SocialCache: A Pervasive Social-Aware Caching Strategy for Self-Operated Content Delivery Networks of Online Social Networks

Tiancheng Guo, Yuke Ma, Mengying Zhou, Xin Wang, Jun Wu, Yang Chen
Shanghai Key Lab of Intelligent Information Processing, School of Computer Science, Fudan University, China
{*tcguo20,myzhou19,xinw,wujun,chenyang*}*@fudan.edu.cn, ykma22@m.fudan.edu.cn*

***Abstract*—Online Social Networks (OSNs) play a significant role in people's daily life. Increasing OSN traffic promotes the requirement for building self-operated Content Delivery Networks (CDNs) to deliver OSN media data efficiently and reduce traffic costs. OSN data in CDNs is heavily influenced by social connectivity, such as friendships. To reduce CDN network traffic by better using social connectivity information, we propose SocialCache, a pervasive social-aware caching strategy in self-operated CDNs. SocialCache supports several social connectivity metric options that reflect the importance of the OSN users and the popularity of the media files. Through the improvement of the cache replacement algorithm in the CDN node and the communication design between nodes, SocialCache realizes the optimization of network traffic. Worth mentioning, SocialCache can easily integrate into mainstream CDN architectures while protecting user privacy. We implement SocialCache on Mininet, using real-world network measurements for CDN hierarchy and a hill-climbing algorithm for parameter selection. SocialCache outperforms a range of state-of-the-art baselines on three real-world OSN datasets. On the Twitter dataset, SocialCache reduces the network traffic volume by 6.40% and improves the cache hit ratio by 14.22%.**

***Index Terms*—Online Social Networks, Self-Operated Content Delivery Networks, Social-Aware, Caching Strategy**

1. Introduction

Online Social Networks (OSNs) [1] are developing rapidly, attracting billions of users worldwide. Given the high popularity of OSN services, pictures and videos transferred on OSNs generate enormous network traffic [2]. To cope with the massive network traffic and worldwide requests, many OSNs choose CDN services to assist the data delivery [3]. Although many existing commercial CDN service providers like Akamai [4] and Amazon CloudFront [5] offer high-quality and convenient CDN services, OSN providers increasingly prefer to use self-operated CDNs [6]. Compared with commercial CDNs, self-operated CDNs provide OSN providers with better cost reduction [6], privacy protection [7], and network controllability [8].

Numerous caching strategies benefit from the information available to CDNs, such as network conditions [6] and geographic locations [9]. The trend of OSNs using self-operated CDNs brings several research opportunities. For instance, the optimization of CDN caching strategies with finer-grained user-activity information [10]. Many studies considered content prefetching based on OSN information [11] [12] [13] to improve user experience. However, existing studies did not explore social connectivity information deeply.

We propose our solution, **SocialCache**, to fill the gap above. SocialCache is a pervasive caching strategy that uses different types of social connectivity information for CDN optimization, especially in the emerging situation of self-operated CDNs of OSNs. The core idea of SocialCache is to reduce network traffic between different CDN nodes, thus primarily saving the service cost of self-operated CDNs [14] and secondarily enhancing user experience [15]. Contents recently created by high-impact OSN users are accessed more frequently by geographically neighboring users [16], and contents with large files have a greater impact on CDN's network traffic [5]. Therefore, giving higher priority and longer retention times to such content can reduce traffic costs and maintain cache hit ratios. In addition, SocialCache is readily deployable in increments, requiring only minor adjustments to the caching strategy of existing self-operated or commercial CDNs.

We implement SocialCache using a well-known network simulator called Mininet. Real-world network measurements are carried out to deploy a mainstreaming three-layer CDN hierarchy. A heuristic algorithm named hill-climbing is adopted for parameter selection. We use three real-world OSN datasets to validate SocialCache. The performance of the SocialCache is evaluated from different aspects and compared with several representative baselines. Experiment results show that SocialCache vastly reduces the network traffic volume and maintains high cache hit ratios. Meanwhile, SocialCache is a pervasive caching strategy that performs well on different datasets while maintaining high computational efficiency.

We summarize our key contributions as follows:

This work is supported by the National Natural Science Foundation of China (No. 62072115, No. 61971145, No. 61831018, No. U21A20452). Yang Chen is the Corresponding Author.

- We propose a caching strategy called SocialCache, which is pervasive, easy to use, and privacy-preserved, considering social connectivity information for optimization in self-operated CDNs.
- We implement SocialCache using a popular network simulator named Mininet. The implementation of SocialCache includes a three-layer CDN hierarchy based on real-world network measurements and parameter selection using hill-climbing.
- We compare SocialCache with four representative baselines on three real-world OSN datasets. Experiment results show that SocialCache outperforms other caching strategies on three datasets. On the largest Twitter dataset, SocialCache reduces the network traffic volume by 6.40%, improves the cache hit ratio by 14.22%, and reduces the operation time by a factor of 20x, compared with the state-of-the-art social-aware caching strategy called LRU-Social [3].

2. Method

In this section, we first introduce preliminary knowledge in §2.1. After elaborating on the design of SocialCache in §2.2, we describe how SocialCache protects user privacy and maintains scalability in §2.3.

2.1. Preliminary

2.1.1. CDN Hierarchy. We build a three-layer CDN hierarchy illustrated in Figure 1, which is widely used for mainstream CDN service providers, like Alibaba [15], Akamai [4] and Amazon [5]. Each CDN layer contains several CDN nodes. Only adjacent CDN layers have data transmission, and cache contents are not shared within the CDN nodes of the same layer [5]. The CDN nodes in the lower CDN layer are closer to the users and provide low-latency services. The selection of locations of higher-layer CDNs is cost-conscious, with a relatively small number of CDN nodes but better machine performance. Once a user's request is sent to the geographically nearest CDN node of the L1 CDN layer [17], SocialCache, which runs in every CDN node, will handle all subsequent operations.

2.1.2. Actions: POST and VIEW. There are two major request actions for OSN users: POST and VIEW [1] [18]. POST means a user publishes a post (e.g., a tweet) through its OSN account, sharing it with others. VIEW is the activity of browsing other users' posts. We assume that each request action is either POST or VIEW.

2.1.3. Social Connectivity Metrics. There are two kinds of OSN graphs: a subscriber-publisher-oriented directed graph (e.g., Twitter and Weibo) and a chatting-oriented undirected graph (e.g., Facebook and WeChat). We focus on the kind of directed graphs because where media files like pictures and videos are extremely large and these OSNs are more dependent on CDN services. To better utilize the social connectivity information of the OSN graph, SocialCache supports five representative social connectivity metrics [19] [20] to represent the influence of each user:

In-degree: In-degree is a fundamental metric in an OSN graph, identifying the number of a user's followers. The higher the in-degree is, the more followers the user has, and the higher the exposure possibility of the user's posts.

PageRank: PageRank is originally a website ranking algorithm used by Google, which researchers extend to explain a user's influence [19]. A user with a high PageRank value usually has an important impact.

Laplacian centrality: Laplacian centrality is defined as the drop in the Laplacian energy of a graph when a node is removed [21]. A user with higher Laplacian centrality who leaves the social network will jeopardize the community more severely.

Betweenness centrality: Betweenness centrality is another important centrality metric on an OSN graph [22], which measures each pair of nodes based on the shortest paths. An important user on the spread of information flow in a social network will have a high betweenness centrality. A such user often serves as a bridge from one part of a graph to another and tends to have a greater influence.

Effective size: Effective size is a concept in structural hole theory [23]. Structural hole spanners who occupy important positions will serve as bridges among different communities in the social network, tending to have higher values of effective size [20]. Effective size shows how many different active communities a user is connected to, ignoring redundant connections and associating multiple connections to the same community with one. Therefore, a user with a higher effective size value will affect a wider range of users. Moreover, compared with the other social connectivity metrics, the calculation of effective size only requires the information of a user's ego network, without the information of the whole OSN graph. This efficient calculation is more practical and feasible, as well as better privacy-concerned. Therefore, we choose effective size as the default social connectivity metric for SocialCache.

2.2. System Design

The design paradigm of SocialCache is depicted in Figure 1. It could be divided into two parts, i.e., POST and VIEW, consistent with the two types of request actions. For a POST action, SocialCache solves the insertion and replacement of the cache and recursively sends requests to the corresponding CDN nodes in higher CDN layers. For a VIEW action, SocialCache handles the indexing and updating of the cache and responses to the user.

POST: If a user publishes a post, the system will ask the client first to calculate a Priority $\mathcal{P}$ with Equation 1. The client will use four features to calculate the $\mathcal{P}$.

$$\mathcal{P} = \mathcal{W}_{\mathbf{0}} \cdot \mathcal{S} + \mathcal{W}_{\mathbf{1}} \cdot \mathcal{M} + \mathcal{W}_{\mathbf{2}} \cdot \mathcal{D} + \mathcal{T} \tag{1}$$

$\mathcal{S}$ is the social connectivity metric we calculated to quantify the user's influence in the OSN, which we have discussed in §2.1.3. The feature $\mathcal{M}$ denotes the size of

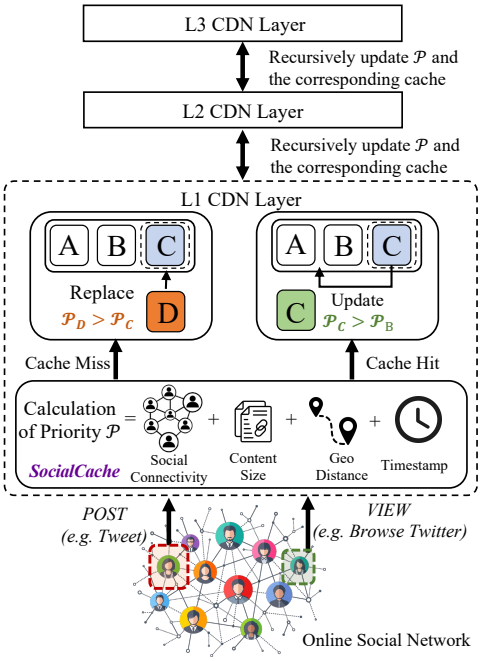

Figure 1. Architecture of SocialCache with a three-layer CDN hierarchy.

Algorithm 1: Workflow of SocialCache

Input: A Request $\mathcal{RQ}$ with Media HashID $\mathcal{MH}$, Request Action $\mathcal{RA}$, Timestamp $\mathcal{T}$, Priority $\mathcal{P}$
Output: Operations $\mathcal{OP}$

if $\mathcal{RA}$ == *POST* **then** // POST
  $\mathcal{E}_{coming} \leftarrow$ {key: $\mathcal{MH}$, value: $\mathcal{P}$, $\mathcal{T}$};
  if $\mathcal{MH} \in$ *Cache* **then**
    $\mathcal{E}_{exist} \leftarrow$ Cache.Find({key: $\mathcal{MH}$});
    $\mathcal{E}_{exist} \leftarrow \mathcal{E}_{coming}$;
  else
    if *Cache.Count*($\mathcal{E}_{exist}$) == *Cache.Size()* **then**
      Cache.Remove($\mathcal{E}_{exist}$ with minimum $\mathcal{P}$);
    end
    Cache.Insert($\mathcal{E}_{coming}$);
  end
  $\mathcal{OP} \rightarrow$ Send $\mathcal{RQ}$ to higher CDN layer;
else // VIEW
  if $\mathcal{MH}$ *In Cache* **then** // Cache Hit
    $\mathcal{E}_{exist} \leftarrow$ Cache.Find({key: $\mathcal{MH}$});
    $\mathcal{P}_{new} \leftarrow$ Calculate with new $\mathcal{T}$;
    $\mathcal{E}_{exist} \leftarrow$ Cache.Update({key: $\mathcal{MH}$, value: $\mathcal{P}_{new}$, $\mathcal{T}$});
    $\mathcal{OP} \rightarrow$ Send $\mathcal{E}_{exist}$ to lower CDN layer;
  else // Cache Miss
    $\mathcal{OP} \rightarrow$ Send $\mathcal{RQ}$ to higher CDN layer;
  end
end

the media file transferred by this request, $\mathcal{D}$ represents the reciprocal of the geographic distance between the user and the nearest CDN node, and $\mathcal{T}$ is the timestamp when the POST event is triggered. Three parameters $\mathcal{W}_0$, $\mathcal{W}_1$, and $\mathcal{W}_2$, together determines the weights of the different features. To strike a balance between accuracy and efficiency, we use a classic heuristic algorithm called hill-climbing [24] to fine-tune these parameters.

SocialCache majorly uses $\mathcal{P}$ to handle cache operations. A CDN node has several elements named $\mathcal{E}_{exist}$ in its cache. When a request arrives at a CDN node, the information and content it carries are wrapped into a single element called $\mathcal{E}_{coming}$. If the same media content as $\mathcal{E}_{coming}$ exists in the cache, regardless of which user publishes the media content, the information of the content in the cache is updated without consequent replacement operations. If the current cache is not full, the insert operation is performed; if the current cache is full, the $\mathcal{E}_{exist}$ with the lowest $\mathcal{P}$ is replaced with $\mathcal{E}_{coming}$. Once the caching operation in the CDN node is completed, the POST request will be recursively sent to the corresponding node in a higher CDN layer. The details are described in Algorithm 1.

VIEW: If a user views a post, the request will be directly sent to the geographically nearest CDN node. If the post exists in an $\mathcal{E}_{exist}$, it is a cache hit event that updates $\mathcal{P}_{new}$. Otherwise, it is a cache miss event. The VIEW request will be recursively sent to the higher-level CDN node, with the returned element $\mathcal{E}_{coming}$ inserted into this cache like POST action. See Algorithm 1 for the detailed workflow.

2.3. Privacy and Scalability

In the scenario of self-operated CDNs, SocialCache protects the user's privacy and maintains scalability. On the one hand, OSN users' requests are transmitted in the self-operated CDN, and data logs are recorded internally. On the other hand, the default social connectivity metric is effective size, which calculation requires only the data of an OSN user and its ego network. The selection of effective size also ensures scalability because the graph size of a user's ego network is usually small enough to calculate fast [20].

However, although OSN providers tend to use self-hosted CDNs, commercial CDNs are still indispensable in reality [5]. When requests are forwarded to commercial CDNs, the information carried by the requests will be at risk of being exposed [2]. The design of SocialCache considers such scenarios, which also carefully protect users' privacy and maintain scalability in sophisticated hybrid CDN scenarios. In a request, $\mathcal{P}$ is the only extra information that can be computed locally from the user side. It masks the origin data, which is almost impossible to deduce back. Meanwhile, if a commercial CDN node receives a request, it can operate as normally as it handles other requests without additional modifications, just needing to ignore the extra information $\mathcal{P}$. It is difficult for a commercial CDN to deanonymize user data by computing the metric $\mathcal{P}$, because $\mathcal{W}_0$, $\mathcal{W}_1$, $\mathcal{W}_2$ and $\mathcal{S}$ in the calculation formula of P are almost impossible for CDN nodes to obtain. Therefore, SocialCache maintains scalability for different scenarios without introducing obvious risks of privacy exposure.

3. Experiment

In this section, we conduct a series of experiments to demonstrate the usefulness of SocialCache. We first introduce the datasets in §3.1. Afterward, in §3.2, we present the experiment's setup, including environment configuration, baseline methods, and evaluation metrics. Finally, the description and rigorous interpretation of the results will be illustrated in §3.3.

3.1. Datasets

For the scenario of self-operated CDNs of OSNs, a realistic and holistic dataset requires a social network graph and detailed information on each request. Unfortunately, such datasets could not be easily released to the public since they are highly related to OSN providers' confidential information and user privacy. As a result, many researchers in the field have to use synthetical datasets [3] [18]. Existing data generation methods are broadly idealistic and primarily based on randomness and theoretical mathematical distributions, lacking actual datasets as factual support. To better validate SocialCache, we use three real-world datasets:

TwitterSmall: We extract the largest ego network from a real Twitter user network dataset [25] as the social network, with 223 nodes and 4,481 directed edges. The following configurations are applied to both TwitterSmall and TwitterLarge datasets. The information carried out by request includes the user's ID, timestamp, request action, geolocation, and the media file size specific only to a POST request. We construct the total number of requests per user using the Zipf distribution (α=1.765, β=4.888) [26], defined as:

$$Zipf(x) = \beta x^{-\alpha} \tag{2}$$

The time interval between the requests obeys the LogNormal distribution (μ=1.789, σ=2.366) [26], given by:

$$LogNormal(x) = \frac{1}{\sqrt{2\pi}x\sigma} e^{-\frac{(\ln x-\mu)^2}{2\sigma^2}} \tag{3}$$

The ratio of VIEW actions to POST actions in all requests is 95:5 [18]. For geolocation information, we obtained the proportion of Twitter users in different countries in 2022[1], placing each user to the capital of the corresponding country. The media file sizes of all POST requests obey a trace-driven distribution, where we run the "wget" command in parallel to access the URLs of 26,952,281 media files from another Twitter dataset [27]. Finally, we sort all users' requests chronologically into a sequence, totaling 37,278.

TwitterLarge: We choose a larger Twitter dataset [25] to validate the reliability of our method on a large scale. The network has 11,088 nodes and 2,420,766 directed edges. The generation configurations and process of the requests are the same as that of the TwitterSmall dataset, which finally produces a sequence of requests of length 342,542.

Brightkite: We select the largest community as the OSN graph from an OSN dataset Brightkite [28] with detailed user login times and geographic locations, containing 5,773 nodes and 44,302 edges. The data ranges from January 1st, 2010, to June 1st, 2010. The media file sizes and access traces were generated according to the TwitterSmall configuration, totaling 749,558 requests.

3.2. Setup

3.2.1. Configuration. To emulate a realistic CDN environment, we utilize Mininet[2] as the network simulator, deploys on a host in CloudLab[3] with 40 CPU cores and 157GB memory, and runs Ubuntu 18.04 LTS. SocialCache is deployed based on Mininet, with a three-layer CDN hierarchy. Furthermore, we obtain a realistic network topology through real-world network measurements. We also fine-tune the parameters in Equation 1 using a hill-climbing algorithm [24].

Measurement: We select 29 Points of Presence (PoPs) from Google Cloud Platform, distributed worldwide on five continents, to build a realistic network topology for the CDN hierarchy discussed in §2.1.1. We deploy one cloud machine on each PoP for measuring network conditions, from May 15th to May 29th, 2022. For each pair of cloud machines, we measure the average results over one minute using Ping instruction at a time and a total of 10 times at different periods, using the average results as the latency and the loss rate. In addition, we use iPerf3 instruction to measure the network bandwidth between the two cloud machines during idle hours, and take the average result of 10 times as the set network bandwidth. The geographic location of CDN nodes placement refers to Artemis [29].

Parameter Selection: We adopt a heuristic algorithm named hill-climbing [24] to select the parameters $\mathcal{W_0}$, $\mathcal{W_1}$ and $\mathcal{W_2}$ of Equation 1. To demonstrate the scalability, we select the TwitterSmall dataset for parameter selection and apply the parameters to the TwitterLarge dataset.

3.2.2. Baselines. We select four representative caching strategies as baselines and compare SocialCache with them:

1) RAND. When a cache miss event happens, and the cache is full, a random $\mathcal{E}_{exist}$ will be replaced.
2) FIFO. The content comes into the cache in order, and the first $\mathcal{E}_{exist}$ is selected when replacing.
3) LRU. Contents are maintained according to the visit time, and replace the earliest visited $\mathcal{E}_{exist}$.
4) LRU-Social. LRU-Social [3] is an advanced baseline. It uses the spreading power of each user calculated by the Susceptible-Infected-Recovered (SIR) spreading model to extend the retention time of the contents posted by significant users.

3.2.3. Metrics. We use two typical metrics to evaluate the performance of our experiments:

1) Network traffic volume. Network traffic volume is defined by the sizes of all files transferred

1. https://www.statista.com/statistics/242606

2. http://mininet.org

3. https://www.cloudlab.us

TABLE 1. NETWORK TRAFFIC VOLUME (GB) ON DIFFERENT DATASETS AND CDN LAYERS

Method	TwitterSmall				TwitterLarge				Brightkite			
	L1 CDN	L2 CDN	L3 CDN	All	L1 CDN	L2 CDN	L3 CDN	All	L1 CDN	L2 CDN	L3 CDN	All
RAND	3.50	5.91	2.60	12.02	35.78	66.63	32.81	135.22	97.51	139.81	69.61	306.93
FIFO	3.01	4.13	1.31	8.45	33.98	61.50	29.48	124.96	71.55	73.92	29.68	175.15
LRU	2.94	3.38	0.63	6.95	34.65	61.12	28.43	124.22	70.17	72.55	29.68	172.41
LRU-Social	2.94	3.38	0.63	6.95	34.60	61.06	28.42	124.08	81.78	104.62	50.15	236.55
SocialCache	**2.47**	**2.90**	**0.61**	**5.99**	**33.31**	**57.09**	**25.74**	**116.14**	**67.61**	**69.99**	**29.68**	**167.28**

TABLE 2. NETWORK TRAFFIC VOLUME (GB) FOR DIFFERENT SOCIAL CONNECTIVITY METRICS ON DIFFERENT CDN LAYERS

Social Connectivity Metric	Network Traffic Volume (GB)			
	L1 CDN	L2 CDN	L3 CDN	All
In-degree	33.32	57.42	26.05	116.79
PageRank	33.44	57.64	26.16	117.24
Laplacian centrality	33.47	57.82	26.31	117.61
Betweenness centrality	33.43	57.78	26.31	117.52
Effective size	33.31	57.09	25.74	116.14
Standard deviation	0.07	0.27	0.21	0.54

through the CDN. Network traffic cost accounts for a sizable portion of total CDN spending. Reducing network traffic is our primary goal of SocialCache's optimization, which saves operational costs of about $0.085 per GB [14] and alleviates load pressure [13] for CDN service providers.

2) Cache hit ratio. Cache hit ratio is the percentage of successfully served requests in the cache. A higher cache hit ratio means the caching strategy performs better, due to users receiving results faster [3] and fewer requests being transferred through the CDN.

3.3. Results

3.3.1. Network Traffic Volume. We evaluate the performance of network traffic volume from perspectives including datasets, CDN layers, and social connectivity metrics.

Datasets: In Table 1, experiment results show that SocialCache outperforms four baselines for the network traffic volume metric on different datasets. Compared with the second-best results for each dataset, SocialCache considerably reduces network traffic volume by 13.81%, 6.40%, and 2.98%. Although LRU and LRU-Social are generally better than FIFO and RAND, they still have a large gap with SocialCache's optimization of network traffic volume. It shows that SocialCache has strong generalizability and remarkable optimization effects on different datasets.

CDN layers: SocialCache's advantage of network traffic volume savings on different CDN layers is also significant, displayed in Table 1. Taking the TwitterLarge dataset as an example, compared with the second-best method LRU-Social, SocialCache reduces 3.73%, 6.50%, and 9.43% for network traffic volume in L1, L2, and L3 CDN layers.

Social connectivity metrics: We compare the results of SocialCache with different social connectivity metrics illustrated in Table 2. The results show that the standard deviations of SocialCache using different social connectivity metrics is small, which means a similar impact on network traffic. SocialCache using effective size performs the best, with a visible advantage in network traffic volume reduction. This demonstrates the usefulness of the effective size that SocialCache chooses by default.

3.3.2. Cache Hit Ratios. We evaluate the performance of cache hit ratios for different caching strategies and datasets, displaying the results in Table 3. SocialCache outperforms four baselines on three datasets, maintaining high cache hit ratios. This is because SocialCache can provide superior cache replacement using factual social connectivity information and actual file size distribution. The conclusion of the cache hit ratio matches our design expectations. SocialCache prioritizes reducing network traffic volume while trying to maintain high cache hit ratios.

3.3.3. Operation Time. We define the total time of caching operations for all CDN nodes as the operation time, where less operation time means higher efficiency [13]. As shown in Table 3, the baselines RAND, FIFO, and LRU are relatively simple with low time complexity, hence they have relatively short operation times. LRU-Social takes extraordinary extra time than other methods because its inherent logic requires time-consuming enumeration for each request, which is not practical for large-scale use. Although SocialCache is not that straightforward, the lowest operation counts result from the highest cache hit ratio, enabling overall operation times equivalent to FIFO and LRU, with negligible overhead. This is because, in the specific implementation of SocialCache, we introduce a priority queue to accelerate the process of fetching the minimum $\mathcal{P}$ value in the current cache, which reduces the time complexity from O(C) to O($\log_2 C$), where C is the current cache size.

4. Conclusion and Future Work

In this paper, we propose SocialCache, a pervasive social-aware caching strategy for the emerging trend of using self-operated CDNs of OSNs. SocialCache leverages different types of social connectivity information for CDN optimization. It reduces network traffic volume while maintaining high cache hit ratios, further saving service costs and uplifting user experience quality. The design of SocialCache protects user privacy and maintains scalability. We implement SocialCache on the Mininet with a mainstreaming three-layer CDN hierarchy, real-world network measurements, and parameter selection using hill-climbing. The improvement of SocialCache is evident in three real-world OSN datasets and different scenarios. Experiments

TABLE 3. CACHE HIT RATIO AND OPERATION TIME ON DIFFERENT DATASETS

Method	Cache Hit Ratio (%)			Operation Time (s)		
	TwitterSmall	TwitterLarge	Brightkite	TwitterSmall	TwitterLarge	Brightkite
RAND	14.60	1.95	36.01	3.76	37.60	90.08
FIFO	33.77	7.34	71.23	3.30	36.79	67.38
LRU	48.40	9.03	71.77	3.40	38.71	73.02
LRU-Social	48.39	9.07	48.99	6.13	874.65	32463.93
SocialCache	48.46	10.36	71.81	3.46	48.38	69.53

show that SocialCache vastly reduces network traffic volume and maintains high cache hit ratios with low computational overhead. Experiments also demonstrate that SocialCache is a promising caching strategy for CDNs with pervasive and practical implications.

Our method represents clear advantages, and there still exists room for further forecasting improvement. A possible future direction is to explore the dynamic evolution of the social graph and incorporate more datasets to analyze under different scenarios. For parameter selection, advanced heuristic methods like Artificial Bee Colony [30] and recent machine learning algorithms such as Deep Reinforcement Learning [31], offer broad prospects for future work.

References

[1] L. Jin, Y. Chen, T. Wang *et al.*, "Understanding User Behavior in Online Social Networks: A Survey," *IEEE Communications Magazine*, vol. 51, no. 9, pp. 144–150, 2013.

[2] K. Wang, J. Zhang, G. Bai *et al.*, "It's Not Just the Site, It's the Contents: Intra-domain Fingerprinting Social Media Websites Through CDN Bursts," in *Proceedings of WWW*, 2021.

[3] A. Ghasemi and A. Ahmadi, "Cache Management in Content Delivery Networks Using the Metadata of Online Social Networks," *Computer Communications*, vol. 189, pp. 11–17, 2022.

[4] E. Nygren, R. K. Sitaraman, and J. Sun, "The Akamai network: a platform for high-performance internet applications," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 3, pp. 2–19, 2010.

[5] J. Yang, A. Sabnis, D. S. Berger *et al.*, "C2DN: How to Harness Erasure Codes at the Edge for Efficient Content Delivery," in *Proceedings of USENIX NSDI*, 2022.

[6] Z. Wang, W. Zhu, M. Chen *et al.*, "CPCDN: Content Delivery Powered by Context and User Intelligence," *IEEE Transactions on Multimedia*, vol. 17, no. 1, pp. 92–103, 2014.

[7] K. Akpinar and K. A. Hua, "PPNet: Privacy Protected CDN-ISP Collaboration for QoS-Aware Multi-CDN Adaptive Video Streaming," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 16, no. 2, pp. 1–23, 2020.

[8] F. Chen, K. Guo, J. Lin *et al.*, "Intra-cloud Lightning—Building CDNs in the Cloud," in *Proceedings of IEEE INFOCOM*, 2012.

[9] S. Scellato, C. Mascolo, M. Musolesi *et al.*, "Track Globally, Deliver Locally: Improving Content Delivery Networks by Tracking Geographic Social Cascades," in *Proceedings of WWW*, 2011.

[10] B. Fan, Y. Jiang, F. Zheng *et al.*, "Social-aware Cooperative Caching in Fog Radio Access Networks," in *Proceedings of IEEE ICC*, 2022.

[11] Y. Wang, X. Liu, D. Chu *et al.*, "EarlyBird: Mobile Prefetching of Social Network Feeds via Content Preference Mining and Usage Pattern Analysis," in *Proceedings of ACM MobiHoc*, 2015.

[12] C. Wu, X. Chen, W. Zhu *et al.*, "Socially-Driven Learning-Based Prefetching in Mobile Online Social Networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 4, pp. 2320–2333, 2017.

[13] K. Zhou, S. Sun, H. Wang *et al.*, "Improving Cache Performance for Large-Scale Photo Stores via Heuristic Prefetching Scheme," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 9, pp. 2033–2045, 2019.

[14] Z. Zheng, Y. Ma, Y. Liu *et al.*, "XLINK: QoE-Driven Multi-Path QUIC Transport in Large-scale Video Services," in *Proceedings of ACM SIGCOMM*, 2021.

[15] J. Li, Z. Li, R. Lu *et al.*, "LiveNet: A Low-Latency Video Transport Network for Large-Scale Live Streaming," in *Proceedings of ACM SIGCOMM*, 2022.

[16] H. Hu, Y. Wen, T.-S. Chua *et al.*, "Community based effective social video contents placement in cloud centric CDN network," in *Proceedings of IEEE ICME*, 2014.

[17] M. Zhou, T. Guo, Y. Chen *et al.*, "Polygon: A QUIC-Based CDN Server Selection System Supporting Multiple Resource Demands," in *Proceedings of ACM/IFIP Middleware, Industry Track*, 2021.

[18] C. Bernardini, T. Silverston, and O. Festor, "Socially-Aware Caching Strategy for Content Centric Networking," in *Proceedings of IFIP Networking*, 2014.

[19] Q. Gong, Y. Chen, X. He *et al.*, "Cross-Site Prediction on Social Influence for Cold-Start Users in Online Social Networks," *ACM Transactions on the Web*, vol. 15, no. 2, pp. 1–23, 2021.

[20] Z. Lin, Y. Zhang, Q. Gong *et al.*, "Structural Hole Theory in Social Network Analysis: A Review," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 3, pp. 724–739, 2022.

[21] X. Qi, E. Fuller, Q. Wu *et al.*, "Laplacian centrality: A new centrality measure for weighted network," *Information Sciences*, vol. 194, pp. 240–253, 2012.

[22] L. C. Freeman, "A Set of Measures of Centrality Based on Betweenness," *Sociometry*, pp. 35–41, 1977.

[23] R. S. Burt, "Structural Holes and Good Ideas," *American Journal of Sociology*, vol. 110, no. 2, pp. 349–399, 2004.

[24] J. Hill and K. Fu, "A learning control system using stochastic approximation for hill-climbing," in *Proceedings of IEEE JACC*, 1965.

[25] J. J. McAuley and J. Leskovec, "Learning to discover social circles in ego networks," in *Proceedings of NIPS*, 2012.

[26] F. Benevenuto, T. Rodrigues, M. Cha *et al.*, "Characterizing User Behavior in Online Social Networks," in *Proceedings of ACM IMC*, 2009.

[27] Q. Gong, J. Zhang, X. Wang *et al.*, "Identifying Structural Hole Spanners in Online Social Networks Using Machine Learning," in *Proceedings of ACM SIGCOMM, Posters and Demos*, 2019.

[28] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and Mobility: User Movement in Location-Based Social Networks," in *Proceedings of ACM KDD*, 2011.

[29] X. Li, Y. Chen, M. Zhou *et al.*, "Artemis: A Latency-Oriented Naming and Routing System," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4874–4890, 2022.

[30] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[31] A. Ben-Ameur, A. Araldo, and T. Chahed, "Cache Allocation in Multi-Tenant Edge Computing via online Reinforcement Learning," in *Proceedings of IEEE ICC*, 2022.