MONSTOR: An \textbf{Inductive} Approach for Estimating and Maximizing Influence over \textbf{Unseen} Networks

Jihoon Ko  Kyuhan Lee  Kijung Shin  Noseong Park
Social relationship can be represented as a graph!
Social relationship can be represented as a graph!
Information cascade in social relationship
Information cascade in social relationship
Influence Maximization

• Find a certain number of seed nodes to maximize the spread of information through a social network

• Two major issues
  1) How to model the information cascade
  2) How to solve the problem based on information cascade model
How to model the information cascade

• Independent Cascade (IC)
  • When node $v$ becomes active, it has a single chance of activating each currently inactive neighbor $w$
  • The activation attempt succeeds with some probability $p_{vw}$

• Linear Threshold (LT)
  • Each node $v$ has threshold $p_v$, and it is activated by its neighbors when at least $p_v$ fraction of its neighbors are active
How to solve the problem based on IC model

• **Greedy** approach [KKT03]
  • Greedily choose nodes using Monte-Carlo simulations repeatedly
  • Guarantees the approximation ratio of \(1 - \frac{1}{e}\)

• \(d\) (usually set to 10,000) simulations takes \(O(d|\mathcal{E}|)\) time!
  => performance bottleneck 😞

• CELF [LKGFGV07], UBLF [ZZGZG13] still rely heavily on MC simulations!
  • CELF significantly reduced MC simulations using submodular property
  • UBLF derived upper bound of marginal gain for every node at initialization step
How to solve the problem based on IC model

- **Greedy** approach [KKT03]
  - Greedily choose nodes using Monte-Carlo simulations repeatedly
  - Guarantees the approximation ratio of \( 1 - \frac{1}{e} \)

- \( d \) (usually set to 10,000) simulations takes \( O(d|\mathcal{E}|) \) time!

  \( \Rightarrow \) performance bottleneck 😞

- CELF [LKGFVG07], UBLF [ZZGZG13] still rely heavily on MC simulations!

**Solution**: Estimate results from repeated MC simulations in fast!
Outline

• Preliminaries
• Proposed method: MONSTOR
• Experimental Results
• Summary
Preliminaries

• Activation Probability from u to v
  • The success probability that the node u activates its neighbor v when u is infected

• Bernoulli Trial (BT):
  \[ p_{uv} = \frac{|\text{actions}(u,*) \cap \text{actions}(v,*)|}{|\text{actions}(u,*)|} \]

• Jaccard Index (JI):
  \[ p_{uv} = \frac{|\text{actions}(u,*) \cap \text{actions}(*,v)|}{|\text{actions}(u,*) \cup \text{actions}(*,v)|} \]

• Linear Probability (LP):
  \[ p_{uv} = \frac{|\text{actions}(u,*) \cap \text{actions}(*,v)|}{|\text{actions}(*,v)|} \]

Notation

actions(x,*) : the set of actions done by node x
actions(*,x) : the set of actions whose object is node x
Preliminaries

• Activation probability matrix
  • The adjacency matrix when weighting each directional edge \((u, v)\) by \(p(u, v)\)

• Infection Probability for node \(v\) given a seed set \(S\)
  • The probability that \(v\) is infected under the IC model with \(S\)
  • Defined as \(\rho(v)\)

• Infection probability vector
  • \(\pi := [\rho(v)] \in [0,1]^{|V|}\) be the vector of \(\rho(v)\)
  • \(\pi_i\) be the infection probability vector during the first \(i\) steps
Our problems

• **Influence Estimation (IE)**
  • Given a seed set $S$,
  • Estimate its influence $\sum_{v \in V} \rho(v)$ (the expected number of infected nodes)

• **Influence Maximization (IM)**
  • Given the number of seed nodes $k$,
  • Find the set $S$ of $k$ seed nodes
  • to Maximize the influence $\sum_{v \in V} \rho(v)$
Outline

• Preliminaries
• Proposed method: MONSTOR
• Experimental Results
• Summary
Proposed method: MONSTOR

• Neural network-based method for estimating MC simulation results under IC model

• MONSTOR can estimate MC simulation results in social networks unseen during training

• Significantly speeds up existing IM methods by replacing simulations
Overall Workflow

- Training data
  - Social Network #1
  - Social Network #2
  - Social Network #3

- GCN-based model with \( l \) layers
  - Train

- Stack to estimate multi-hop simulations

- MONSTOR
  - \( \pi_0 \)
  - \( \pi_1 \)
  - \( \pi_2 \)
  - \( \pi_3 \)

- Training Phase
- Testing Phase
Overall Workflow

1) Collect one or more social networks \( \{G_1, G_2, \ldots \} \)

|      | \(|V|\) | \(|\mathcal{E}|\) | \(\Sigma p_{(u,v)}/|\mathcal{E}|\) in BT | \(\Sigma p_{(u,v)}/|\mathcal{E}|\) in JL | \(\Sigma p_{(u,v)}/|\mathcal{E}|\) in LP |
|------|-------|-------|---------------------------------|------------------------|------------------------|
|      | Train | Test  | Train  | Test                    | Train  | Test                    |
| Extended | 11,409 | 58,972 | 0.0797 | 0.0919                   | 0.0335 | 0.0410                   | 0.1614 | 0.1837                   |
| WannaCry | 35,627 | 169,419 | 0.0726 | 0.0947                   | 0.0298 | 0.0449                   | 0.1979 | 0.1630                   |
| Celebrity | 15,184 | 56,638 | 0.0321 | 0.0279                   | 0.0016 | 0.0016                   | 0.2614 | 0.256                    |
Overall Workflow

2) From each $G_j$, collect the tuples $\{(\pi_i, \pi_{i-1}, \cdots, \pi_{i-e}, P_j): i \geq e\}$, after choosing a seed set $S$ randomly

- $e > 1$ is a hyperparameter
- $P_j \in \{BT, JI, LP\}$
- Repeat multiple times with different seed sets

Notation

$\pi_i$ be the infection probability vector during the first $i$ steps
Overall Workflow

3) Train GCN-based model $M$ with $l$ layers, estimating $\pi_i$ given $\pi_{i-1}, \ldots, \pi_{i-e}$
• Estimates a single step of the IC model
Overall Workflow

4) Stack $s$ times the pre-trained model
   • The stacked model estimates $\pi_s$ from $\pi_0$
   • Estimates end-to-end simulations

5-1) For IE problem, compute $\langle 1, \pi_s \rangle$
5-2) For IM problem, replace the MC simulation subroutine of existing IM algorithms with MONSTOR
Detailed Design

• Final output of our model is determined to

\[ M(\pi_{i-1}, ..., \pi_{i-e}, P; \theta) := \min\{\pi_{i-1} + h^l, u_i\} \]

, where \( u_i := \pi_{i-1} + (\pi_{i-1} - \pi_{i-2})P \) (theoretical upper bound)
Detailed Design

- Training/Validation: online postings (and their cascade logs) during the first 50% of time (1,600 training / 400 validation tuples)
- Test: those during the remaining 50% of time (2,000 testing tuples)
Outline

• Preliminaries
• Proposed method: MONSTOR
• Experimental Results
• Summary
Experiments

• Q1. Accuracy in Influence Maximization

• Q2. Accuracy in Influence Estimation

• Q3. Scalability

• Q4. Submodularity
Competitors

- **Simulation-based algorithms**: Greedy with MC simulations
  - UBLF for BT, JI
  - CELF for LP

- **Non-simulation-based algorithms**
  - SSA / D-SSA [NTD16]
  - PMIA [CWW10] / IRIE [JHC12]

- **Our proposed approach**
  - For BT, JI: **U-MON** (UBLF with MONSTOR)
  - For LP: **C-MON** (CELF with MONSTOR)
**Experimental Settings**

- For training: two out of three networks
- For testing: Choose each of the three networks

**Inductive** setting: testing with the graph unseen during training
- Ex) U/C-MON (E+W)
Q1. Influence Maximization (IM)

• **Question:** How **accurate** are simulation-based IM algorithms equipped with MONSTOR, compared to competitors?

• **Answer:** Test with BT/JI - **U-MON** was most accurate in most cases
Q1. Influence Maximization (IM)

• Question: How accurate are simulation-based IM algorithms equipped with MONSTOR, compared to competitors?

• Answer: Test with LP - **C-MON** was most accurate in most cases

|                | Extended |             |             |             |             |             |             |             |             |
|----------------|----------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
|                | $k=10$   | 50         | 100         | 10          | 50          | 100         | 10          | 50          | 100         |
| Target Influence | 1852.4   | 2876.5     | 3264.9      | 5271.6      | 7880.0      | 9098.3      | 5508.4      | 5616.7      | 5657.7      |
| C-MON (E+W)    | 1843.0   | **2863.0** | **3253.8**  | 5246.4      | **7862.1**  | **9073.5**  | 5508.9      | 5615.0      | 5664.9      |
| C-MON (E+C)    | 1840.6   | 2848.5     | 3236.5      | **5253.0**  | 7844.5      | 9041.7      | 5508.8      | 5616.4      | 5666.4      |
| C-MON (W+C)    | 1839.5   | 2853.1     | **3242.0**  | 5248.6      | 7850.7      | 9045.7      | 5508.8      | 5615.0      | 5665.0      |
| D-SSA          | 1844.3   | 2858.7     | 3236.1      | 5256.7      | 7783.4      | 8977.3      | 5509.0      | 5606.2      | 5633.8      |
| SSA            | 1843.8   | 2858.6     | 3236.1      | **5257.2**  | 7783.6      | 8977.0      | 5508.8      | 5606.3      | 5633.9      |
| IRIE           | 1816.2   | 2829.8     | 3201.2      | 5109.1      | 7714.1      | 8840.1      | **5509.1**  | **5617.4**  | **5667.4**  |
| PMIA           | 1830.0   | 2828.9     | 3243.2      | 5196.7      | 7807.6      | 8981.8      | 5508.5      | 5604.2      | 5630.2      |
Q2. Influence Estimation (IE)

- The ground-truth influences of test seed sets and the estimated influences were highly correlated.

<table>
<thead>
<tr>
<th>Model</th>
<th>Target graph</th>
<th>BT</th>
<th>JI</th>
<th>LP</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-MON (E+W) or C-MON (E+W)</td>
<td>Celebrity (Unseen)</td>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
</tr>
<tr>
<td>U-MON (E+C) or C-MON (E+C)</td>
<td>WannaCry (Unseen)</td>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
</tr>
<tr>
<td>U-MON (W+C) or C-MON (W+C)</td>
<td>Extended (Unseen)</td>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
<td><img src="image" alt="Graph" /></td>
</tr>
</tbody>
</table>
Q3. Scalability

- **Question:** How rapidly does the estimation time grow as the size of the input graph increase?
- **Answer:** The runtime per stacked GCN was near-linear in the number of edges in the input graph.

| $|\mathcal{E}|$ | $2^{20}$ | $2^{21}$ | $2^{22}$ | $2^{23}$ | $2^{24}$ | $2^{25}$ | $2^{26}$ |
|----------------|----------|----------|----------|----------|----------|----------|----------|
| Estimation time (sec) | 11.5 | 17.7 | 31.0 | 56.3 | 108.9 | 411.0 | 819.7 |
Q4. Submodularity

• Question: Is MONSTOR submodular as the ground-truth influence function is?
• Using each pair $S$ and $T$ of the seed sets, we tested whether
  \[ f(S) + f(T) \geq f(S \cup T) + f(S \cap T) \]
  is met or not
• Answer: Influence estimation by MONSTOR can be considered as submodular in practice

<table>
<thead>
<tr>
<th></th>
<th>Extended</th>
<th>WannaCry</th>
<th>Celebrity</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-MON (E+W)</td>
<td>0.9993</td>
<td>0.9997</td>
<td>0.9938</td>
</tr>
<tr>
<td>C-MON (E+C)</td>
<td>0.9994</td>
<td>0.9997</td>
<td>0.9970</td>
</tr>
<tr>
<td>C-MON (W+C)</td>
<td>0.9992</td>
<td>0.9996</td>
<td>0.9945</td>
</tr>
</tbody>
</table>

The ratio of the cases where the submodularity holds (Tested with LP)
Outline

• Preliminaries
• Proposed method: MONSTOR
• Experimental Results
• Summary
Summary

we present **MONSTOR**, an inductive learning algorithm for estimating the influence of seed nodes under the IC model.

- **Accurate** in IM/IE Tasks

  Most accurate in 17/27 Cases

- **Scalable & Submodular**

The code and datasets used in the paper are available at [https://github.com/jihoonko/asonam20-monstor/](https://github.com/jihoonko/asonam20-monstor/)
MONSTOR: An Inductive Approach for Estimating and Maximizing Influence over Unseen Networks

Jihoon Ko  Kyuhan Lee  Kijung Shin  Noseong Park