A Game Theoretic Approach for Fake News Minimization

Project Mentor: Ayush Nagal

Details about the Simulation

We conducted the simulations using Python. For the simulations we used the `snap.py` package available from the Stanford Network Analysis Project. To generate the graphs, we first generate a directed Erdös-Renyi graph. To add further functionality to represent the properties of social networks, we convert it into a `networkx` package network. Now we initialize the gullibility of the all vertices `g(u)` as uniform random values in $(0, 1)$. The dilemma indices `d(u)` are also initialized as 1. We initialize the state of nodes as `UNINFORMED` and type as `COMMON`, and initialize extra nodes with no edges to act as oracle nodes. Then we call two functions `deploy_bots()` and `deploy_activators()` with an additional real argument `exponent` for each of the two agents that will distribute the selection probability (for activators and monitored nodes for bots) among the nodes based on one of their properties (we experimented by taking this as the out-degree, closeness and betweenness of the nodes). The larger this value, the more probable is the selection of the vertices with greater value of the given property and vice versa.

Now we simulate the game as described hereon. We perform a multi-source BFS on the network starting from all the activator nodes. On every BFS visit, we check if the node is `DOUBTFUL`, `UNINFORMED` or `IMMUNE`, we will continue ahead without any processing. Otherwise it starts spreading the misinformation to its out-edges. If there is any bot in its followers, then it tries to validate the information. We convert the level upto which BFS has occured till now into a probability for reporting misinformation (the deeper we are, the more probable it must be to report the fake news, we use an increasing function to map this). If reporting occurs, then the spread from the current node is stopped and all followers are notified of the report. Consequently, all the involved nodes (spreader and followers excluding activators) will go into `DOUBTFUL` state and their `d(u)` gets incremented by one. Contrarily, if no bot reports, then each follower initially in `DOUBTFUL` or `UNINFORMED` state becomes `MISINFORMED` with a probability that is a function of its gullibility `g(u)`, weight/influence of the edge, and its dilemma `d(u)`. The spread continues with all the followers that become misinformed continuing the BFS. We continue performing the BFS starting with all nodes that can spread the information, for a number of iterations, which ensures that we finally have a nearly stable distribution (the variance in the fractions of vertices in different state is nearly constant).

Now we perform such simulations for multiple random graphs, and by varying various parameters such as the number of activators and bots, the exponents of both the players, various strategies that the bots and activators follow (whether they select small number of dense nodes or large number of sparse nodes), selection criteria in the exponent and bot reports, and also for the size of the total graph. Since the set of strategies is extremely large for both players, we restrict ourselves to only some strategies that the players select probabilistically. We find the utilities of both the players for all these variations. The constants used for the utilities in the expressions given in model description are taken as $\alpha_P = \alpha_T = 1, \beta_P = \beta_T = 100/N$, where N is the number of nodes in the network. The cost of deploying the bots is taken as the total number of monitored nodes, and the cost of buying activator nodes is the sum of the out-degrees of all activators. We thus plot the trends of the utilities of the platform on above variations, and derive some optimal results relevant to the decisions taken by the players. We also repeat the simulations by taking a graph generated from a real social network Facebook from the Stanford Large Network Dataset Collections.