Christos’ template - v14: META PAPER: Spotting Fake Reviews in App Stores

first name SCS CMU first@cs.cmu.edu
second name SCS CMU second@andrew.cmu.edu
Christos Faloutsos SCS CMU christos@cs.cmu.edu

March 16, 2023

Abstract

How can you find strange nodes in a who-calls-whom graph? Spotting anomalies in graphs is an important topic. Our META PAPER method has the following properties (a) Scalability, being linear on the input size (b) Effectiveness, spotting 90% of the anomalies in real data (c) Parameter-free, requiring no user-defined parameters. Experiments on 3GB of real data from epinions.com illustrate the benefits of our method.

1 Introduction

Given a large count of reviews for products, how can one spot the fake ones. On line reviews are important. They are often faked, for monetary gain. How to spot the truth?

Here we propose META PAPER, a method to spot fake reviews. The main idea behind our method is a principled way to merge several warning signals. Figure 1 shows the results of our method where META PAPER outperforms the competition by up to 999%.

The advantages of our method are

- Scalability: it scales linearly with the input size
- Effectiveness: it gives very good reconstruction error, on real data
- Parameter-free: it requires no user-defined parameters.

Reproducibility: we publish our data and our code, at www.cs.cmu.edu

Figure 1: META PAPER wins: Execution time for META PAPER, on epinions.com

The outline of the paper is typical: we give the survey (Section 2), the proposed method (Section 3), experiments (Section 4), and conclusions (Section 5).

2 Background and Related Work

There is a lot of work on app reviews, and we group it in the following sub-sections.

2.1 Fraud detection

Bla-bla-bla - fake citation: [3] [2]

2.2 Anomaly detection

bla-bla - oddball, subdue etc [1]
However none of the above methods fullfils all the specs of our method: (a) scalability (b) effectiveness. Table 1 contrasts METAPAPER against the state of the art competitors.

<table>
<thead>
<tr>
<th>Property</th>
<th>Method1</th>
<th>Method2</th>
<th>METAPAPER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalability</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Effectiveness</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Parameter-free</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>other-stuff</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: **METAPAPER matches all specs**, while competitors miss one or more of the features.

### 3 Proposed Method

In this section we present the proposed method, we analyze it and provide the reader with several interesting -at least in our opinion- observations. Table 2 gives the list of symbols we use.

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G$</td>
<td>a graph</td>
</tr>
<tr>
<td>$A$</td>
<td>adjacency matrix</td>
</tr>
</tbody>
</table>

Table 2: Symbols and Definitions

### 3.1 Intuition

The main idea behind our METAPAPER is to exploit network effects: if we see a bi-partite core, then we suspect fraud.

Figure 2 and 3 illustrates the intuition

![Figure 2: Sample tikz figure](image1)

![Figure 3: Another Sample tikz figure](image2)

### 3.2 Algorithm

Algorithm 1 shows the pseudo code and Algorithm 2 shows a variation

1. **Data:** a graph $G$
2. **Result:** the communities in $G$
3. **Initialization:**
4. **Put graph $G$ on stack:**
5. **While stack is not empty do**
   1. **Process current graph:**
   2. **If has $n^2$ edges then**
      1. **Add to output:**
   3. **Else**
      1. **Split in two:**
         1. **Put both on stack:**
   4. **End**
   5. **Pop stack:**

Algorithm 1: FakeCom: Community detection algorithm

### 3.3 Complexity Analysis

**Theorem 1** METAPAPER requires time linear on the input size

**Proof 1** from Eq *bla*, with lagrange multipliers

### 3.4 SQL implementation

In fact, we can use SQL to implement our algorithm:
Algorithm 2: How to write algorithms

\begin{verbatim}
select name, ssn
from student
where name = 'smith'
and grade = 'A';
\end{verbatim}

4 Experiments

Here we report experiments to answer the following questions
Q1. **Scalability**: How fast is our META\textsc{Paper}?
Q2. **Effectiveness**: How well does META\textsc{Paper} work on real data?

The graphs we used in our experiments are described in the table 3.

4.1 Q1 - Scalability

In figure 4, we present the experimental results for the real-world datasets we used.

4.2 Q2 - Effectiveness

In Figure bla, we show the precision/recall of META\textsc{Paper} for the epinions.com dataset. Notice that

4.3 Discussion - practitioner’s guide

Given the above, we recommend that practitioners choose for the parameter values of META\textsc{Paper}, and for their datasets.

4.4 Discoveries - META\textsc{Paper} at work

Thanks to our method, we processed real data and noticed the following observations

**Observation 1** META\textsc{Paper} works better than expected, on real data.

The reason is that we assumed uniformity, while real data have skewed distributions (Pareto-like), and thus favor our approach.

**Observation 2** META\textsc{Paper} works faster than expected.

bla-bla

5 Conclusions

We presented META\textsc{Paper}, which addresses the fake-review problem, using network effects. The main idea is to spot anomalous graph substructures, using belief propagation.

The advantages of the method are

1. **Scalability**: it scales linearly with the input size, as shown in Figure 4 and Lemma
2. **Effectiveness**: it gives excellent precision, on real-world data
3. **Parameter-free**: it requires no user intervention - META\textsc{Paper} sets all parameters to reasonable defaults, and it is insensitive to the exact choices anyway

We also presented experiments on 3GB of real data, where META\textsc{Paper} outperformed the competitors.
<table>
<thead>
<tr>
<th>Nodes</th>
<th>Edges</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>13,579</td>
<td>37,448</td>
<td>AS Oregon</td>
</tr>
<tr>
<td>23,389</td>
<td>47,448</td>
<td>CAIDA AS 2004 to 2008</td>
</tr>
</tbody>
</table>

Table 3: Summary of real-world networks used.

by 10 percentage points of accuracy, and 3x faster execution time.

**Reproducibility:** We have already open-sourced our code, at [www.cs.cmu.edu](http://www.cs.cmu.edu)

**Acknowledgements:** We would like to thank Christos Faloutsos for his *MetaPaper* list of suggestions on the presentation.

**References**

