第六章、社交网络数据分析

讲师：武永亮

教学目标

- 了解获取社交网络数据的途径
- 了解社交网络数据简介
- 理解基于社交网络的推荐
- 理解给用户推荐好友

目录

利用社交网络数据

- 继搜索引擎之后的下一个互联网热潮就是社交网络：
  - Facebook和Twitter
  - 新浪微博和QQ
- 基于社交网络的推荐可以很好地模拟现实社会。在现实社会中，很多时候我们都是通过朋友获得推荐。美国著名调查机构调查了影响用户相信某个推荐的因素。调查结果显示，**90%的用户**相信朋友对他们的推荐，**70%的用户相信网上**其他用户对广告商品的**评论**。从该调查可以看到，好友的推荐对于增加用户对推荐结果的信任度非常重要。

获取社交网络数据的途径

- 社交网络数据的获取方式-电子邮件

获取社交网络数据的途径

- 社交网络数据的获取方式-用户注册信息

获取社交网络数据的途径

- 社交网络数据的获取方式-用户的位置数据
- 社交网络数据的获取方式-论坛和讨论组
- 社交网络数据的获取方式-即时聊天工具
- 社交网络数据的获取方式-社交网站
  - 以Facebook为代表的社交网络称为社交图谱（ social graph ）
  - 以Twitter为代表的社交网络称为兴趣图谱（ interest graph ）。

目录

社交网络数据简介

- 社交网络定义了用户之间的联系，因此可以用图定义社交网络。
- 用图$G(V,E,w)$定义一个社交网络，其中V是顶点集合，每个顶点代表一个用户， E是边集合，如果用户va和vb有社交网络关系，那么就有一条$e(v_a,v_b)$连接这两个用户，而 $w(v_a,v_b)$定义了边的权重。
- 业界有两种著名的社交网络：
  - 一种以Facebook为代表，它的朋友关系是需要双向确认的，因此在这种社交网络上可以用无向边连接有社交网络关系的用户。
  - 另一种以Twitter为代表，它的朋友关系是单向的，因此可以用有向边代表这种社交网络上的用户关系。

社交网络数据简介

- 此外，对图G中的用户顶点u，定义out(u)为顶点u指向的顶点集合，定义in(u)为指向顶点u的顶点集合。那么，在Facebook这种无向社交网络中显然有out(u)=in(u)。
- 有3种不同的社交网络数据。
  - 双向确认的社交网络数据
  - 单向关注的社交网络数据
  - 基于社区的社交网络数据

社交网络数据简介

- 社交网络数据也满足长尾分布
- Slashdot的社交网络数据集统计了用户入度和出度的分布。 Slashdot的社交网络是一个有向图结构。因此，用户的入度反映了用户的社会影响力。

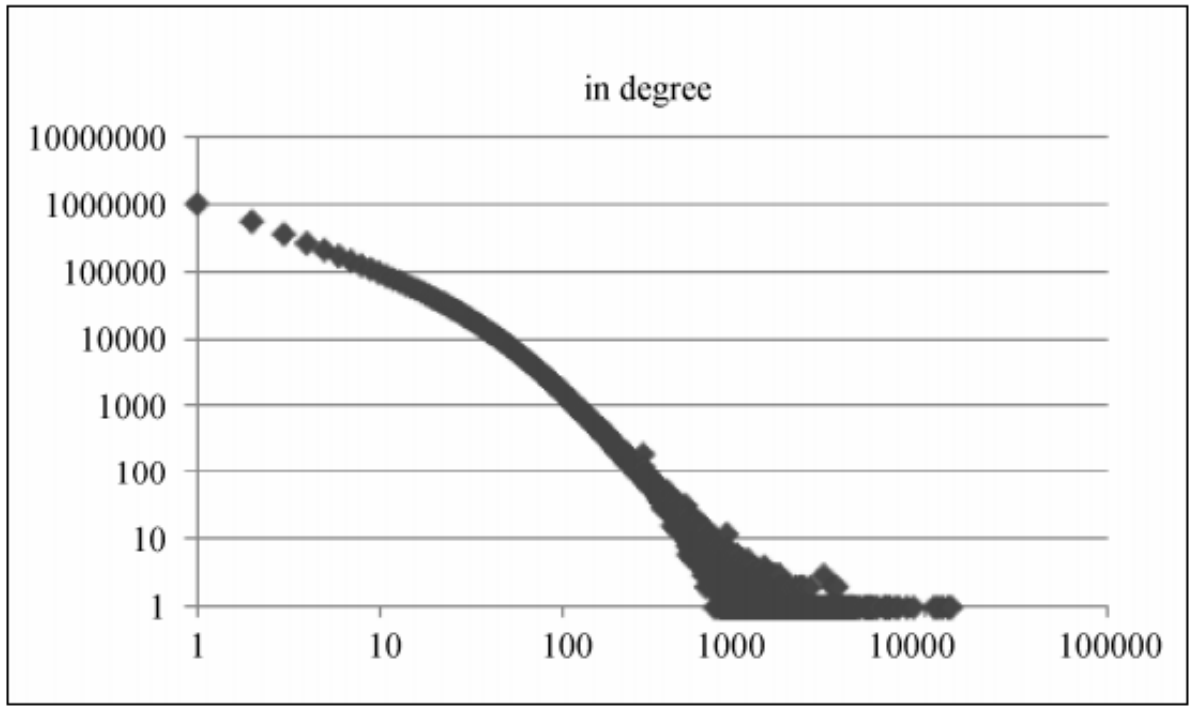


图6-3　社交网络（Slashdot）中用户入度的分布

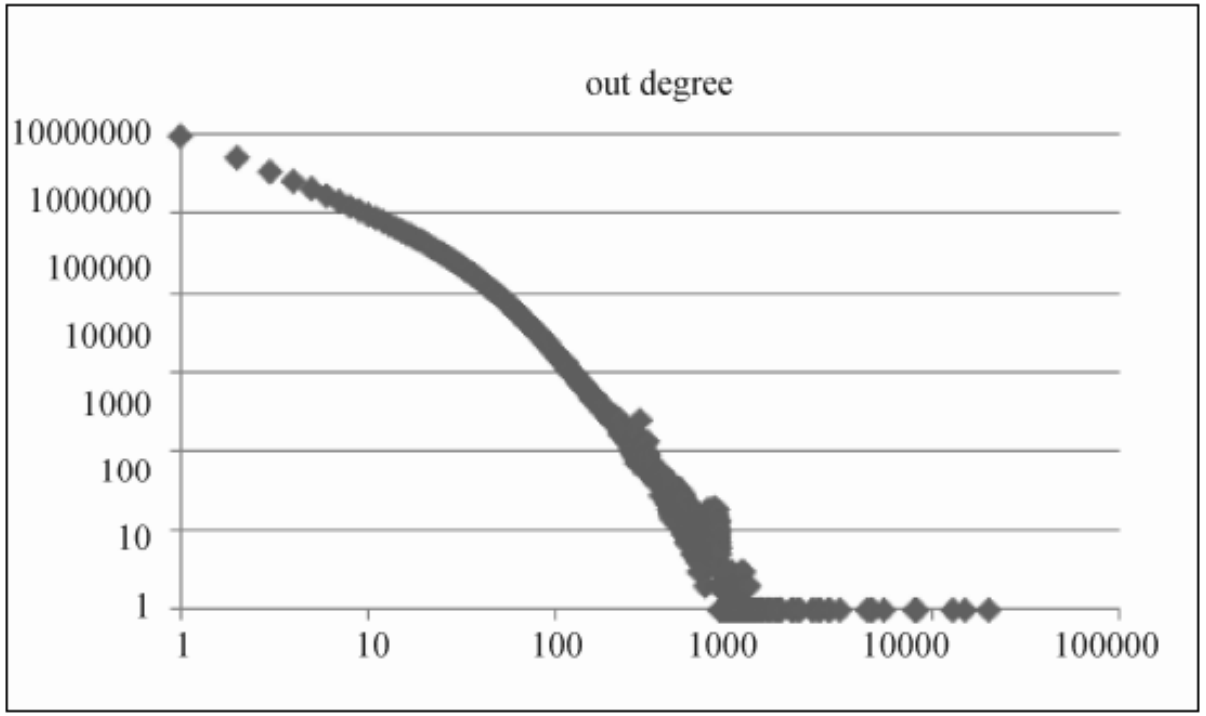


图6-4　社交网络（Slashdot）中用户出度的分布

目录

基于社交网络的推荐

- 很多网站都利用社交网络数据给用户提供社会化推荐。

基于社交网络的推荐

- 社会化推荐的优点：
  - 好友推荐可以增加推荐的信任度
  - 社交网络可以解决冷启动问题
- 缺点：很多时候并不一定能提高推荐算法的离线精度（准确率和召回率）。特别是在基于社交图谱数据的推荐系统中，因为用户的好友关系不是基于共同兴趣产生的，所以用户好友的兴趣往往和用户的兴趣并不一致。

基于社交网络的推荐-基于邻域的社会化推荐算法

- 如果给定一个社交网络和一份用户行为数据集。其中社交网络定义了用户之间的好友关系，而用户行为数据集定义了不同用户的历史行为和兴趣数据。那么我们想到的最简单算法是给用户推荐好友喜欢的物品集合。即用户u对物品i的兴趣p_{ui}可以通过如下公式计算：

$$p_{ui} = \sum_{v \in \text{out}(u)} r_{vi}$$

- 其中out(u)是用户u的好友集合，如果用户v喜欢物品i，则$r_{vi}=1$，否则$r_{vi}=0$。

基于社交网络的推荐-基于邻域的社会化推荐算法

- 即使都是用户u的好友，不同的好友和用户u的熟悉程度和兴趣相似度也是不同的。我们应该在推荐算法中考虑好友和用户的熟悉程度以及兴趣相似度：

$$p_{ui} = \sum_{v \in out(u)} w_{uv} r_{vi}$$

- w_{uv}由两部分相似度构成，一部分是用户u和用户v的熟悉程度，另一部分是用户u和用户v的兴趣相似度。

基于社交网络的推荐-基于邻域的社会化推荐算法

- 用户u和用户v的熟悉程度（ familiarity ）描述了用户u和用户v在现实社会中的熟悉程度。一般来说，用户更加相信自己熟悉的好友的推荐，因此我们需要考虑用户之间的熟悉度。熟悉度可以用用户之间的共同好友比例来度量。也就是说如果用户u和用户v很熟悉，那么一般来说他们应该有很多共同的好友。

$$\text{familiarity}(u,v)=\frac{|\text{out}(u)\bigcap\text{out}(v)|}{|\text{out}(u)\bigcup\text{out}(v)|}$$

基于社交网络的推荐-基于邻域的社会化推荐算法

- 除了熟悉程度，还需要考虑用户之间的兴趣相似度。我们都和父母很熟悉，但很多时候我们和父母的兴趣却不相似，因此也不会喜欢他们喜欢的物品。因此，在度量用户相似度时还需要考虑兴趣相似度（similarity），而兴趣相似度可以通过和UserCF类似的方法度量，即如果两个用户喜欢的物品集合重合度很高，两个用户的兴趣相似度很高。

$$\text{similiarity}(u,v)=\frac{|N(u)\bigcap N(v)|}{|N(u)\bigcup N(v)|}$$

- 其中$N(u)$是用户u喜欢的物品集合。

基于社交网络的推荐-基于邻域的社会化推荐算法

- 下面的代码实现社会化推荐的逻辑。

```
def Recommend(uid, familiarity, similarity, train):
    rank = dict()
    interacted_items = train[uid]
    for fid,fw in familiarity[uid]:
        for item,pw in train[fid]:
            # if user has already know the item
            # do not recommend it
            if item in interacted_items:
                continue
            addToDict(rank, item, fw * pw)
    for vid,sw in similarity[uid]:
        for item,pw in train[vid]:
            if item in interacted_items:
                continue
            addToDict(rank, item, sw * pw)
    return rank
```

基于社交网络的推荐-基于图的社会化推荐算法

- 用户的社交网络可以表示为社交网络图，用户对物品的行为可以表示为用户物品二分图，而这两种图可以结合成一个图。

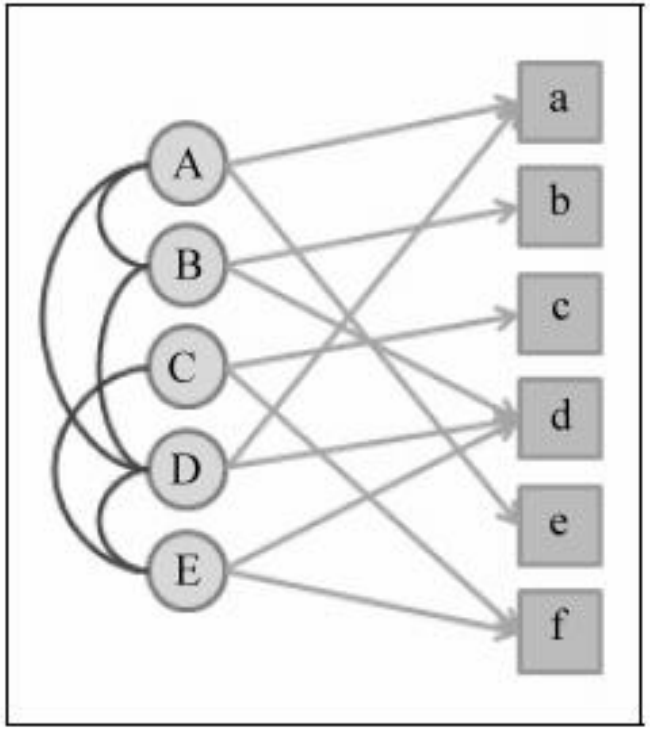


图6-7　社交网络图和用户物品二分图的结合

基于社交网络的推荐-基于图的社会化推荐算法

- 社交网络的社交网络关系:
  - 用户和用户之间直接的社交网络关系, friendship
  - 两个用户属于同一个社群, membership

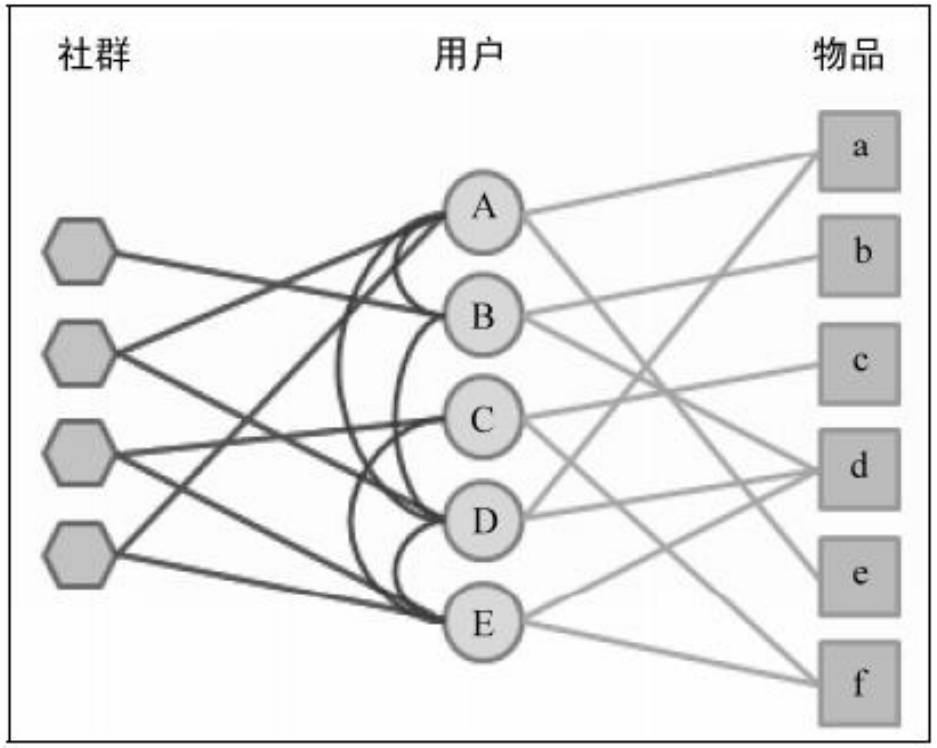


图6-8　融合两种社交网络信息的图模型

基于社交网络的推荐-实际系统中的社会化推荐算法

- 基于邻域的社会化推荐算法看起来简单，但在实际系统中却是很难操作的，这主要是因为该算法需要拿到用户所有好友的历史行为数据，而这一操作在实际系统中是比较重的操作。所以，如果一个算法再给一个用户做推荐时，需要他所有好友的历史行为数据，这在实际环境中是比较困难的。
- 从几个方面改进基于邻域的社会化推荐算法
  - 两处截断。第一处截断就是在拿用户好友集合时只拿出和用户相似度最高的N个好友。此外，在查询每个用户的历史行为时，可以只返回用户最近1个月的行为.
  - 重新设计数据库。给每个用户维护一个消息队列（message queue），当一个用户发表一条微博时，所有关注他的用户的消息队列中都会加入这条微博。这个实现的优点是用户获取信息墙时可以直接读消息队列，所以终端用户的读操作很快。

基于社交网络的推荐-实际系统中的社会化推荐算法

- 如果将Twitter的架构搬到社会化推荐系统中，我们就可以按照如下方式设计系统：
  - 首先，为每个用户维护一个消息队列，用于存储他的推荐列表；
  - 当一个用户喜欢一个物品时，就将（物品ID、用户ID和时间）这条记录写入关注该用户的推荐列表消息队列中；
  - 当用户访问推荐系统时，读出他的推荐列表消息队列，对于这个消息队列中的每个物品，重新计算该物品的权重。计算权重时需要考虑物品在队列中出现的次数，物品对应的用户和当前用户的熟悉程度、物品的时间戳。同时，计算出每个物品被哪些好友喜欢过，用这些好友作为物品的推荐解释。

社会化推荐系统和协同过滤推荐系统

- 对社会化推荐系统进行用户调查示例，评测3个真实的电影推荐系统（ Amazon、 MovieCritic、 Reel.co ）和3个真实的图书推荐系统（ Amazon.com、 RatingZone、 Sleeper ）。在评测每一个真实推荐系统时，参试者需要按照如下顺序完成实验 ：
  - 利用一个虚假的邮箱注册推荐系统，从而保证这些新账号没有任何历史行为。
  - 利用新注册的账号给电影、图书进行评分。
  - 查看推荐列表。
  - 如果一开始的推荐列表中没有符合用户兴趣的物品，用户将被要求从该网站中搜索到至少一个令他们感兴趣的物品，如果实在找不到，可以停止搜索。
  - 回答调查问卷。

社会化推荐系统和协同过滤推荐系统

- 在完成真实推荐系统的评测任务后，参试者还需要评测社会化推荐系统。实验步骤如下。
  - 每个参试者需要提供3个他们认为了解其兴趣的好友的邮箱。
  - 实验人员给每个好友发邮件，要求他们给参试者推荐3本书和3部电影，并且要求这些书和电影不能是参试者之前和他们讨论过的，即不能是参试者之前告诉他们并表示喜欢的物品。
  - 参试者将会看到好友推荐的书和电影的缩略图以及一段简单的介绍。
  - 回答调查问卷。
- 社会化推荐系统推荐结果的用户满意度明显高于主要基于协同过滤算法的几个真实推荐系统。

基于社交网络的推荐-信息流推荐

- 信息流推荐是社会化推荐领域的新兴话题，它主要针对Twitter和Facebook这两种社交网站。

基于社交网络的推荐-信息流推荐

- 目前最流行的信息流推荐算法是Facebook的EdgeRank，该算法综合考虑了信息流中每个会话的时间、长度与用户兴趣的相似度。 Facebook将其他用户对当前用户信息流中的会话产生过行为的行为称为edge，而一条会话的权重定义为：

$$\sum_{\text{edges } e} u_e w_e d_e$$

  - u_e指产生行为的用户和当前用户的相似度，这里的相似度主要是在社交网络图中的熟悉度；
  - w_e 指行为的权重，这里的行为包括创建、评论、 like（喜欢）、打标签等，不同的行为有不同的权重。
  - d_e指时间衰减参数，越早的行为对权重的影响越低。

基于社交网络的推荐-信息流推荐

- EdgeRank算法的个性化因素仅仅是好友的熟悉度，它并没有考虑帖子内容和用户兴趣的相似度。为此，GroupLens的研究人员Jilin Chen深入研究了信息流推荐中社会兴趣和个性化兴趣之间的关系。他们的排名算法考虑了如下因素:
  - 会话的长度
  - 话题相关性
  - 用户熟悉程度

基于社交网络的推荐-信息流推荐

- Jilin Chen同样也设计了一个用户调查。首先，他通过问卷将用户分成两种类型。第一种类型的用户使用Twitter的目的是寻找信息，也就是说他们将Twitter看做一种信息源和新闻媒体。而第二种用户使用Twitter的目的是了解好友的最新动态以及和好朋友聊天。然后，他让参试者对如下5种算法的推荐结果给出1~5分的评分，其中1分表示不喜欢，5分表示最喜欢。
  - Random　给用户随机推荐会话。
  - Length　给用户推荐比较长的会话。
  - Topic　给用户推荐和他兴趣相关的会话。
  - Tie　给用户推荐和他熟悉的好友参与的会话。
  - Topic+Tie　综合考虑会话和用户的兴趣相关度以及用户好友参与会话的程度。

基于社交网络的推荐-信息流推荐

- 通过收集用户反馈，Jilin Chen发现（如图6-11所示），对于所有用户不同算法的平均得分是：
  - Topic+Tie > Tie > Topic > Length > Random
- 而对于主要目的是寻找信息的用户，不同算法的得分是：
  - Topic+Tie ≥ Topic > Length > Tie > Random
- 对于主要目的是交友的用户，不同算法的得分是：
  - Topic+Tie > Tie > Topic > Length > Random

基于社交网络的推荐-信息流推荐

- 综合考虑用户的社会兴趣和个人兴趣对于提高用户满意度是有帮助的。

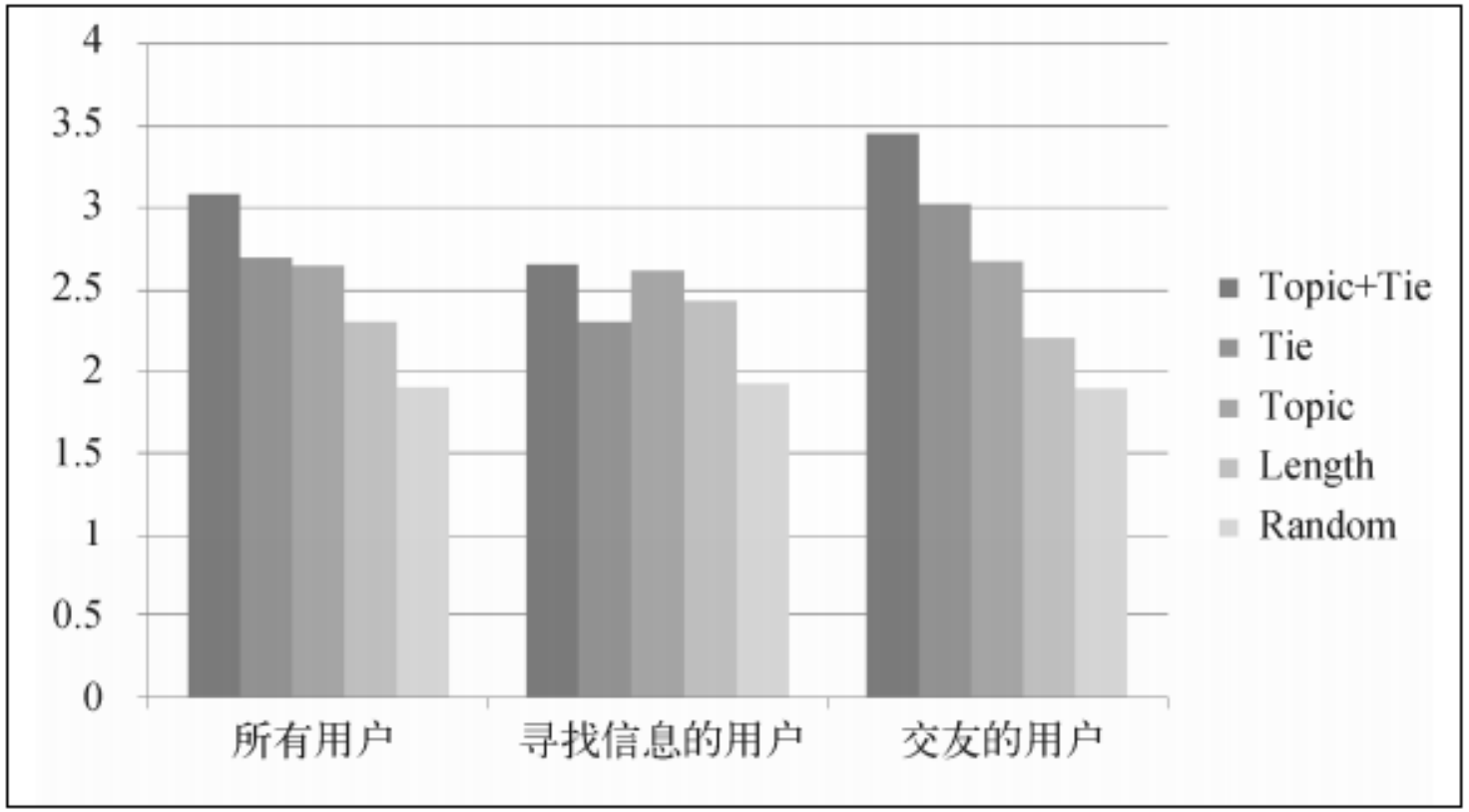
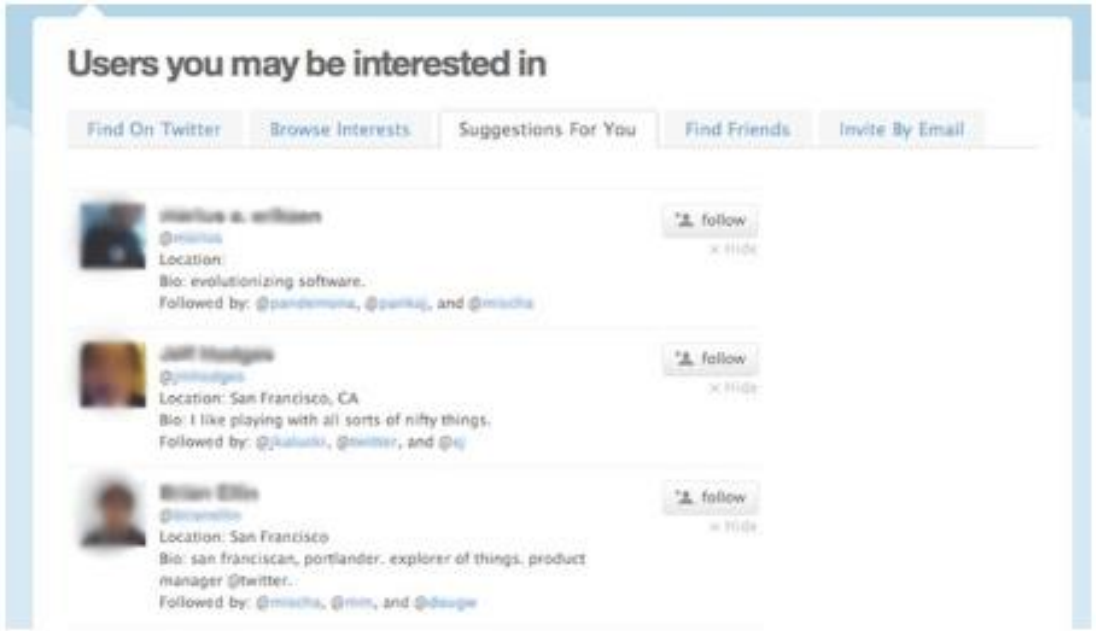


图6-11　Jilin Chen的用户调查实验结果①

目录

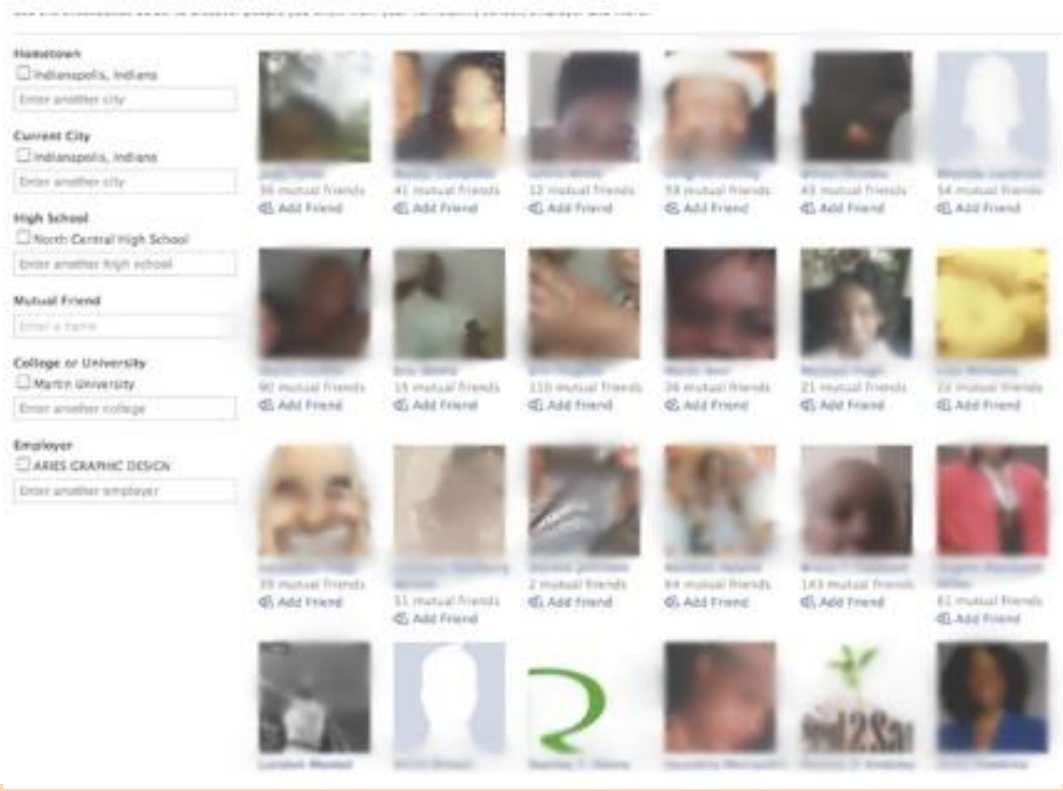
给用户推荐好友

- 好友关系是社会化网站的重要组成部分，如果用户的好友很稀少，就不能体验到社会化的好处。因此好友推荐是社会化网站的重要应用之一。好友推荐系统的目的是根据用户现有的好友、用户的行为记录给用户推荐新的好友，从而增加整个社交网络的稠密程度和社交网站用户的活跃度。



给用户推荐好友

- 好友关系是社会化网站的重要组成部分，如果用户的好友很稀少，就不能体验到社会化的好处。因此好友推荐是社会化网站的重要应用之一。好友推荐系统的目的是根据用户现有的好友、用户的行为记录给用户推荐新的好友，从而增加整个社交网络的稠密程度和社交网站用户的活跃度。



给用户推荐好友

- 我们可以给用户推荐和他们有相似内容属性的用户作为好友，常用的内容属性。
  - 用户人口统计学属性，包括年龄、性别、职业、毕业学校和工作单位等。
  - 用户的兴趣，包括用户喜欢的物品和发布过的言论等。
  - 用户的位置信息，包括用户的住址、IP地址和邮编等。

给用户推荐好友

- 在Twitter和微博为代表的以兴趣图谱为主的社交网络中，用户往往不关心对于一个人是否在现实社会中认识，而只关心是否和他们有共同的兴趣爱好。因此，在这种网站中需要给用户推荐和他有共同兴趣的其他用户作为好友。
- 我们在第3章介绍基于用户的协同过滤算法（UserCF）时已经详细介绍了如何计算用户之间的兴趣相似度，其主要思想就是如果用户喜欢相同的物品，则说明他们具有相似的兴趣。在新浪微博中，可以将微博看做物品，如果两个用户曾经评论或者转发同样的微博，说明他们具有相似的兴趣。在Facebook中，因为有大量用户Like（喜欢）的数据，所以更容易用UserCF算法计算用户的兴趣相似度。

给用户推荐好友

- 在社交网站中，我们会获得用户之间现有的社交网络图，然后可以基于现有的社交网络给用户推荐新的好友，最简单的好友推荐算法是给用户推荐好友的好友。对于用户u和用户v，我们可以用共同好友比例计算他们的相似度：

```
def FriendSuggestion(user, G, GT):
    suggestions = dict()
    friends = G[user]
    for fid in G[user]:
        for ffid in GT[fid]:
            if ffid in friends:
                continue
            if ffid not in suggestions:
                suggestions[ffid] = 0
            suggestions[ffid] += 1
    suggestions = {x: y / math.sqrt(len(G[user]) * len(G[x]) for x,y in suggestions}
```

$$w_{out}(u,v) = \frac{|out(u) \cap out(v)|}{\sqrt{|out(u)||out(v)|}}$$

给用户推荐好友

- 我们也可以定义in(u)是在社交网络图中指向用户u的用户的集合。在无向社交网络图中，out(u)和in(u)是相同的集合。但在微博这种有向社交网络中，这两个集合就不同了，因此也可以通过in(u)定义另一种相似度：

```
def FriendSuggestion(user, G, GT):
    suggestions = dict()
    for fid in GT[user]:
        for ffid in G[fid]:
            if ffid in friends:
                continue
            if ffid not in suggestions:
                suggestions[ffid] = 0
            suggestions[ffid] += 1
    suggestions = {x: y / math.sqrt(len(GT[user]) * len(GT[x]) for x,y in suggestions}
```

$$w_{\text{in}}(u,v) = \frac{|\text{in}(u) \bigcap \text{in}(v)|}{\sqrt{|\text{in}(u)||\text{in}(v)|}}$$

给用户推荐好友

- 我们还可以定义第三种有向的相似度：

$$w_{out,in}(u,v)=\frac{|out(u)\cap in(v)|}{|out(u)|}$$

- 这个相似度的含义是用户u关注的用户中，有多大比例也关注了用户v。但是，这个相似度有一个缺点，就是在该相似度的定义下所有人都和名人有很大的相似度。这是因为这个相似度在分母的部分没有考虑$|in(v)|$的大小。因此，我们可以用如下公式改进上面的相似度：

$$w'_{out,in}(u,v)=\frac{|out(u)\cap in(v)|}{\sqrt{|out(u)||in(v)|}}$$

```
def FriendSuggestion(user, G, GT):
    suggestions = dict()
    for fid in GT[user]:
        for ffid in G[fid]:
            if ffid in friends:
                continue
            if ffid not in suggestions:
                suggestions[ffid] = 0
            suggestions[ffid] += 1
    suggestions = {x: y / math.sqrt(len(GT[user]) * len(GT[x]) for x,y in suggestions}
```

给用户推荐好友

- 通过离线实验评测提出的相似度，评测哪种相似度能更好地预测用户之间的好友关系我们使用该集合中提供的Slashdot社交网络数据集。该数据集是一个有向图，包含82 168个顶点和948 464条边。

表6-1　3种不同好友推荐算法的召回率和准确率

	Slashdot		Epinion	
	召回率	准确率	召回率	准确率
w_{out}	14.09%	3.63%	7.40%	1.87%
w_{in}	12.32%	3.17%	7.20%	1.82%
$w_{out.in}$	8.62%	2.22%	11.94%	3.02%
$w'_{out.in}$	9.12%	2.35%	8.77%	2.21%

给用户推荐好友

- 通过用户调查对比了不同算法的用户满意度，其中算法如下：
  - InterestBased　给用户推荐和他兴趣相似的其他用户作为好友。
  - SocialBased　基于社交网络给用户推荐他好友的好友作为好友。
  - Interest+Social　将InterestBased算法推荐的好友和SocialBased算法推荐的好友按照一定权重融合。
  - SONA　SONA是IBM内部的推荐算法，该算法利用大量用户信息建立了IBM员工之间的社交网络。这些信息包括所在的部门、共同发表的文章、共同写的Wiki、IBM的内部社交网络信息、共同合作的专利等。
- 从结果可以发现，对推荐结果的新颖性不同算法的排名如下：
  - InterestBased > Interest+Social > SocialBased > SONA

给用户推荐好友

表6-2　不同好友推荐算法的问卷调查结果[②]

	认　识		不　认　识	
	好	不　好	好	不　好
InterestBased	19.5%	3.0%	30.1%	41.5%
SocialBased	55.4%	5.2%	23.8%	15.5%
Interest+Social	31.8%	4.4%	24.9%	38.9%
SONA	75.9%	10.0%	6.6%	7.6%

内容回顾

1. 获取社交网络数据的途径
2. 社交网络数据简介
3. 基于社交网络的推荐
4. 给用户推荐好友

Thank you !