Report for Project IMP

11610511 Shiyi Li

December 2018

1 Preliminaries

Influence maximization is the problem of finding a small subset of nodes in a network that can maximize the spread of influence. The stochastic diffusion models include independent cascade (IC) model, linear threshold (LT) model, general threshold model and others. The IM is NP-hard and the major ways to solve IMP is greedy algorithm and heuristic algorithm[1].

2 Methodology

The basic definition of each variables will be introduced in the methodology.

2.1 Representation

Several variables should be defined in advance in the algorithm.

- my_graph: a graph that store all the nodes, edges and weight of network.
  - nodes: all the nodes of graph
  - in_edges: the income edges of the node
  - out_edges: the outcome edges of the node
  - weight: weight of each edges
- pre_defined: parameters given in the command line.
  - network_file: address of the network file
  - model: the concrete model that should be used in the algorithm
  - seed_size: size of the output seed set
  - seed_file: address of the seed file
  - time_limit: maximum time

2.2 architecture

For ISE, construct a graph from the given network file at first. Then, read the seed file to get the seed set. Last, calculate the average influence according to the given model.

- Class Graph:
  - add_edge: record the edge and its weight
- IC: calculate the influence of the seed set according to model "IC"
- LT: calculate the influence of the seed set according to model "LT"
- read_network: read the network file and construct the graph
- read_seed: read the seed file and get the seed set

For IMP, construct a graph from the given network file at first. Then, determine which algorithm should be used according to the number of nodes in the network and the seed size. If the seed size is small, using the greedy algorithm CELF. If the seed size is larger, using the heuristic algorithm to get the waiting seed set, then, get the final seed set by greedy algorithm. For those extremely large seed size, using the heuristic algorithm.

- Class Graph:
  - add_edge: record the edge and its weight

- IC: calculate the influence of the seed set according to model ”IC”
- LT: calculate the influence of the seed set according to model ”LT”
- read_network: read the network file and construct the graph
- CELF: greedy algorithm to find the seed set
- degree_discount: heuristic algorithm to find the seed set

2.3 Algorithm

Here shows the exact algorithm used in the ISE and IMP, which includes IC, LT, CELF and DegreeDiscount.

Algorithm 1 ISE for IC

```
Input: graph, SeedSet
Output: count
 1: initialize ActivitySet as SeedSet
 2: count = ActivitySet.length
 3: while !ActivitySet.IsEmpty do
 4:    initialize newActivitySet as an empty set
 5:    for each seed in ActivitySet do
 6:       for each inactive neighbor in seed do
 7:          try to activate neighbor with random probability
 8:          if the neighbor is activated then
 9:             update the state of neighbor
10:             newActivitySet.add(neighbor)
11:          end if
12:       end for
13:    end for
14:    count = count + newActivitySet.length
15:    ActivitySet = new ActivitySet
16: end while
```

ISE for IC: find the influence of seed set by model ”IC”.

Algorithm 2 ISE for LT

```
Input: graph, SeedSet
Output: count
 1: initialize ActivitySet as SeedSet
 2: initialize the threshold randomly for each seed
 3: if the threshold == 0.0 then
 4:    join the seed into ActivitySet
 5: end if
 6: count = ActivitySet.length
 7: while !ActivitySet.IsEmpty do
 8:    initialize newActivitySet as an empty set
 9:    for each seed in ActivitySet do
10:       for each inactive neighbor in seed do
11:          calculate the weight of all activated neighbors of the seed w_total
12:          if w_total ≥ neighbor.threshold then
13:             update the state of neighbor
14:             newActivitySet.add(neighbor)
15:          end if
16:       end for
17:    end for
18:    count = count + newActivitySet.length
19:    ActivitySet = new ActivitySet
20: end while
```

ISE for IC: find the influence of seed set by model ”LT”.

Algorithm 3 DegreeDiscount

```
Input: graph, seed_size
Output: seed_set
 1: initialize $S = 0$
 2: for each vertex $v$ do
 3:    compute its degree $d_v$
 4:    $dd_v = d_v$
 5:    initialize $t_v$ to 0
 6: end for
 7: for $i = 1$ to $k$ do
 8:    select $u = argmax_v\{dd_v | v \in V\S\}$
 9:    $S = S \cup \{u\}$
10:    for each neighbor $v$ of $u$ and $u \in V\S$ do
11:       $t_v = t_v + 1$
12:       $dd_v = d_v - 2t_v - (d_v - t_v)t_v p$
13:    end for
14: end for
```

DegreeDiscount: heuristic algorithm to find the seed set [1].

Algorithm 4 CELF

```
Input: graph, seed_size
Output: seed_set
 1: $S \leftarrow \emptyset; Q \leftarrow \emptyset; last\_seed = null; cur\_best = null$
 2: for each $u \in V$ do
 3:     $u.mg1 = \sigma(\{u\})$; $u.prev\_best = cur\_best$; $u.mg2 = \sigma(\{u, cur\_best\})$; $u.flag = 0$
 4:     add $u$ to $Q$. Update $cur\_best$ based on $mg1$
 5: end for
 6: while $|S| < k$ do
 7:     $u$ = top element in $Q$
 8:     if $u.flag == |S|$ then
 9:         $S \leftarrow S \cup \{u\}$; $Q \leftarrow Q - \{u\}$; $last\_seed = u$
10:     else if $u.prev\_best == last\_seed$ then
11:         $u.ng1 = u.mg2$
12:     else
13:         $u.mg1 = \Delta_u(S)$; $u.prev\_best = cur\_best$; $u.mg2 = \Delta_u(S \cup \{cur\_best\})$
14:     end if
15:     $u.flag = |S|$
16:     update $cur\_best$
17: end while
```

CELF: algorithm to find the seed set[2].

3 Empirical Verification

3.1 Dataset

Dataset includes: network and NetHEPT, seeds and seeds2

3.2 Performance measure

The version of Python is 3.7.0. For ISE, performance is measured by the running time. For IMP, performance is measured by the running time and maximum influence.

3.3 Hyperparameters

The number of waiting seed set of DegreeDiscount is depend on seed size. If seed size is less than 10, the number is 100. If seed size is less than 30, the number is two times of seed size. If not, DegreeDiscount will give the final seed set directly. The number is equal to seed size.

Table 1: Performance of ISE

Instances	Running Time	Influence
network-seeds-IC	0.847	5.03
network-seeds-LT	0.89	5.04
network-seeds2-IC	0.86	30.43
network-seeds2-LT	3.72	37.08
NetHEPT-5seeds-IC	27.81	276.07
NetHEPT-5seeds-LT	57.83	337.51

Table 2: Performance of IMP

Instances	Running Time	Influence
network-5-IC	1.42	30.51
network-s-LT	1.10	36.93
NetHEPT-5-IC	4.07	270.90
NetHEPT-5-LT	104.71	336.10

3.4 Experimental results

Experimental result is showed in Table 1 and Table 2.

3.5 Conclusion

For those have small seed size, CELF can reach a great influence spread. However, it still takes a few hours to complete in a graph with a few tens of thousands of vertices. Degree discount can give the result of a graph in a few seconds, but the result has a large bias to the optimal influence spread. A simple combination of greedy algorithm and heuristic algorithm can only balance the efficiency and result, but not give an acceptable result in a certain time.

References

[1] Wei Chen, Yajun Wang, and Siyu Yang. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD inter-*

national conference on Knowledge discovery and data mining, pages 199–208. ACM, 2009.

[2] Amit Goyal, Wei Lu, and Laks VS Lakshmanan. Celf++: optimizing the greedy algorithm for influence maximization in social networks. In *Proceedings of the 20th international conference companion on World wide web*, pages 47–48. ACM, 2011.