Influence Maximization via Representation Learning

George Panagopoulos[1], Fragkiskos Malliaros[2], Michalis Vazirgiannis[1]

[1]LIX, École Polytechnique, France, [2]CentraleSupélec and Inria Saclay

Data Science and Mining Team
École Polytechnique

Introduction to the Problem

Typical Influence Maximization:

- Relies on diffusion simulation models.
- Influence probabilities are set at random.
- Faces scalability issues.
- Estimated influence spread can wildly diverge from realistic spread [1].

Influence Learning Models:

- Assume influence independence, overlooking the effects of network assortativity [2].
- Ignore higher order influence between nodes.
- Are based on propagation network which has high computational cost [3].

Contribution:

1. Analysis of influencer activity for efficient node-context creation.
2. Multi-task neural network architecture to compute influencer embeddings.
3. An algorithm to perform influence maximization using the learnt representations.

I. Influence Analysis on Sina Weibo

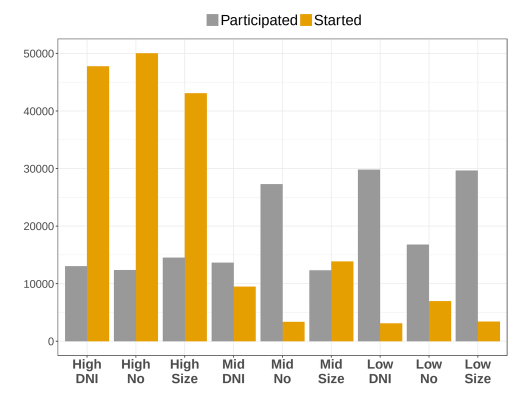
Influencers create or copy more?

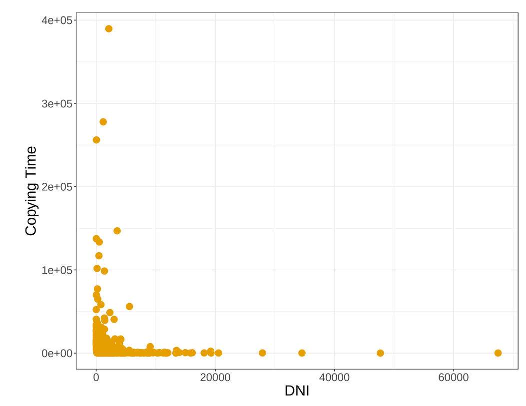
- Rank initiators in the test set based on success metrics.
- Successful influencers are *more prone to start than participate* in train cascades → Derive only the context of the cascade initiator.

Does copying time play a role?

- Average copying time in train cascades for every level of DNI in the test set.
- Influencers *tend to initiate fast cascades*. → Sample context based on the inverse of copying time.

*11 months of retweet cascades as "train", 1 month as "test".





II. Learning INFluencer vECTORs (INFECTOR)

Embed at the same hidden layer a node's:

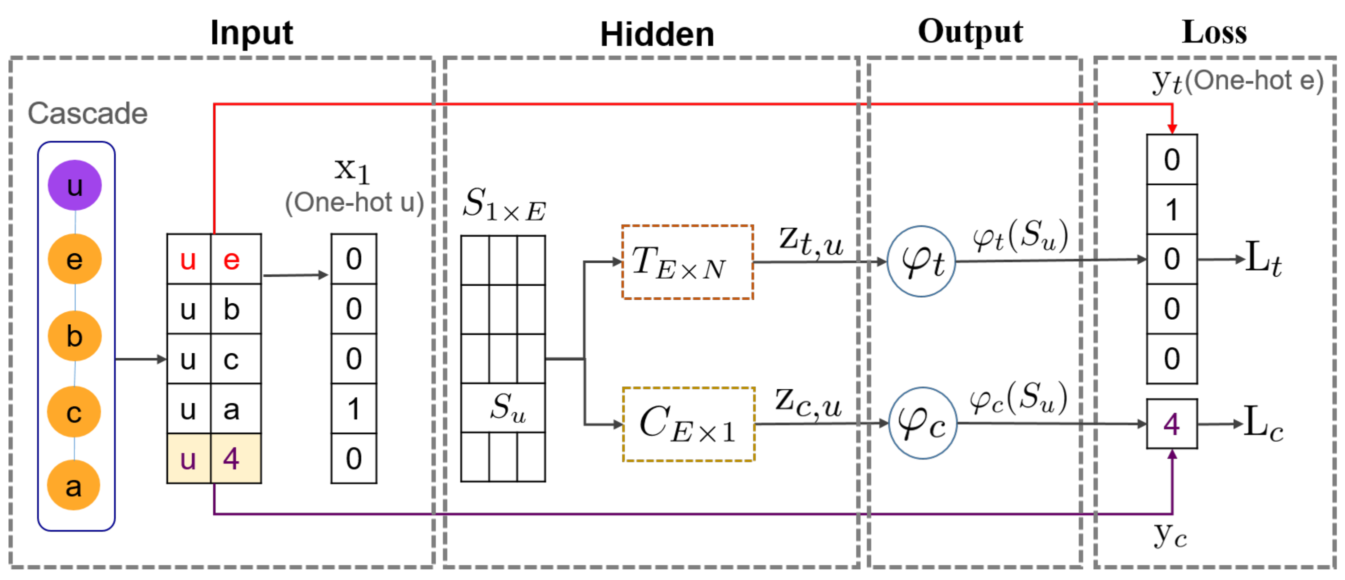
- likelihood to influence another node.
- aptitude to create lengthy cascades.

S is updated alternately by both inputs.

- S and T form the diffusion probabilities between two nodes.
- $|S|$ captures the nodes' cascade size.

	Classify Influenced Node	*Regress Cascade Size*
Hidden	$z_{t,u} = S_u T + b_t$	$z_{c,u} = S_u C + b_c$
Output	$\varphi_t(S_u) = \frac{e^{(z_{t,u})}}{\sum_{u' \in G} e^{z_{t,u'}}}$	$\varphi_c(S_u) = \frac{1}{1+e^{-(z_{c,u})}}$
Loss	$L_t = y_1 \log(\varphi_t(S_u))$	$L_c = (y_c - \varphi_c(S_u))^2$

Table: The layers of INFECTOR



III. Influence Maximization with INFECTOR

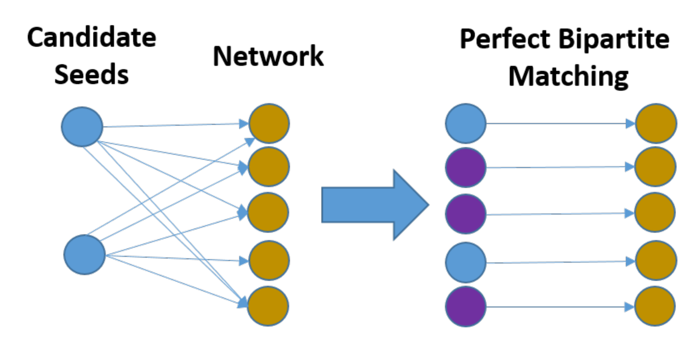
1. Predict diffusion probabilities:

$$D = \begin{bmatrix} \varphi_t\Big((ST)_{[1,1:N]}\Big) \\ \vdots \\ \varphi_t\Big((ST)_{[I,1:N]}\Big) \end{bmatrix},$$

2. Reduce the number of candidate seeds by keeping the top P% based on their expected influence spread.

$$\Lambda_u = \left\lceil N \frac{||S_u||_2}{\sum_{u' \in \mathcal{I}} ||S_{u'}||_2} \right\rceil,$$

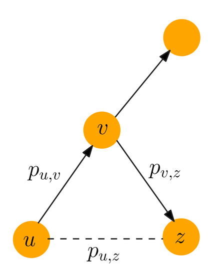
3. IM by weighted bipartite matching.



4. Influence spread is the seed's edge weight because there are no higher order paths.

$$\sigma'(s) = \sum_{j}^{\Lambda_s} \hat{D}_{s,j},$$

5. Optimize $\sigma'(S)$ in greedy manner and remove the node added in each iteration[1]. $\sigma'(S)$ is submodular → retains theoretical guarantees.
* Diffusion probabilities capture higher order correlations that other IM techniques fail to.



IV. Results

Methods

- Top nodes based on k-cores decomposition.
- Top nodes based on the average size of their train cascades.
- IMM: SOTA classic influence maximization algorithm [3].
- IMINFECTOR with 5 epochs, 0.1 learning rate, P=40 for Digg and 10 for the rest [4].

Evaluation

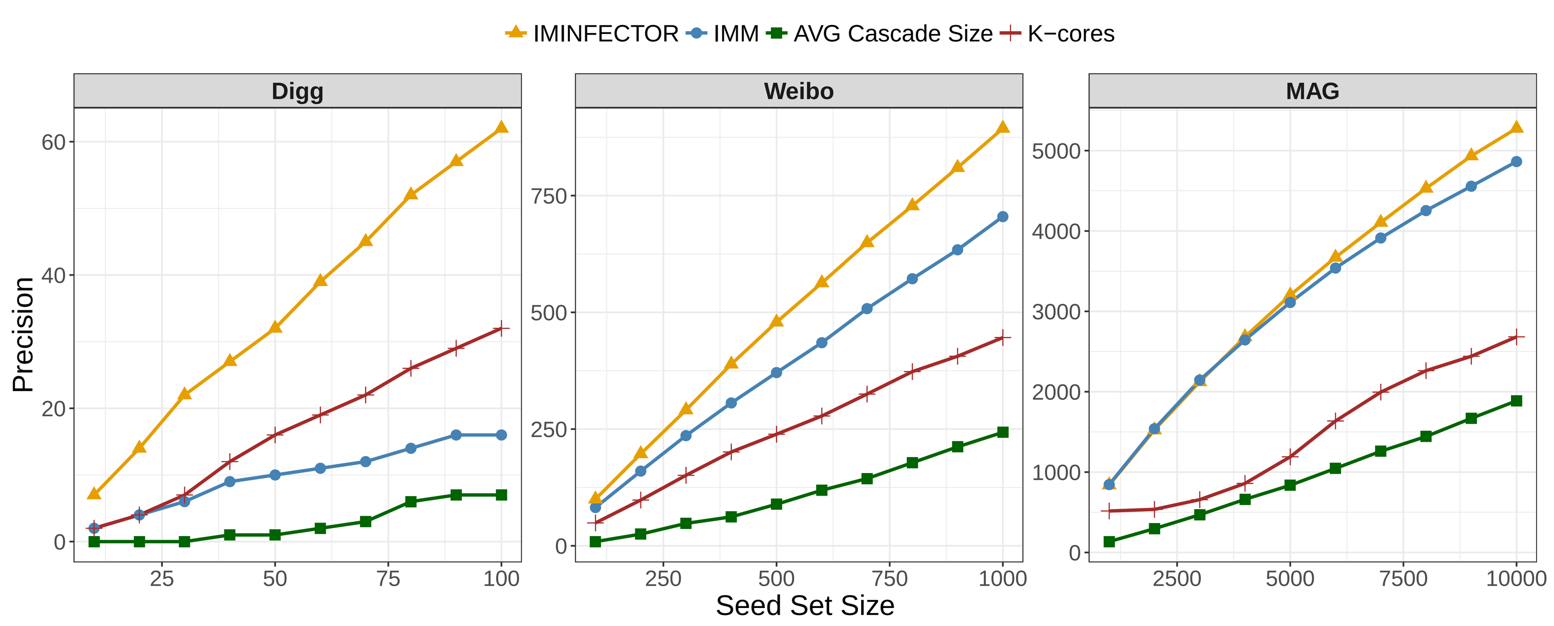
- Precision: How many of the predicted seeds initiate test cascades.
- DNI: Total number of nodes influenced by the predicted seeds in the test cascades.

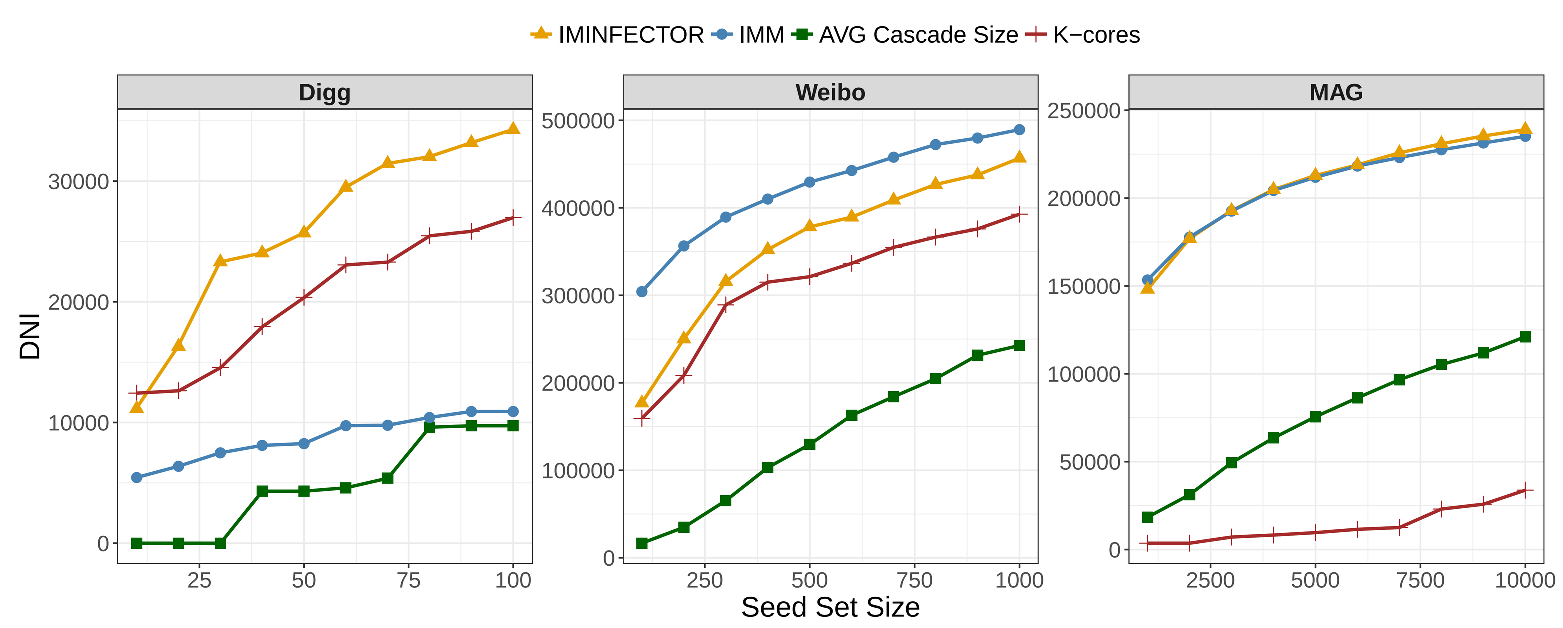
Data

- Static Directed Network + Diffusion cascades.
- 80-20% train and test split on cascades based on time precedence.

	Sina Weibo	*Digg*	*MAG*
Nodes	1,170,689	279,631	1,436,158
Edges	225,877,808	2,251,166	20,456,480
Cascades	115,686	3,553	181,020
Avg Cascade size	847	148	29

Table: Summary of the datasets used.





IV. Conclusion & Future Work

Conclusion

- Diffusion cascades should be taken into account for realistic influence maximization.
- Representation learning can be used to bridge influence learning and maximization.

Future Work

- Use complementary representations derived from Graph Neural Networks.
- Combine learnt representations from diffusion logs with online influence maximization.
- Compare with diffusion-based techniques and more metrics from network science.

Code and Instructions: `https://github.com/GiorgosPanagopoulos/IMINFECTOR`

IV. References

[1] Aral, Sinan, and Paramveer S. Dhillon. "Social influence maximization under empirical influence models." Nature human behaviour 2.6 (2018): 375.
[2] Aral, Sinan, and Dylan Walker. "Identifying influential and susceptible members of social networks." Science 337.6092 (2012): 337-341.
[3] Feng, Shanshan, et al. "Inf2vec: Latent Representation Model for Social Influence Embedding." ICDE, 2018.
[4] Tang, Youze, Yanchen Shi, and Xiaokui Xiao. "Influence maximization in near-linear time: A martingale approach."SIGMOD, 2015.
[5] Panagopoulos, George, Michalis Vazirgiannis, and Fragkiskos D. Malliaros. "Influence Maximization via Representation Learning." arXiv preprint arXiv:1904.08804 (2019)

george.panagopoulos@lix.polytechnique.fr