Bringing Focus to Social Media Applications

Brian J. Karlberg[1*], Samuel R. Schreiber[1*]

[1]Washington State University

This paper defines a metric for determining the relatedness between pairs of nodes located in a hierarchical structure, and uses this metric to define the amount of focus in a social network, where focus is defined as the amount of variability in skills and interests which individuals have within a network. From this relatedness metric, a follower recommendation algorithm is proposed to address the tendency for social media networks to lose focus as they become increasingly populated and more dense.

Introduction

In 2017, there were 2.46 billion users subscribed to at least one social media network (https://www.statista.com/sta of-worldwide-social-network-users/). Social media has become an integral part of how individuals obtain information on a day-to-day basis. Algorithms have been implemented to aid individuals in constructing their personal social network by recommending others to connect with based on mutual friends or followers, proximity, and other information. These algorithms tend to produce social networks that contain very high variability in interests and skills between users, while favoring networks in which any two users have a high probability of knowing each other personally. In order to have a discussion with maximum engagement within a social network, the discussion must center around a subject which is familiar to the greatest amount of users as possible. Otherwise, the discussion is liable to be seen as esoteric (and therefore irrelevant) to users ignorant of the particular subject, leading to sub optimal engagement. Therefore, due to the aforementioned high variability of interests innate to social media networks, the deepest subjects which are familiar to the greatest number of individuals within most networks tend to be relatively trivial, while having engaging conversations related to more esoteric subjects becomes more cumbersome and therefore less efficient. In this paper, a solution is proposed to minimize variability of interests across social media networks as they expand in an effort to maximize community engagement with relatively (relative to the average user in the network) non-trivial discussions.

Subject Hierarchy

Subjects tend to vary in their levels of generality, where their generality can be viewed as the number of sub fields which can be found within that particular subject. Take sports for example. This is a general subject, encapsulating sub fields such as football, track, baseball, and so on. Thus, it follows that if we were to select any given sub field from a parent subject, the chosen sub field's generality would be less than that of the parents. To continue with the sports example, we can say that track is less generic than sports, since sports subsumes track, and hurdling is less generic than track, since track subsumes hurdling. Then, it follows that the set of all subjects can be represented in a hierarchical structure, where deeper levels within the hierarchy corresponds to less general, and therefore more focused, subjects.

Relatedness Between Two Nodes in a Hierarchical Structure

Suppose we have a general tree T with a set of nodes N. Let n be a node such that $n \in N$, and E_N be the set of all outgoing edges from node N such that $e_{N,M} \in E_N$, and $e_{N,M}$ is the edge connecting node N to some node M. Furthermore, let α be a value in the range [0, 1], and β

be another value in the range [0, 1] such that $\alpha \geq \beta$, and LCA[N_0, N_1] be the least common ancestor between N_0 N_1. Finally, let Dist[N_0, N_1] be the minimum distance between nodes N_0 and N_1. Then, we define the relatedness between two nodes in N as follows:

$$R[N_0, N_1] = \texttt{Dist}[\texttt{LCA}[N_0, N_1], N_0]^{\alpha} \cdot \texttt{Dist}[\texttt{LCA}[N_0, N_1], N_1]^{\beta} \text{ (Equation 1.1)}$$

Notice that α may be conceptualized as the penalty for walking up the hierarchy, where β can be defined as the penalty for traveling down the hierarchy.

Also, we have:

$$\texttt{Dist}[N_0, N_1] = \texttt{Dist}[\texttt{LCA}[N_0, N_1], N_0] + \texttt{Dist}[\texttt{LCA}[N_0, N_1], N_1] \text{ (Equation 1.2)}$$

In other words, the sum of the two nodes distances from their least common ancestor is equal to the minimum distance between two nodes within the hierarchy. Equation 1.1 may efficiently be calculated by using the Farach-Colton and Bender Algorithm to compute the lowest common ancestor in constant time.

Recommendation Algorithm

Suppose that F_u^1 represents the set of users that user u is following in a social network, where u is a type of node compatible with Equation 2.1. Further define F_u^k as the set $\{i | (V = \{v | v \in F_u^{k-1} \wedge k \geq 2\}) \wedge (\forall j \in V, i \in F_j^1)\}$. For example, F_u^2 would represent the set of all users who are being followed by the set of users u is following. Then, the number of mutual followings that a user u has with a user v is

$$M[u, v] = \sum_{\forall i \in F_u^1} \begin{cases} 1 & \text{if } v \in F_i^1 \\ 0 & \text{otherwise} \end{cases} \text{ (Equation 2.1)}$$

Next, suppose that each user u contains a set of attribute references A_u where $a_i \in A_u$, and each attribute a resides in a hierarchy H. The relatedness between two users may be defined as:

$$L[f, i, j] = f_{a_0 \in A_i, a_1 \in A_j}[R[a_0, a_1]] \text{ (Equation 2.2)}$$

where f is an aggregate function which returns a value between 0 and 1. An example of such a function is the max function, since the relatedness between two nodes in a hierarchy is always in the range [0, 1] as mentioned previously.

Now, a relatedness score between user u and v can be calculated by using Equation 2.1 and Equation 2.2 as follows:

$$P[f, u, v] = M[u, v] \cdot L[f, u, v] \text{ (Equation 2.3)}$$

This means that the product of the mutual followings that user u has with user v multiplied by the relatedness derived from an underlying hierarchical structure (and filtered with an aggregate function f) defines the relatedness between two users.

Given all of these equations, we can define a recommendation algorithm which returns the top K recommended users for user u to follow, where $K \geq 0$. This is the algorithm:

procedure TOPKRECOMMENDED(u, f, K) ▷ Get top K recommended users for user u
    $TopK \leftarrow \{\}$ ▷ list
    for $v \in F_u^2$ **do**
        $score \leftarrow P[f, u, v]$
        $TopK.addElement[(v, score)]$ ▷ Add (Key, Value) tuple to list
    end for
    $SortInDecreasingOrderByValue[TopK]$ ▷ Sort list on the 2nd element of each tuple
    return $TopK.subset[0, K]$ ▷ Return first 0 up to K elements
end procedure

(Note that this is a naive outline of the algorithm to demonstrate the behavior. A real implementation should use a max heap instead of performing a sort operation on a list for time and space improvements.)

This algorithm iterates through the user u's F_u^2 set and calculates and stores the scores (using Equation 2.3) for each u, v pair where $v \in F_u^2$ in a list. Then, the list is sorted in descending order, where the highest scores are first, and the first K elements are returned.