Exploiting the Spam Correlations in Scalable Online Social Spam Detection

Hailu Xu, Liting Hu, Pinchao Liu, and Boyuan Guan

School of Computing & Information Science, Florida International University
Miami FL 33174, USA
{hxu017, lhu, pliu002, bguan003}@cs.fiu.edu

Abstract. The huge amount of social spam from large-scale social networks has been a common phenomenon in the contemporary world. The majority of former research focused on improving the efficiency of identifying social spam from a limited size of data in the algorithm side, however, few of them target on the data correlations among large-scale distributed social spam and utilize the benefits from the system side. In this paper, we propose a new scalable system, named SpamHunter, which can utilize the spam correlations from distributed data sources to enhance the performance of large-scale social spam detection. It identifies the correlated social spam from various distributed servers/sources through DHT-based hierarchical functional trees. These functional trees act as bridges among data servers/sources to aggregate, exchange, and communicate the updated and newly emerging social spam with each other. Furthermore, by processing the online social logs instantly, it allows online streaming data to be processed in a distributed manner, which reduces the online detection latency and avoids the inefficiency of outdated spam posts. Our experimental results with real-world social logs demonstrate that SpamHunter reaches 95% F1 score in the spam detection, achieves high efficiency in scaling to a large amount of data servers with low latency.

Keywords: Social Spam Detection, DHT-based Overlay.

1 Introduction

Online social networks (OSNs) have been an integral part of human life. More and more people are acquiring the latest news, advertisements, social activities, and breaking topics directly from the current popular OSNs such as Facebook, Twitter, and WeChat. For example, a report said that the percentage of US adults who primarily receive news and information from OSNs is as high as 62% [3]. However, the openness of widespread OSNs couple with massive spam activities, which are damaging as they cause public panic and social unrest. For example, in February of 2019, social users in Paris watched a lot of photos of kidnappings on Facebook and videos of vans speeding away on Snapchat and Twitter, all of which hinted that the Roma (Gypsies) robbed children with vans

in the suburbs of Paris [5]. Although the information proved to be wrong later, they brought serious consequences to the Roma and the whole society: dozens of young men wielding sticks and knives attacked a Roma camp and burned two vans, and tens of people were arrested. Another example is that one latest report said the global enterprise spam filter market was valued approximately USD 849 million in 2018 and is expected to generate around USD 2,675 million by 2026 [2]. And it pointed out that the increasing number of social spam is driving the enterprise spam filter market globally.

The unprecedented success of online social networks has created tremendous opportunities for the emergence and rapid spread of spam. By leveraging a large social user, social spam often dominates and influences social life in a short period of time and can reach every corner of the social world. Therefore, quickly detecting spam from large-scale social activities is an urgent need in the current situation.

Furthermore, as our observation, the spammers in the online social networks are not only active on a single platform, but are often active on different social platforms, by simultaneously manipulating dozens or hundreds of fake accounts. Naturally, the information published by these fake accounts is highly similar. This phenomenon has been pointed out by several former studies [28]. Spammers certainly desire to spread similar posts on different platforms to attract as many people as possible to target on these topics. A case study of social spam posts for multiple different news sites also demonstrates that spam posts show a high degree of similarity in content and topics during the same period of time and will immediately propagate from one site to another [1]. Therefore, this correlation between cross-platform social spam is a common phenomenon in the current social media world. Although there are not many direct relationships between users, geographic locations, creation purposes, and regions in these various groups or platforms, the spam contents are highly correlated within similar topics during the same period of time.

However, former studies rarely utilized the spam correlations to handle the large-scale social data from distributed data servers. They either focused on the algorithm side to achieve high accuracy in the detection [12, 20, 22, 23, 25], or the entire processing only targeted on a small size of dataset without the global view from similar data across large-scale data sources [7, 9, 26, 30]. In this paper, to explore the efficient method in dealing with large-scale social data sources, we present a new social spam detection system, named **SpamHunter**, to take advantage of the spam correlation among distributed data sources for efficient large-scale social spam detection. SpamHunter implements multiple groups, where each group contains a DHT-based functional tree that jointly connecting multiple data sources (e.g., servers, datasets, etc.) to share the spam correlations (e.g., updated spam features) in a distributed manner. The DHT-based functional trees response to data delivery, spam identification, and correlation exchanges. Besides, the group-level coordination ensures multiple groups or clusters can instantly exchange and share the correlated features during the process-

ing, that is, they collectively leverage the latest spam correlations to enhance the performance of spam detection.

This paper makes the following technique contributions:

- An ***distributed*** spam detection system named SpamHunter is presented that defend against social spam activities from large-scale social networks.
- The ***scalable*** DHT-based functional tree guarantees the orchestration and flexible management of a large amount of data sources.
- SpamHunter supports the use of the spam ***correlations*** among large-scale data sources to identify the latest spam activities and enables efficient online spam detection.

We outline the rest of this paper as follows: Section 2 presents the design and functional components of the system. Section 3 describes the detailed evaluation results with real-world social logs. Section 4 introduces the related works and we finally conclude this work in Section 5.

2 Design

In this section, we will introduce an overview of the SpamHunter system, describe the details of the functional components in the system, and outline the workflow for processing.

2.1 SpamHunter Overview

Figure 1 shows the designed architecture of the SpamHunter system. SpamHunter is built upon a peer-to-peer DHT-based Pastry overlay [17]. The overlay is utilized to orchestrate large-scale distributed social servers. As shown in the first step of Figure 1, these data servers can be grouped by various kinds of features (e.g., geo-location, topic tags, or institutions), and the large amount of servers are connected to the DHT-based overlay. In the second step, SpamHunter creates a functional tree upon Pastry for each group, where nodes jointly route around a specific key (see details in subsection 2.2). The functional tree for each group will respond to the primary workload during processing, for example, data dissemination, spam detection, and results aggregation.

In the third step, SpamHunter deploys online social spam detection within the group management. In each group, SpamHunter manages the functional tree to fulfill the online social data processing. The root of the tree is responsible for the data/model dissemination and in charge of the entire workflow. The distributed leaf nodes will complete the processing of spam detection by following the root's instructions by coordinating the classified models. The root of the tree also aggregates the identified results from the following nodes, updates the spam dataset, and extracts the latest spam. Furthermore, as shown in the fourth step of Figure 1, after the online spam detection, multiple groups in SpamHunter will periodically exchange and share the latest spam with others, so that all groups

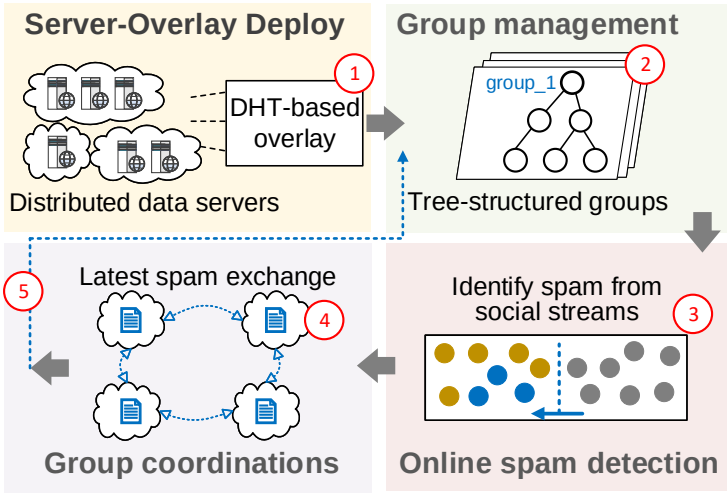


Fig. 1: The overview of the system design.

have a global view of the newest social spam and then utilize the correlated new spam in the continuing processing, as shown in the fifth step of the Figure 1.

Next, we will introduce the system's functionality and implementations details. We first introduce the deployment of SpamHunter and the group management. Then we introduce the online social spam processing. Finally, we propose the group coordination and communication in enhancing the detection performance by leveraging the large-scale spam correlations.

2.2 SpamHunter Group Management

We first present the details of the overlay in SpamHunter. SpamHunter is built upon the peer-to-peer Pastry overlay [17], where each node has a unique 128-bit nodeId with a nodeId space ranging from $0 \sim 2^{128} - 1$. Note that all nodeIds are evenly distributed, so that the deployment of nodes can be flexibly scaled to a large amount of instances. The message is the main link between nodes: nodes can route messages towards a specific key, for example, the key can be a target nodeId, a groupId, or a specific topic concatenates with a groupId. With the targeted key, messages can be routed to the node which nodeId is numerically closest to the key in $\lceil log_{2^b} N \rceil$ steps, where the default value of b is 4.

By leveraging Scribe [6], each node in SpamHunter can create a group by a groupId. Typically, the groupId is obtained by hashing (SHA-1) the name of the group with the name of its creator. Other nodes can randomly join a group by routing a JOIN message towards the groupId as the key, which enables flexible group membership. The nodeId of the rendezvous node in the group is closest in value to the groupId. Each group constitutes a functional tree which creates valid paths for the root to communicate with multiple layer nodes. The key idea is the

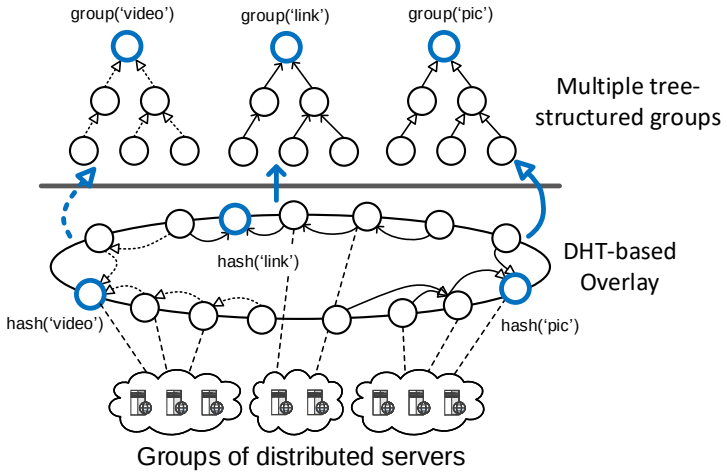


Fig. 2: The server-overlay structure and group management.

use of a DHT-based application-level multicast tree [6] to propagate data/model replicas through the tree path, which has the advantage of not maintaining N point-to-point connections for N leaf nodes. For example, assuming there are 7 nodes jointly work as group "*video*", if *hash*(video + creator name) equals to *EA34*, the node whose nodeId is closest to it, such like *EA34* or *EA35*, will serve as the root of the functional tree. The other six nodes will then subscribe to this tree and follow the root node. Due to the tree structure, the tree root can multicast the messages, instructions, or models to all leaf nodes in $O(logN)$ hops.

SpamHunter Tree's Functions. SpamHunter creates multiple groups to support the scalability of social data processing. Each group constitutes as a functional tree, where the spam detection is fulfilled in this tree. As shown in Figure 3, the group's functional tree mainly has four functions: *spam detection*, *aggregate function*, *spam extract*, and *external/inner tunnel*. We next present the details of these functions.

The *spam detection* is fulfilled by the coordination of the root and leaf nodes. In the group's tree, the tree root is in charge of the workflow of spam identification via the *inner tunnel* in root and leaf. The root of the tree will build a spam detection model by training the model using the training data set and then testing the model using the test data set. In addition, it manages the processing workflow by propagating instructions and models through the functional tree to the following branches and leaf nodes. By following the instructions of the tree root, the leaf nodes complete the pre-data processing and spam identification with the model. Details are presented in subsection 2.3.

The SpamHunter functional tree supports *aggregate function* during the processing to collect the interim results after spam detection. The tree branches and middle-level nodes are able to jointly work with the tree root to fulfill the

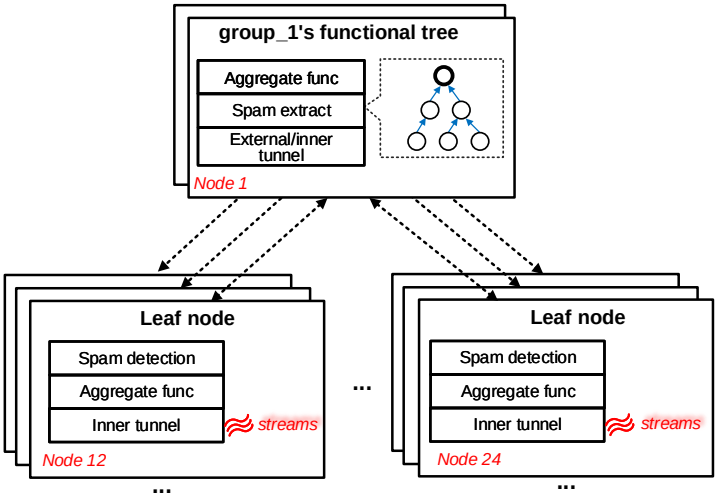


Fig. 3: The DHT-based functional tree.

aggregation. We next use an example to present it. After the local social spam classifications in the leaf nodes, batches of social logs are parsed as mappings from the content (*posts*) to categorical tag (*labels*), i.e., (*post_1*, 0) and (*post_2*, 1) in leaf nodes (here the tag 1 represents spam and the tag 0 represents non-spam). The leaf node will first filter out the identified spam data, (i.e., the social data has been detected as spam and marked with label 1), then sends the paired instances, e.g., (*post_i*, 1), to the upper layer via the *deliver tunnel*.

The third function supported in SpamHunter tree is the *spam extract*. After results aggregation, the tree root will accumulate the latest spam posts from the collected interim results and identify the prospected posts which are most highly be spam. For example, in a specific case, when 6 servers' interim results notify that the social post *post_k* as spam post, after the aggregation, the root will acquire the final votes for this post as (*post_k*, 6). The root node will extract this new identified spam post and join this post into the new training dataset, a set of data with identified spam and ham post which is used for creating spam models. After the default batch size, the root node will generate a new dataset consisting of latest spam posts and then periodically create a new spam model upon this dataset. After that, the tree root will disseminate the newly trained model to all following nodes in the continuing processing. Besides, when necessary, the tree root can multicast to its nodes within the group, to notify them to empty their sliding windows and/or synchronously start a new batch [29].

As shown in Figure 4, after the date processing in the tree-level, the *external tunnel* ensures multiple groups' roots exchanging and sharing the latest spam posts at the runtime, which means that each group can leverage spam correlation to enhance its own processing and get better performance. As mentioned earlier, in order to allow distributed data servers to obtain a global view of spam infor-

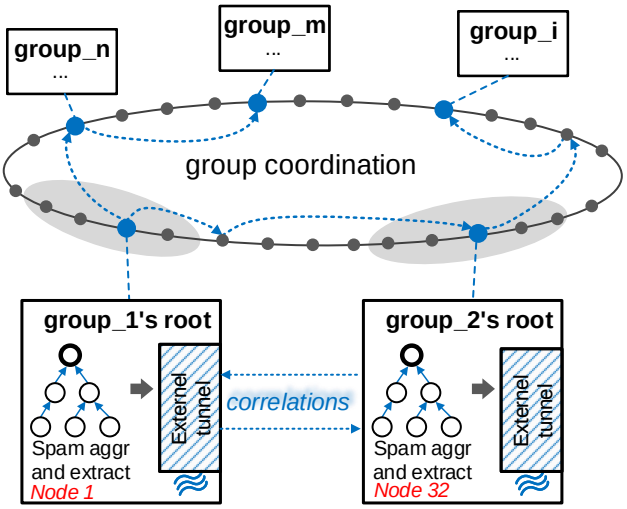


Fig. 4: The group coordination in the system. Group_1's root shares the spam correlations with group_2's root. Besides, the group_1's root communicates with other groups' roots via the overlay.

mation, SpamHunter allows the root of the group to send its aggregated spam to the roots of other peer groups. Details of data delivery among groups can be seen in Section 2.2. Each group's root can periodically update and exchange its extracted spam data with other groups.

SpamHunter ensures the entire spam detection in flexible processing granularity, including both globally large-scale data processing and locally distributed data processing. To support multiple processing targets, such as high latency sensitivity or good failure recovery, the functional tree is able to self-tune at the tree structure level by adjusting the tree fan-out element n, with achieving 2^n fan-outs per node.

Specifically, when the latency is the prime target of the users' defined applications, SpamHunter can customize the depth of functional tree by adjusting the fan-out element n. For instance, when 10^b (i.e., b = 4) nodes exist in the system, the original depth of the functional tree is $log_{2^b}(10^b)$. By adjusting the fan-out element from 4 (2^4) to 5 (2^5), the average depth of the tree can be pruned from 5 to 4. Consequently, the overall latency of root-to-leaf transmission will obtain 20% decrement.

2.3 Online Social Spam Detection

The SpamHunter leaf node is responsible for two functionality: (1) social raw log collection and normalization and (2) local spam detection. Each leaf node collects the social network logs (i.e., social posts, images, news, Tweets, and so on) from the distributed web servers. The leaf node can utilize the openly APIs (e.g., Twitter API, Facebook API) to collect the online/real-time streaming social logs. The logs will be collected from scripts and saved to the local

server, which can be utilized next by the leaf node that connected to this server. Furthermore, the leaf node will pre-process and normalize the raw social logs into the same formatted separate set. The majority of posts contain URLs, typically, to confuse the malicious URLs, spammers will add white spaces and unicode characters into them [10]. This is a simple but effective way to bypass the filters that blacklist URLs only by simple string matching. Inspired by [10], we de-confuse URLs by removing whitespace padding and normalizing the encoded characters (e.g., subsexvideo%26ip%3Dauto%26click%3D1 becomes subsexvideo&ip=auto&click=1). For social contents, specifically, we remove punctuation, tokenize each word, and remove stopwords. We extract the tf-idf values of the terms in each document. The tf-idf weight of the term represents the frequency at which the term appears throughout the document [32].

After the data collection, SpamHunter leaf node will first normalize the original data into unified formats. The SpamHunter leaf node extracts the posts' contents from the JSON formatted log. Then it divides these social data into same sized datasets for the local online spam detection. Next, the leaf node will utilize the trained spam model which is disseminated from SpamHunter root to complete the local data processing task. Original data which consists of unprocessed social logs without identified labels. After the spam classification with the trained model, an identified label will be created to each instance of the original social logs (here 1 presents spam and 0 presents non-spam). Besides, the SpamHunter leaf node will facilitate completing the results aggregation flow by sending intermediate results to the upper layers. Note that the leaf nodes will instantly process the collected social logs without long latency. Besides, they will follow the root's instructions to clean its slides and start new batches with the updated spam model.

The online social spam detection is completed by coordinating both tree-level and group-level. The tree-level processing has been presented before, we next introduce the group-level coordination in online processing, which primarily relies on group communications.

2.4 SpamHunter Group Coordination

The group communications in SpamHunter are responsible for the main function of spam correlated model update and data exchange among the entire detection. We fulfill the group communication by implementing diffusion broadcasting group. We now present the details of this functional component. SpamHunter group provides two major functions: *multicast* and *anycast*. *Multicast* is used to construct a hierarchical functional tree, which acts as a fundamental frame for scalability in SpamHunter. *multicast* allows messages or instructions can be delivered to all the members in one group. As presented before, any nodes can create a group in the overlay; and other nodes can flexibly join the group and then *multicast* messages from the rendezvous point to all member of the group along the functional tree.

Anycast can be used for group communications and model transmissions among multiple groups. It is implemented by the distributed depth-first search

(DFS). Each node in the overlay (may in/out of group k) can *anycast* to the group k by routing a message towards the group k's groupId [17]. The convergence of local routing in Pastry guarantees that this message can highly reach a group member near the sender's nodeId. *Anycast* can also be used to serve the communications between multiple groups, such as exchanging the updated dataset and exploring the spam correlations among them.

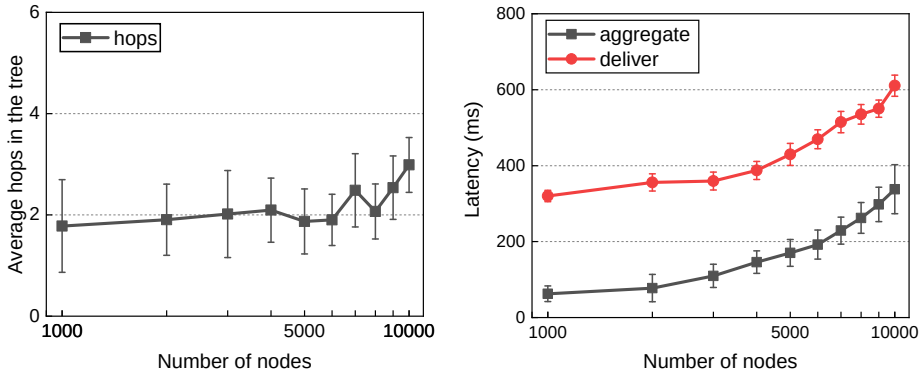
SpamHunter supports group-level communications to allow multiple groups to exchange their updated models to enhance the final performances. Once a group finishes its whole processing in the leaf nodes, the root node aggregates the results that contain the newest spam information and then updates its model. Further, root nodes in groups exchange the updated models by disseminating their updated models to other peer-groups. Then all groups own the newest models from other groups and can utilize the new models in the continuing processing.

SpamHunter originally supports star group that allows each root of one group *anycasts* the updated model to other groups. Given a graph G with N nodes, one root needs to send $n - 1$ messages during one round time. In this case, SpamHunter group has to send out $m = 1/2 \times n \times (n - 1)$ messages which takes $O(n^2)$ time. To diminish the group communication latency, we design diffusion group in SpamHunter. We now present the details of this type of group communication.

The diffusion group communication lies in: each root node in a group holds a table where contains the model version and the original group, which denotes as a $<groupID, versionNum>$. The root within updated model in one group randomly chooses two other groups to disseminate the model with the new version number. The root of these two groups will check its model table to see if the current received model is the latest one. If the *versionNum* is larger than the value in the table, they will save the model and update the model table with the new version number. If not, they will return a message to the original group to notify they already own the newest one. These two roots will act as new propagators and begin to deliver the latest model to other groups. Finally, all groups' roots will receive the updated model and update their model tables. It's easy to refer that the number of rounds to propagate a single update model to all groups is $O(logn)$, where n is the number of groups in SpamHunter.

3 Evaluations

The experimental evaluation of the system is carried out with online real-world streaming data from social media. We utilize the data which is collected from Twitter streaming APIs [29]. We manually labeled the dataset for examining the performance of spam detection. These data were labeled based on the posts' URL, content, and Twitter official identifications. The dataset contains 60,000 posts which including of 43,897 ham posts and 26,100 spam posts. The applications purpose is to identify social spam posts, which produces predicted labels



(a) Average communication hops. (b) Latency of deliver and aggregate.

Fig. 5: The average hops of node communications, the latency of delivery and aggregation in the SpamHunter system. (a) represents the average number of hops when nodes communicate in the functional tree. (b) shows the average latency of delivery and aggregation in the functional tree, the delivery mainly refers to model dissemination and the aggregation mainly refers to the results aggregation.

from online data streams via the system. Note that near 1'000,000 posts are used for evaluating the scalability of the system.

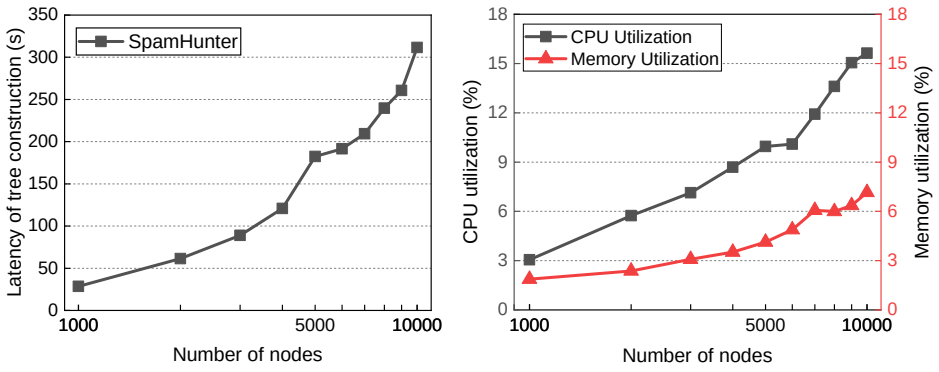
Experiments are conducted on a testbed of 10,000 nodes hosted by 10 servers. Each server has a 3.4GHz CPU, 4G of memory and 30 GB hard drives. Our evaluations mainly answer the following questions:

- What are the performances of system metrics in the scale, such as the delivery and aggregation latency, functional tree construction latency, and runtime overhead (Sec. 3.1)?
- What are the performances of group coordination and online spam detection by utilizing the spam correlations (Sec. 3.2)?

3.1 System performances with scalability

SpamHunter achieves effective spam detection in large-scale online social data sources, therefore, scalability is a major part of the overall evaluation. To evaluate SpamHunter, we deploy the system with the nodes ranging from 1,000 to 10,000, which consists of ten groups (functional trees) in the cluster of servers.

SpamHunter implements functional tree to complete the local distributed social spam detection, which means that the functionality of the tree directly affects the final performance of the process. We first look at the tree paths in the group. The functional tree responses for the message routing, delivery, and communications between multiple layers of nodes, therefore, the average hops (steps) among node communication should affect the performances within scaling to a large amount of data sources. The evaluated average hops are shown in Figure 5a. From this figure, we can see that the average hops between multiple layers of



(a) Average tree construction time. (b) CPU and memory utilization.

Fig. 6: The average functional tree construction time and the runtime overhead in CPU and memory. (a) represents the average latency when creating functional tree with different numbers of nodes in SpamHunter. (b) shows the runtime overhead of CPU and memory utilization.

nodes are consistent when the system scales to large amount of nodes. The typical hops among the functional tree are around 2 to 3. This demonstrates that SpamHunter can flexibly and conveniently support large-scale data sources and servers, and can guarantee the total communication between servers/instances in a relatively small distance, which indicates the low latency in handling node interactions and communications.

The communication latency among SpamHunter is mainly from two parts: the delivery latency from the root and the aggregations latency from the leaf. The delivery latency refers to the root node of the functional tree disseminating the messages, data, spam model, and instructions to all following nodes. The aggregation latency generally refers to the root node aggregates the interim results (i.e., identified social spam posts) from the leaf nodes. The results of these two kinds of latency are shown in the Figure 5b. From this figure, we can observe that when scaling the nodes to a large scale (up to 10,000), the latency is slowly increased with a few hundreds of milliseconds. This is reasonable since a large amount of nodes will cause part of delay in the message delivery and results aggregation. The difference between these two kinds of latency is usually from the delivered data size, for example, the delivered spam model is up to several megabytes, which causes the delivery latency is higher than the result aggregation latency.

Moreover, we evaluate the latency in constructing functional trees with a different number of nodes. As shown in Figure 6a, when the nodes scale from 1,000 to 10,000 (note that these are ten trees here, for each tree, the nodes scale from 100 to 1,000), the construction latency is linearly increased with the nodes' increment. The latency is usually from the hash of nodeId and the joining of the overlay. Note that the functional tree only needs to be built once at the beginning, and it will not cause other latency during the data processing.

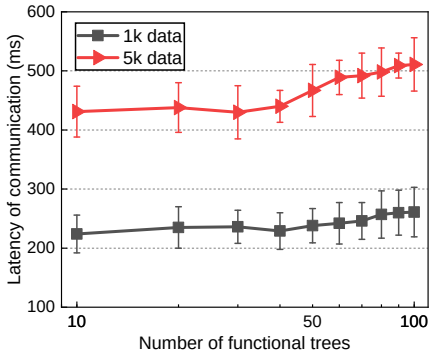


Fig. 7: Group communication latency.

Table 1: Results of spam detection.

Model	F1	Precision	Recall
RF	0.951	0.951	0.951
SVM	0.942	0.945	0.944
RT	0.927	0.928	0.927
Logistic	0.859	0.866	0.855

Further, we evaluate the runtime overhead of the system in deploying the functionality. Results of the CPU and memory utilization are shown in Figure 6b. The values of utilization present the overhead in one server and here they leave out the processing of spam detection since the data processing will periodically cost lots of computations and will make the overhead confusing in evaluating the functional tree's performances. From this figure, we can see that with the number of nodes scales to 10,000, the runtime overhead linearly increases by 16% in CPU and 7% in memory. It presents that SpamHunter achieves relatively lightweight overhead in guaranteeing the tree and group functions at runtime. And it can be beneficial for the large-scale data processing in the future.

3.2 Group coordination and spam detection

SpamHunter supports multiple groups to exchange and share the latest spam posts with each other. In this subsection, we present the experimental results of group coordination and the performance of spam detection.

Figure 7 shows the latency of communication among multiple groups' roots. Here we use two sizes of dataset, with 1,000 and 5,000 posts separately. From the figure, we can see that the latency of communication is consistent with the increment of groups from 10 to 100. When deployed with a large number of groups, for instance, 100 groups, the average latency has linear increment. In general, the latency mainly depends on the size of the delivered data. When the root shares a large size of spam posts, it will incur longer latency.

Table 1 presents the performances of spam detection in SpamHunter. We implement several classical algorithms such like RF (Random Forest), SVM (Support Vector Machine), RT (Random Tree), and Logistic in the detection with the labeled dataset. We present the major parameters of the performance including F1, Precision, and Recall, where the F1 score responses for an important factor in measuring the performance. From the table we can see the Random Forest (RF) achieves the best performance with the F1 score near 95%. This presents that SpamHunter can achieve good performances in dealing with the online real-world social spam data.

4 Related Work

There is a large body of studies that identified social spam from a variety of perspectives, such like from the view of follower/friend relationships [24], social accounts' behavior/activities patterns [18], and the review and user linguistic [21]. However, these studies are still limited to historical data of a certain size and are difficult to adapt to current online large-scale social data streams and applied to changing spam activities. Some recent applications had begun to integrate with scalable platforms for efficient processing [4, 14, 16, 19]. For instance, a model within CELAR cloud platform is designed to catalog the distributed, dynamic and redundant cancer data [27]. Different from them, we consider to utilize spam correlations from various distributed data sources and achieve online social data processing.

Several prior studies had present the correlations between feature-similar social networks or social activities. Coviello et al. [8] found that online social network owes large-scale spillover where individual activities may ripple through whole social networks to generate large-scale synchrony. GSRank [15] demonstrated that the spam group presented strong correlation in their behaviors with the statistical validation. [13] presented the high correlation between social owners comments and the followers' comments, and demonstrated that the user interactions also represent correlations. [11, 31] presented that social influence now is a common phenomenon among social networks where highly connected users or central users of one group/activity will be the core disseminator of contents. Besides, more shared contents will get higher attention and propagation in a short period of time. Inspired by the correlations among large-scale data sources, we choose to apply the spam correlations from distributed data sources into traditional social spam detection and with achieving the large-scale online data processing in multiple data servers.

5 Conclusion

Social spam has become an inevitable part of the current social world. Various garbage activities surround people and cause huge negative impacts on both virtual and real life. In this paper, we present an online social spam detection system, named SpamHunter, which leverages the spam correlations among large-scale distributed data sources to enable efficient spam detection in a scalable manner. SpamHunter supports multiple groups to manage social data from various topics, areas, and geo-location. Each group forms a functional tree that guarantees flexible management across a large number of data servers/instances. Moreover, group coordination in SpamHunter allows multiple groups to exchange and share spam correlations from distributed data sources, enabling efficient processing with the latest social spam from online data streams. In future work, we will explore the deployment of extensions across multiple social platforms, which will build a unified platform for orchestrating social spam from a global view.

Acknowledgment

We gratefully thank the anonymous reviewers for their feedback that significantly improved the paper. We thank Florida International University School of Computing and Information Sciences for the travel award to present this work. This work is supported by Department of Homeland Security (DHS 2017-ST-062-000002).

References

1. How fake news goes viral: A case study (2016), https://www.nytimes.com/2016/11/20/business/media/how-fake-news-spreads.html
2. Global enterprise spam filter market (2019), https://www.zionmarketresearch.com/report/enterprise-spam-filter-market
3. Allcott, H., Gentzkow, M.: Social media and fake news in the 2016 election. Journal of economic perspectives 31(2), 211–36 (2017)
4. Bhimani, J., Mi, N., Leeser, M.: Performance prediction techniques for scalable large data processing in distributed mpi systems. In: Performance Computing and Communications Conference (IPCCC), 2016 IEEE 35th International (2016)
5. Breeden, A.: Child abduction rumors lead to violence against roma in france (March 2019), https://www.nytimes.com/2019/03/28/world/europe/roma-kidnap-rumors-france.html
6. Castro, M., Druschel, P., Kermarrec, A.M., Rowstron, A.I.: Scribe: A large-scale and decentralized application-level multicast infrastructure. IEEE Journal on Selected Areas in communications 20(8), 1489–1499 (2002)
7. Chen, C., Wang, Y., Zhang, J., Xiang, Y., Zhou, W., Min, G.: Statistical features-based real-time detection of drifted twitter spam. IEEE Transactions on Information Forensics and Security 12(4), 914–925 (2017)
8. Coviello, L., Sohn, Y., Kramer, A.D., Marlow, C., Franceschetti, M., Christakis, N.A., Fowler, J.H.: Detecting emotional contagion in massive social networks. PloS one 9(3) (2014)
9. Gao, H., Chen, Y., Lee, K., Palsetia, D., Choudhary, A.N.: Towards online spam filtering in social networks. In: NDSS. vol. 12, pp. 1–16 (2012)
10. Gao, H., Hu, J., Wilson, C., Li, Z., Chen, Y., Zhao, B.Y.: Detecting and characterizing social spam campaigns. In: Proceedings of the 10th ACM SIGCOMM conference on Internet measurement. ACM (2010)
11. Hodas, N.O., Lerman, K.: The simple rules of social contagion. Scientific reports 4, 4343 (2014)
12. Hoefler, T., Barak, A., Shiloh, A., Drezner, Z.: Corrected gossip algorithms for fast reliable broadcast on unreliable systems. In: Parallel and Distributed Processing Symposium (IPDPS) (2017)
13. Jiang, J., Wilson, C., Wang, X., Sha, W., Huang, P., Dai, Y., Zhao, B.Y.: Understanding latent interactions in online social networks. ACM Transactions on the Web (TWEB) (2013)
14. Kayes, I., Iamnitchi, A.: Privacy and security in online social networks: A survey. Online Social Networks and Media (2017)
15. Mukherjee, A., Liu, B., Glance, N.: Spotting fake reviewer groups in consumer reviews. In: Proceedings of the 21st international conference on World Wide Web. ACM (2012)

16. Pop, D., Iuhasz, G., Petcu, D.: Distributed platforms and cloud services: Enabling machine learning for big data. In: Data Science and Big Data Computing, pp. 139–159. Springer (2016)
17. Rowstron, A., Druschel, P.: Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing. pp. 329–350. Springer (2001)
18. Ruan, X., Wu, Z., Wang, H., Jajodia, S.: Profiling online social behaviors for compromised account detection. IEEE transactions on information forensics and security 11(1), 176–187 (2016)
19. Salaria, S., Brown, K., Jitsumoto, H., Matsuoka, S.: Evaluation of hpc-big data applications using cloud platforms. In: Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (2017)
20. Sedhai, S., Sun, A.: Effect of spam on hashtag recommendation for tweets. In: Proceedings of the 25th International Conference Companion on World Wide Web. pp. 97–98. International World Wide Web Conferences Steering Committee (2016)
21. Shehnepoor, S., Salehi, M., Farahbakhsh, R., Crespi, N.: Netspam: A network-based spam detection framework for reviews in online social media. IEEE Transactions on Information Forensics and Security 12(7), 1585–1595 (2017)
22. VanDam, C., Tan, P.N.: Detecting hashtag hijacking from twitter. In: Proceedings of the 8th ACM Conference on Web Science. ACM (2016)
23. Viswanath, B., Bashir, M.A., Crovella, M., Guha, S., Gummadi, K.P., Krishnamurthy, B., Mislove, A.: Towards detecting anomalous user behavior in online social networks. In: USENIX Security Symposium (2014)
24. Wang, A.H.: Don't follow me: Spam detection in twitter. In: 2010 international conference on security and cryptography (SECRYPT). pp. 1–10. IEEE (2010)
25. Wang, D., Pu, C.: Bean: a behavior analysis approach of url spam filtering in twitter. In: Information Reuse and Integration (IRI), 2015 IEEE International Conference on. IEEE (2015)
26. Xie, W., Zhu, F., Jiang, J., Lim, E.P., Wang, K.: Topicsketch: Real-time bursty topic detection from twitter. IEEE Transactions on Knowledge and Data Engineering 28(8), 2216–2229 (2016)
27. Xing, W., Jie, W., Tsoumakos, D., Ghanem, M.: A network approach for managing and processing big cancer data in clouds. Cluster Computing 18(3) (2015)
28. Xu, H., Guan, B., Liu, P., Escudero, W., Hu, L.: Harnessing the nature of spam in scalable online social spam detection. In: 2018 IEEE International Conference on Big Data (Big Data). IEEE (2018)
29. Xu, H., Hu, L., Liu, P., Xiao, Y., Wang, W., Dayal, J., Wang, Q., Tang, Y.: Oases: An online scalable spam detection system for social networks. In: 2018 IEEE 11th International Conference on Cloud Computing (CLOUD) (2018)
30. Xu, H., Sun, W., Javaid, A.: Efficient spam detection across online social networks. In: 2016 IEEE International Conference on Big Data Analysis (2016)
31. Zhang, J., Tang, J., Li, J., Liu, Y., Xing, C.: Who influenced you? predicting retweet via social influence locality. ACM Transactions on Knowledge Discovery from Data (TKDD) 9(3), 25 (2015)
32. Zhang, Y., Hong, J.I., Cranor, L.F.: Cantina: a content-based approach to detecting phishing web sites. In: Proceedings of the 16th international conference on World Wide Web. pp. 639–648. ACM (2007)