing?

Now suppose i is biased. That is, i accepts RED with probability x_i and BLUE with probability $1 - x_i$. Then the probability that the recommended book was RED given i accepted the recommendation is given by

$$\frac{p_r x_i}{p_r x_i + (1 - x_i)(1 - p_r)} > \frac{p_r x_i}{p_r x_i + p_r(1 - x_i)} = x_i$$

since $p_r > \frac{1}{2}$. So *SimpleSALSA is polarizing if i is biased.*

Summary