Content-based Network Influence Probabilities: Extraction and Application

Alvis Logins
Aarhus University

Panagiotis Karras
Aarhus University

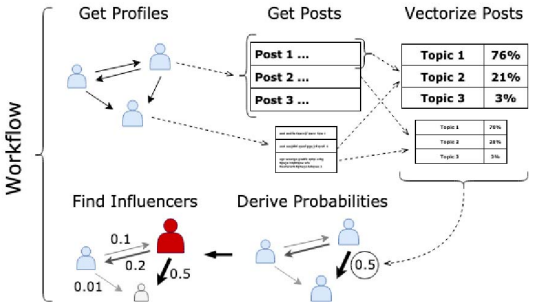
***Abstract*—The diffusion of information and influence in networks shapes ideas, habits, and behaviors. Several *diffusion control* methods have been proposed to harness, maximize, limit, or direct such diffusion processes in desirable ways. State-of-the-art algorithms for prescriptive fine-grained diffusion control rely on simple models, most prominently the Independent Cascade (IC) model, rather than on advanced machine learning approaches. The simplicity of such models can be an advantage. Yet, to exploit this advantage, one needs not only well-designed algorithms, but also a powerful model-training framework that yields well-informed models. Unfortunately, much research effort has been devoted to algorithm design, while the development of techniques for informing the underlying model has been largely neglected.**

We propose a new content analysis workflow that derives realistic IC model parameters for diffusion control algorithms. We rely on a log of user text messages to derive a measure of similarity among those messages, and therefrom calculate the probability that one node influences another. We evaluate our model in terms of its predictive power and apply it to two representative diffusion control problems under the IC model. Our results showcase the capacity of our methods to make correct predictions, and provide the first, to our knowledge, study of diffusion control problems with a real-world probability model.

I. Introduction

Network diffusion. Phenomena of diffusion in networks involve the spread of information, attitudes, or infections. Some of those phenomena are captured by the *Independent Cascade* (IC) model [1], which is has been applied in problems such as Influence Maximization (IM) [2], [3] and Node Immunization (NI) [4]. The IM problem asks to find a set of k nodes in a network that maximizes the expected spread of a diffusion starting out from these nodes. On the other hand, the NI problem asks for a set of nodes that, if removed, would minimize the expected spread a diffusion emanating from some other source. These problems arise in social network analysis, viral marketing, and epidemiology [5], [6].

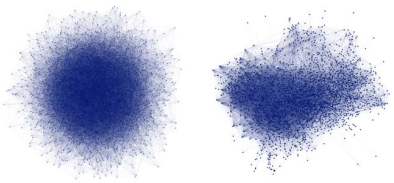
IC parameters. The IC model requires a single probability value that describes the influence between two nodes that are connected in the network. Seed nodes are initially *active*. Once any node is activated, it tries to activate (influence) its neighbors, and succeeds with the corresponding edge probability. The assessment and exploitation of diffusion control methods requires an accurate estimation of IC model probability parameters, as these probabilities may emphasize network localities. For example, as Zhang et al. [7] show, in real-world networks, the retweeting probability is negatively correlated to the amount of social circles a user belongs to, i.e. the number of friends that do not know each other. Such effects are hardly captured by synthetic models.



IC training Despite the wide usage of the IC model, the question of its *training* has been poorly covered. Some works train the IC model based on a set of *actions*. For instance, Saito et al. [8] learn model parameters using an expectation-maximization approach, applied on a set of references to a particular topic in a blogging platform. Netrapalli et al. [9] use such approach to learn graph parameters and the graph itself. Goyal et al. [10] suggest a more scalable approach to learn model parameters; they define a propagation probability between two adjacent nodes as $p_{uv} = \frac{A_{v2u}}{A_{\text{total}}}$, where A_{v2u} is the number of alike actions performed by v and then by u, and A_{total} is the total number of actions performed either by v (A_v), or by both users ($A_{u|v}$); they apply this solution on a dataset of Flickr accounts, where actions are group subscriptions.

Content-aware IM Other works [11], [12] introduce content-aware variants of the IM problem. Most pertinently, Barbieri et al. [11] apply content analysis to the topic-aware IC model, a generalization of the IC model where the propagated item (e.g., a piece of news) is associated with a *topic vector*. Likewise, each edge is associated with a *vector* of influence probabilities, calculated in terms of actions of its adjacent nodes. The activation probability over an edge depends on its *topic closeness* to thr propagated item, defined in terms of the item's topic vector and common interests of adjacent nodes. Still, the topic-aware IC model assumes topics are the *ultimate cause* of influence across network nodes. This model cannot infer influence due to the *relationship* among nodes. For example, a user subscribed to a news source she trusts will tend to be influenced by that source, regardless of topic. In this case, content analysis is useful for *inferring* influence (i.e., the user posts messages of content similar to that of posts by the trusted source), yet the inferred influence is

2375-9259/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDMW.2019.00020



(a) BA Graph (b) VK Graph

Fig. 1: Synthetic (left) and Real-World (right) Dataset

due to topic-independent factors. To our knowledge, there has been no attempt to train the IC model based on such general inferences from the *content* of exchanged messages, where items belonging to the same cascade may not be identical.

Frequent flow paths. Subbian et al. [13] study the problem of mining flow paths in a graph having frequency above a threshold f; thus, their focus is not to extract probabilities of flow across each edge, but to identify sequences of nodes across which information flows frequently.

Influence prediction Some works use richer models than the basic IC to deliver better *prediction* of influence. Cheng et al. [14] use message and user features to predict the size and shape of cascades by machine learning. Liu et al. [15] use Gibbs sampling to train joint probabilities of influence between users as well as between items, modeled as distinct types of nodes in a heterogeneous network. Wang et al. [16] use an embedding model and apply Maximum Likelihood Estimation for prediction, using the network structure for regularization. However, these prediction models are inapplicable with modern *diffusion control* algorithms, as they do not scale to the computations such algorithms rely upon, and violate those algorithms' assumption of edge-based influence independence.

In this paper, we propose an end-to-end framework, that so far has been missing, for the extraction of accurate, general-purpose, topic-independent influence probability values from the real-world content of user text messages. We start out with a topic vector representation of messages, similarly to previous works, yet consider circumstances in which, in a propagation trace, the content of a reposted message has no reference information and is not identical to, but merely *similar to*, the original. We emphasize that we do *not* propose some new model with better prediction power than others; instead, we propose a new learning framework for the *existing*, widely used IC model, so as to enhance the real-world applicability of IC-based diffusion control algorithms, and conduct the first, to our knowledge, experimental study of such algorithms under model parameters trained by real-world interactions.

II. Framework

The Independent Cascade (IC) model captures a diffusion in a network. Let $G = (E, V)$ be a directed graph, where nodes correspond to users of a social network and edges correspond to subscriptions. A node can be in active or passive state. Each edge is associated with a probability $p : E \to (0, 1)$, which indicates how likely it is that the source influences the target. Information propagates in discrete time stamps t_i. Nodes that are active at t_0 are called seeds. If a node v becomes active at t_i, then for each $(v, u) \in E$ such that u is not yet active, u becomes active at t_{i+1} with probability $p((v, u))$. We train the IC model by assessing similarities on a log of posts, using tokenization and morphological analysis, as follows.

A. Data Collection

We use the VKontakte (VK)[1] social network. A *profile* in VK represents either a single user, or a community page (a *group*). The VK API allows to query only public data of active profiles. A profile can subscribe to another profile. A mutual subscription is called a *friendship*. We model profiles as nodes in a social graph. Each profile has a *wall* — a blog with posts. A post may contain a text message, pictures, audio, video, or documents. We use text message contents to derive post similarity, and ignore all attachments. A post is labeled by publishing time, author, content, and a history of reposts. A *repost* is a post that refers to another post; a post can refer only to one other post; the referred post appears on the wall of the repost author, along with content by the author. We do not concatenate the texts of all reposts, yet we define all reposts to be similar to original posts by default. Yet, as only 0.2% of all downloaded posts are reposts, we can not use reposts for influence probability learning. Therefore, we calculate message similarity based on content. We query the following information about profiles:

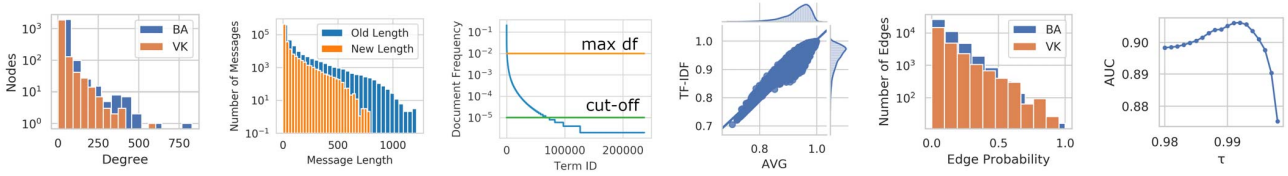
- Information about a profile, incl. state (active or inactive).
- The list of subscribers of a profile.
- The latest 100 posts from the wall of a profile, including a history of reposts; reposts may belong to any network profile, including closed or deleted ones; this limit is due to VK API's constraint on a query to a profile's wall.

We query groups and users independently. We initiate the database with a few random nodes, and continue to query nodes on first-come-first-served basis, whereby new node IDs come from the retrieved lists of friends/subscribers. For those nodes with known lists of friends, we queried the latest posts.

After retrieving several thousands of nodes, we change the approach so as to enhance graph connectivity. First, we query a detailed information about a set of profiles. We then query lists of friends and posts for publicly available profiles with more than 10 connections; we filter those with less than 10 connections to reduce data sparsity. Then we go through a process in which we iteratively collect profile ids by a priority queue, increasing the score of a node v by 1 each time v appears as a friend of a user in the previous set, and by 2 if v has authored any repost from the previous set.

Eventually, we extract a list of nodes, each having a full friend list and wall available and appearing in at least another node's friend list, amounting to $5.6 \cdot 10^5$ nodes and $1.5 \cdot 10^8$ edges; 14% of those are user profiles, while others are group profiles; groups are more likely to set public profiles. We select nodes having at least 5 posts and pick the largest weakly connected component to obtain a graph of 2452 nodes, 28108 edges, and $106,217$ posts; the graph, depicted on Figure 1b,

[1] https://vk.com/



(a) Degree Distr. (b) Message Length (c) Term Frequency (d) Message Sim. (e) Probability Distr. (f) AUC

Fig. 2: Dataset statistics

has clustering coefficient [17] 0.121 and mean degree 22; its degree distribution appears in Figure 2a.

B. Probability Extraction

We used as tokens morphological word lemmas obtained by MyStem[2] [18]; we ignore unrecognized words, e.g., URLs and smileys, and omit stopwords with the NLTK library [19]. Figure 2b shows message length distribution before and after preprocessing; in VK there is no character limit.

To calculate the similarity of posts, we need a term similarity measure. We considered two such measures: (i) the Jensen-Shannon divergence of term co-frequency, suggested in [20]; and (ii) a skip-gram model [21] pretrained on Wikipedia, provided by FastText[3]. FastText utilizes word stem information to represent words that are not in the training data. Figure 3 shows the correlation among these two measures, based on pairwise cosine similarities, by on two measures, among a random sample of terms. We see that there is no much correlation. Table I shows examples of some of the most similar (translated) terms according to FastText. Based on the observation of such examples, we opted for FastText.

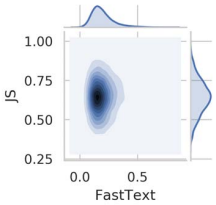


Fig. 3: Term sim.

Term 1	Term 2	Similarity
mess up	curl up	0.7704
pour	knead	0.7066
economic	sociological	0.6649
button	shirt	0.6308
fanatic	aggressive	0.6113

TABLE I: FastText sim. examples

Next, we consider two ways to calculate post similarity: (i) *AVG* [22], the cosine similarity among the averages of term vectors in each post; (ii) *enriched TF-IDF* [20], which is tailored for embeddings of short messages, as it amplifies the entry for a term w in the tf-idf vector of a message m using all terms w' present in m, weighted by similarity to w:

$$\text{tf-idf}_{w,m} = 1 - \Pi_{w'}(1 - \text{tf}_{w',m} \cdot \text{idf}_{w'} \cdot p(w|w'))$$

where $p(w|w')$ is the cosine similarity between term embeddings, given by FastText. We applied a minimum document frequency (cut-off) of 0.001%, and a maximum document frequency of 1%, yielding $6.6 \cdot 10^4$ terms. Figure 2c presents term frequencies. The minimum frequency filters words that appear too rarely to influence message proximity, while the maximum frequency indicates words that appear too often, and are therefore unlikely to reflect specific context. Figure 2d juxtaposes the two post similarity approaches on a random sample of 1000 posts; in this case, as opposed to the case of term similarity in Figure 3, we observe a high correlation. We opt for enriched TF-IDF as the more refined approach.

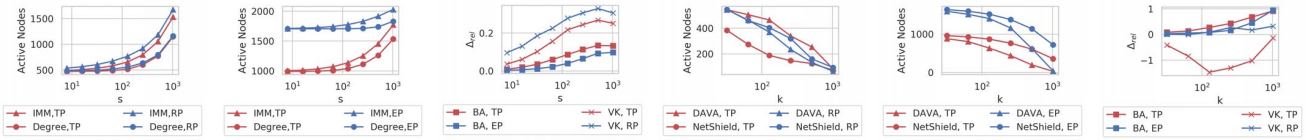
Eventually, we calculate edge probabilities by scanning the log of posts. We define influence probability as $p_{uv} = \frac{A_{v2u}}{A_v}$, where A_{v2u} is the number of posts by user v that are *similar* to an *earlier* post by u. We consider a pair of posts A from u, B from v, as *similar* only if (i) A satisfies a *content similarity threshold* τ with respect to B, i.e., $\cos(\text{tf-idf}_A, \text{tf-idf}_B) \geq \tau$, (ii) A precedes B by at most one month, and (iii) no earlier message B_2 from v satisfies τ with respect to A, and no later post A_2 from u satisfies τ with respect to B; in other words, we assume a user is influenced by a post *only once*. Figure 2e shows the resulting probability distribution.

For each post m and each neighbor of its author, we consider the existence of a similar post m' on the neighbor's wall as a *positive* instance of propagation. We use a cutoff threshold $\theta \in (0, 1)$ to determine our probabilistic predictions of propagation. Scanning the log of posts, as in [10], we derive True Positive and False Positive Rates for all values of θ, $\text{TPR} = \frac{TP}{TP+FN}$, $\text{FPR} = \frac{FP}{FP+TN}$. We evaluate the quality of the trained probability model by the Area Under Curve: $\text{AUC} = \int \text{TPR}\, d\text{FPR}$. Figure 2f presents AUC values for different *content similarity thresholds* τ. We select $\tau = 0.994$, which maximizes AUC (0.9062), as the default τ. Filtering zero-probability edges, under the chosen τ, yields a network of 2094 non-isolated nodes, which we use henceforward.

III. Applications

Here, we investigate the behaviour of real-world VK data in comparison to synthetic data generated by the *Barabási-Albert* (BA) model (Figure 1a) that simulates a power-law degree distribution, on two problems: Influence Maximization (IM), where the objective is to maximize expected spread of selected seeds, and Node Immunization (NI), where the objective is to select nodes to block/remove so that the expected spread of some seeds is minimized. We consider the data-aware NI problem, where seeds are known in advance [4]. We use the algorithm of Holme and Kim [23] that extends the original BA model, selecting parameters so that the BA graph has the same number of nodes and edges, and similar degree distribution (Figure 2a) and clustering coefficient (0.101) as the VK graph. We evaluate expected spread by 10,000 Monte-Carlo simulations. We also use a synthetic probability model, the trivalency model that randomly select low (0.01), medium (0.05) or high value (0.1), applied to both VK and BA data, and an exponential model on BA data, generating probabilities by the probability density function $\frac{1}{c} exp(-\frac{x-\mu}{c})$, where c and

[2] https://tech.yandex.ru/mystem/ [3] https://fasttext.cc/



(a) VK (b) BA (c) $\Delta_{rel} = \frac{\text{IMM} - \text{Degree}}{\text{Degree}}$ (d) VK (e) BA (f) $\Delta_{rel} = \frac{\text{NetShield} - \text{DAVA}}{\text{NetShield}}$

Fig. 4: Results on IM vs. seeds s (a-c); NI vs. blocked nodes k (d-f); Real (RP), Expon. (EP) and Trivalent (TP) prob.

μ are parameters fit to probability distribution of the VK data. The code and the resulting VK dataset are available[4] online.

A. Influence Maximization

We apply the IMM algorithm [24] with accuracy parameter $\epsilon = 0.1$ and the Degree heuristic [2] on the IM problem. Figures 4a-c show our results on different networks and probability models. Δ_{rel} shows the relative performance gap among two algorithms. Revealingly, on VK data, the performance of IMM deviates from that of the naive degree heuristic more with real probabilities than with trivalency (Fig. 4a), while on synthetic graphs, it deviates more with trivalency (Fig. 4b, 4c).

B. Node Immunization

Node Immunization can be solved in a data-aware manner, when seeds are known in advance, or preemptively, when they are not. We use the state-of-the solution for each case: DAVA [4] accepts a seed set S (calculated by IMM) as input and builds an NI solution informed by *domination* relationships among nodes with respect to S; it calculates the *benefit* of removing a node as $\gamma(v) = 1 + \sum_{u \in \text{children of } v} \gamma(u) \cdot p_{vu}$, where p_{vu} is the probability that influence propagates along the *most probable* path from v to u. NetShield [25] blocks preemptively a set of nodes, S, maximizing the *Shield value*:

$$Sv(S) = \sum_{i \in S} 2\lambda \mathbf{u}(i)^2 - \sum_{i,j \in S} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j)$$

where λ and $\mathbf{u}$ are the largest eigenvalue and the corresponding eigenvector of the network's adjacency matrix $\mathbf{A}$. A set S has high Sv if its elements have high eigenscore $\mathbf{u}(i)$ and are not connected to each other (zero $\mathbf{A}(i,j)$). A high eigenscore implies that their removal leads to a significant eigen-drop $\Delta\lambda$. Figures 4d-f present our results with these algorithms, with $|S| = 100$. Remarkably, DAVA outperforms NetShield in limiting spread all cases except with trivalency on VK. As in IM, using trivalency we overestimate the lead of the state-of-the-art algorithm on BA, and underestimate it on VK.

IV. Conclusions

We presented a framework for extracting influence probabilities for the independent cascade model, using textual content analysis and a vector representation of messages in a network. We showed that our trained model has good prediction power, and applied it experimentally on two network diffusion problems, influence maximization (IM) and network immunization (NI), with real-world and synthetic networks. Juxtaposing results obtained with probabilities derived by our framework to those obtained by synthetic ones, we find that state-of-the art algorithms for IM and NI express their lead with real-world probabilities on real-world networks; the use of a synthetic probability model amplifies that lead on synthetic networks but distorts it on real networks, especially in node immunization.

[4] https://github.com/iconvk/LearningIndependentCascadeOnVK

References

[1] J. Goldenberg, B. Libai, and E. Muller, "Talk of the Network: A Complex Systems Look at the Underlying Process of Word-of-Mouth," *Marketing Letters*, vol. 12, no. 3, pp. 211–223, 2001.

[2] D. Kempe, J. M. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *KDD*, pp. 137–146, 2003.

[3] Y. Li, J. Fan, Y. Wang, and K. Tan, "Influence maximization on social graphs: A survey," *IEEE TKDE*, vol. 30, no. 10, pp. 1852–1872, 2018.

[4] Y. Zhang and B. A. Prakash, "Data-aware vaccine allocation over large networks," vol. 10, pp. 20:1–20:32, 2015.

[5] P. Shakarian, A. Bhatnagar, A. Aleali, E. Shaabani, and R. Guo, "The independent cascade and linear threshold models," in *Diffusion in Social Networks*, pp. 35–48, Springer, 2015.

[6] W. Cui, X. Gong, C. Liu, D. Xu, X. Chen, D. Fang, S. Tang, F. Wu, and G. Chen, "Node Immunization with Time-Sensitive Restrictions," *Sensors*, vol. 16, no. 12, p. 2141, 2016.

[7] J. Zhang, J. Tang, J. Li, Y. Liu, and C. Xing, "Who influenced you? predicting retweet via social influence locality," *TKDD*, vol. 9, no. 3, pp. 25:1–25:26, 2015.

[8] K. Saito, R. Nakano, and M. Kimura, "Prediction of information diffusion probabilities for independent cascade model," in *Knowledge-based intelligent information and Engineering Systems*, pp. 67–75, 2008.

[9] P. Netrapalli and S. Sanghavi, "Learning the graph of epidemic cascades," in *SIGMETRICS*, pp. 211–222, 2012.

[10] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan, "Learning influence probabilities in social networks," in *WSDM*, pp. 241–250, 2010.

[11] N. Barbieri, F. Bonchi, and G. Manco, "Topic-aware social influence propagation models," *Knowl.Inf.Syst.*, vol. 37, no. 3, pp. 555–584, 2013.

[12] S. Ivanov, K. Theocharidis, M. Terrovitis, and P. Karras, "Content recommendation for viral social influence," in *SIGIR*, 2017.

[13] K. Subbian, C. C. Aggarwal, and J. Srivastava, "Content-centric flow mining for influence analysis in social streams," in *CIKM*, 2013.

[14] J. Cheng, L. A. Adamic, P. A. Dow, J. M. Kleinberg, and J. Leskovec, "Can cascades be predicted?," in *WWW*, pp. 925–936, 2014.

[15] L. Liu, J. Tang, J. Han, M. Jiang, and S. Yang, "Mining topic-level influence in heterogeneous networks," in *CIKM*, pp. 199–208, 2010.

[16] Z. Wang, C. Chen, and W. Li, "Information diffusion prediction with network regularized role-based user representation learning," *ACM Trans. Knowl. Discov. Data*, vol. 13, pp. 29:1–29:23, May 2019.

[17] T. Schank and D. Wagner, "Approximating clustering coefficient and transitivity," *J. Graph Alg. Appl.*, vol. 9, no. 2, pp. 265–275, 2005.

[18] I. Segalovich, "A fast morphological algorithm with unknown word guessing induced by a dictionary for a web search engine," in *MLMTA*, pp. 273–280, 2003.

[19] S. Bird, E. Klein, and E. Loper, *NLP with Python*. O'Reilly, 2009.

[20] R. Lage, P. Dolog, and M. Leginus, "Vector space models for the classification of short messages on sn services," *WEBIST*, 2013.

[21] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv:1607.04606*, 2016.

[22] Y. Su and X. Yan, "Cross-domain semantic parsing via paraphrasing," in *EMNLP*, pp. 1235–1246, 2017.

[23] P. Holme and B. J. Kim, "Growing scale-free networks with tunable clustering," *Physical review E*, vol. 65, no. 2, p. 026107, 2002.

[24] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: A martingale approach," in *SIGMOD*, pp. 1539–1554, 2015.

[25] C. Chen, H. Tong, B. A. Prakash, C. E. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, and D. H. Chau, "Node immunization on large graphs: Theory and algorithms," *IEEE TKDE*, vol. 28, no. 1, pp. 113–126, 2016.