A Novel Video-on-Demand Caching Scheme using Hybrid Fuzzy Logic Least Frequency and Recently Used with Support Vector Machine

Sovannarith Heng[1], Phet Aimtongkham[2] and Chakchai So-In[3*]

[1]Applied Network Technology (ANT), Department of Computer Science, College of Computing, Khon Kaen University, Khon Kaen, 40002 Thailand. sovannarith@rupp.edu.kh
Orcid: https://orcid.org/0000-0002-8649-1079

[2]Applied Network Technology (ANT), Department of Computer Science, College of Computing, Khon Kaen University, Khon Kaen, 40002 Thailand. phetim@kku.ac.th
Orcid: https://orcid.org/0000-0001-5289-1149

[3*]Applied Network Technology (ANT), Department of Computer Science, College of Computing, Khon Kaen University, Khon Kaen, 40002 Thailand. chakso@kku.ac.th
Orcid: https://orcid.org/0000-0003-1026-191X

Received: December 04, 2022; Accepted: January 10, 2023; Published: March 30, 2023

Abstract

Considering the current era of mobile Internet, the prevalence of user access, such as accessing multimedia via high-speed Internet and, in particular, video contents, such as those from YouTube and Netflix, has become the norm. However, drastically increased video access can lead to several issues, e.g., long delays and low throughput, unless efficient Internet traffic management schemes are applied. One promising approach is based on caching efficacy built into the proxy server or content delivery networks. Several traditional caching policies derived from classic CPU caching are available; various policies provide simplicity gains but at a cost of low precision. Advances in storage technologies and CPU speedup have drawn attention to the latter aspect. This research thus focuses on a two-level caching scheme. The first level combines two classic caching models, Least Frequency Used (LFU) and Least Recently Used (LRU), to enhance the hit rate (HR) and byte hit rate (BHR) for real-time or (fast) online communication with small cache size constraints. Fuzzy logic (FL) is applied to derive a proper weight for this hybrid model. A Support Vector Machine (SVM) is also used for the second level to mainly focus on improving the replacement precision with more knowledge given the larger cache size. Some key features are selected, geographical distance and similarity hashing in particular; in addition, a proper period of training knowledge is selected for the SVM based on human behavior. The proposed scheme is evaluated and compared to two well-known caching schemes, LRU and LFU, in addition to other state-of-the-art intelligent hybrid caching models, e.g., SVM-LRU and SVM-LFU, using a well-known dataset from IRCache. The comparison is performed in terms of performance gain; the proposal has the highest HRs and BHRs, i.e., higher HRs and BHRs than LRU, LFU, SVM-LRU, and SVM-LFU.

Keywords: Fuzzy Logic, Least Frequency Used, Least Recently Used, Hybrid Model, Cache Replacement, Soft Computing, Support Vector Machine, Video Caching.

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), volume: 14, number: 1 (March), pp. 15-36 DOI: *10.58346/JOWUA.2023.I1.002*

*Corresponding author: Applied Network Technology (ANT), Department of Computer Science, College of Computing, Khon Kaen University, Khon Kaen, 40002 Thailand.

1 Introduction

Internet technology, a means of communication in human society, can be considered as one of the key factors affecting our daily life, alongside food, clothing, medicine, and shelter. Based on a report provided by the ITU, from 2005 to 2022, more than 5 billion users have become involved in the Internet community (ITU Measuring, 2022). This rapid increase has led to an increase in Internet traffic demand to 1,200 Terabit per second (ITU International, 2022); in particular, video traffic is one of the most bandwidth-intensive applications due to its unique characteristics: huge file sizes compared to the other formats, such as text, image, and audio (Ramadan, E., 2021).

Based on statistics from Cisco Systems, in 2022 (Cisco System, 2022), video content traffic over the Internet has represented more than 300 Exabytes per Month, representing almost 82 percent of all IP traffic. Nielsen Co. LTD. (Nielsen Video on Demand, 2019) has approximated the ratio of video usage - up to 67% due to the growth in video on-demand (VoD). Ofcom has also found that 58% of the UK population accesses video over the Internet (Of com the communications market, 2016).

Many types of video access modes are available, e.g., broadcasting and multicasting; however, VoD, such as that from YouTube, Netflix, and Amazon Prime Video, is commonly used by Internet users (Raghavendra, R., 2010). VoD is typically unicast; in other words, users or subscribers can select and watch video content based on their desires. Generally, VoD can be accessed from various devices and services, such as on a website, set-top box, mobile phone, or gaming console, normally using Internet Protocol Television (IPTV) technology. The Subscription VoD provider (SVoD) uses a client/server network architecture model; here, the SVoD can control the amount of data transferred and the rights of each user to access a particular video (Podlipnig, S., 2003).

To provide effective video transmission to achieve high quality of service, there are two factors that represent key considerations (Mittal, S., 2016). First, the efficiency of data transmission (bandwidth) is important because the bandwidth increases proportional to the number of concurrent subscribers in a time interval, in addition to coding schemes providing various bitrates. Second, the timing delay suffered when transferring the video from the VoD server to the destination is important in providing the best user experience, in addition to delay jitter, to ensure user satisfaction. The delay is typically caused by several factors, including distance, transmission protocols, and routing paths.

Although SVoDs have sought ways to increase the bandwidth to meet the growth in concurrent subscribers and most likely reduce their waiting times, the current Internet infrastructure may not be able to keep pace with this rapid growth, most likely due to the costs/benefits and standards, including various regulations (Ali, W., 2011). Thus, both academia and industry have attempted to determine ways of mitigating this concern. One promising approach considers the implementation of video caching or proxy, typically built up on a web proxy.

The key concept underlying caching/proxy is similar to that of a traditional caching scheme in hierarchical CPU memory, including CPU caches, RAM, and hard disks (Jang, H., 2014). The inter-memory levels are positioned such that the time required to access data given the speed and distance is reduced. The efficacy of this structure can also be increased via accessible data repetition based on user behavior. This perspective is also applied to cache web objects (in addition to video contents) to reduce the access time for long-distance Internet communication, including bandwidth optimization with repetitive access patterns.

In general, a traditional caching replacement policy (TCRP) simply applies a single factor to determine the object for replacement, e.g., frequency, time of arrival, or size, with a key advantage being

its response time due to its low complexity, which in turn makes TCRPs suitable for real-time/online communications (Ali, W., 2011). However, this simplified model also comes with a trade-off: replacement precision. This will most likely lead to un-optimized caching storage usage and thus a high cost of hardware storage.

Soft Computing (SC), with its key advantages of high precision with comparable computational complexity trade-offs to heuristic approaches, has recently been proposed to resolve the problem of uncertainty in addition to non-linear solvers generally used in science and engineering domains (Bhaumik, H., 2016) (Wang, C., 2016) (Baskaran, K.R., 2016) (So-In, C.,2016), such as clustering and classification, including caching replacement (Sathiyamoorthi, V., 2014) (Albana, A.A., 2015) (Aimtongkham, P., 2016). Several classes of SC, including Neural Networks (NNs); Fuzzy Logic (FL); Evolutionary Computation (EC), e.g., Genetic Algorithms (GAs); and Support Vector Machines (SVMs), can be applied to a particular problem based on the problem characteristics.

Among these classes, one promising approach is SVMs applied to both classification and clustering based on their key advantage of fast training and high precision (Kjernsmo, K., 2015). FL is also generally used in uncertainty scenarios based on constructed rules. Thus, this research seeks the possibility of combining TCRP and SC as a two-level caching scheme for VoD. FL was first used to determine a proper weight as a selection criterion between Least Frequency Used (LFU) and Least Recently Used (LRU) for fast caching (online) but with a small cache size (Andjarwirawan, J., 2015). To improve the replacement precision with more knowledge, SVM was then applied with a larger cache, including proper feature selections, i.e., geographical distance and hashing, to determine object similarity; these schemes are called Hybrid FL LRU/LFU and SVM Caching Schemes (HF-SVM).

The remainder of this article is organized as follows. Section 2 provides a brief survey of caching schemes, in particular, applications in the video caching domain as well as the integration of SC techniques. Then, in section 3, the detailed technique of our proposal, HF-SVM, is explained. To confirm our performance and practicality, section 4 reports on and discusses the comparative performance of HF-SVM against other techniques, including traditional caching schemes, i.e., LRU and LFU, and their hybrids with SVMs. Finally, section 5 presents our conclusions and possible future work.

2 Related Work

For years, the problem of caching was addressed for data transfer between CPU and memory. The task of caching consists of buffering the data for repetitive access to reduce the access time but considering the limitations of the cache size (high cost). Several well-known TCRPs are available, such as LRU, LFU, First in First Out (FIFO), and Size, corresponding to ageing, access frequency, time of arrival, and object size (Mittal, S., 2016).

During the internet era, the caching concept (CPU-memory) was also adopted for Internet users to reduce Internet congestion, typically built in web caching/proxies such as Squid and Varnish; some techniques included the use of LRU and LFU (Andjarwirawan, J., 2015). Considering TCRP limitations, i.e., low replacement precision, SC has one again been proposed to address this constraint. For example, in 2008, J. Cobb and H. ElAarag (Cobb, J., 2008) applied a NN for web caching based on a CPU caching scheme. This algorithm selected several caching parameters, such as URL, frequency, object size, and time stamp, in the training stage, resulting in a high replacement accuracy but at the cost of a very high computational time complexity.

In addition, M.C. Calzarossa and G. Valli proposed another SC, i.e., FL for web caching (Calzarossa, M.C., 2003). The key advantage of FL is that there is no requirement on training and uncertainty support for determining a particular weight for various variables. Fuzzy Mamdani was used, given parameters, i.e., frequency, access time, and object size, to create sixteen fuzzy rules. A triangular membership function was selected as the decision function. Note that FL can boost the replacement precision, especially for small caches but not for all situations, i.e., large cache sizes.

One of the SC techniques, EC, has also been evaluated as a cache replacement scheme. For example, A. Vakali applied a GA to determine the density of web objects, including the retrieval rate (the product of latency and bandwidth) and then used the density to construct the replacement rules (Vakali, A., 2002). The evaluation demonstrated a higher hit rate (HR) and a higher byte hit rate (BHR) compared to LRU; however, due to its random nature, the precision result was not certain.

Considering another SC technique, on the other hand, SVMs were originally used for classification; such a technique cannot be properly used as a direct caching replacement. However, in 2016, P. Aimtongkham et al. integrated an SVM into LFU. After training, whether new objects are to be cached is determined based on the given SVM training models; if the objects are to be cached, LFU will be applied to perform the actual placement. Based on the evaluation, this model improved the HR and BHR compared with traditional LFU and LRU (Aimtongkham, P., 2016).

All previous approaches considered only a single cache, which most likely produces the main drawback on caching operation intervention. Thus, some concepts of two-level caching have been introduced. For example, W. Ali and S. M. Shamsuddin proposed online and offline caching schemes (Ali, W., 2009). LRU was used as the former type of scheme to achieve a fast replacement scheme; in contrast, for the latter type of scheme, an Artificial Neuro-Fuzzy System (ANFIS) was applied due to its key advantage of its high replacement accuracy.

Additionally, the same set the authors in (Ali, W., 2012) enhanced the ANFIS with an SVM for cache replacement to improve the caching precision with comparatively fast training. The Radial Basis Function (RBF) was selected as the kernel function. Five main object types were used for training: ageing, frequency, time stamp, object size, and object type. The experimental results showed that the SVM with LRU (SVM-LRU) can improve the precision of caching replacement compared with LRU and LFU.

In addition to LRU, an SVM has also been hybridized with LFU (SVM-LFU); for example, V. Sathiyamoorthi and P. Ramya instead considered LFU using four features: recency or ageing, frequency, object size, and duration (time to acquire object from the original server) (Sathiyamoorthi, V., 2014). The evaluation was only on one dataset from IRCache, with 70% of the dataset used for training and 30% used for testing, and the results demonstrated a superior performance over LFU.

Specifically considering VoD, few proposals for optimized video caching have been presented. For example, in 2016, H. Zhao et al. evaluated various TCRPs, including LFU, LRU, and FIFO, for video objects and reported that, although LFU outperformed LRU for small cache sizes, in general, there is no significant difference in terms of the HR as the key metric without consideration of the BHR (Zhao, H., 2016).

TCRPs have also been used as a hybrid model for video caching. For example, F. Olmos et al. proposed a dynamic model to cluster video files given the video content using Che approximation with LRU (Olmos, F., 2014). Different classes of video, such as movies and news, are provided with different

weights for caching storage. The experimental results showed that this scheme can increase the HR to 11% on average.

Based on the key characteristics of human nature in terms of accessing video content, J. Chakareski applied graph theory concepts based on community-driven access patterns for social media video. The primary purpose of that research was to reduce the external bandwidth without considering the HR or the BHR (Chakareski, J., 2014). Similarly, M. Claeys et al. investigated messaging client behavior for VoD sessions (Claeys, M., 2016). The results of the analysis demonstrated caching improvements of over 20% for the HR but did not consider the BHR.

Note that, from the above discussion, only a few studies have focused on video caching improvements, some of which only considering user behavior (Chakareski, J., 2014) (Claeys, M., 2016). Their focuses were mostly on the HR and not the BHR, which impacts the caching performance in terms of storage cost. Some researchers have only investigated TCRPs (Zhao, H., 2016), which again are fast but suffer from the key limitation on the caching precision.

3 Hybrid VoD Caching Schemes using Fuzzy Logic LFU and LRU with Support Vector Machine (HF-SVM)

TCRP is appropriate for fast response, and SC is appropriate for high replacement precision; thus, a hybrid mode will be described in this section. Fig. 1 shows that there are two caching levels: Fuzzy TCRP and Optimized SVM. In general, users/subscribers make a video object request from the Internet to the TCRP Cache Engine. This engine will evaluate the request in the 1st-level cache - if found, the content is returned to the users; otherwise, the request is fired back to the 2nd-level cache (via the SVM Cache Engine). Here, if the content is found, the content is returned back to the users; in addition, the 1st-level cache is updated.

In contrast, if there is a cache miss, the TCRP Cache Engine will request the video object from the source (VoD servers) and then, in addition to forwarding back to the users, will update the 1st-level cache if there is some free space remaining. However, if the cache is full, the fuzzy logic component will be fired to determine an appropriate weight to select the replacement policy between LFU and LRU (Fuzzy Cache LFU/LRU); then, the object is replaced in the 1st-level cache. In addition, the replaced object will be forwarded to the 2nd-level cache to update the larger caching storage. SVM Cache Replacement will determine if this object should be cached (SVM Cache Management) based on historical information.

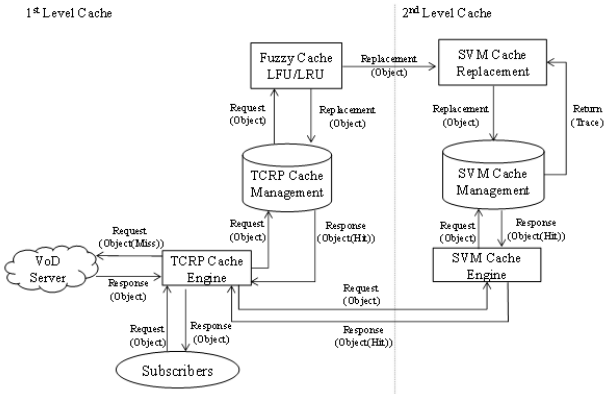


Figure 1: System Overview of the Two-level Intelligent Web Caching Schemes

3.1. First-Level Caching

There are three main components in this level: the TCRP Cache Engine, the TCRP Cache Management, and the Fuzzy Cache TCRP (LFU/LRU).

TCRP Cache Engine: This component is primarily used to interact with the subscribers (request/response). This engine will retrieve the video object from the 1st-level caching storage (TCRP Cache Management) and forward it to the users if hit (Cache Hit). In the case of a miss (Cache Miss), this engine will also forward the request to the 2nd-level cache via the SVM Cache Engine. If hit, the retrieved video object will be sent back to the engine to update the 1st-level cache (cache synchronization) (via TCRP Cache Management) in addition to the object being forwarded to the users. In the case of a cache miss (for both cache levels), this engine will perform the actual object retrieval from the Internet, i.e., from the VoD servers; forward the fresh video object back to the users; and update the 1st-level cache (via TCRP Cache Management).

TCRP Cache Management: This component functions as both caching storage (store the object if there is free space) and management with the actual replacement operation. Again, the module returns the actual video object to the previous component if hit. This module also keeps track of the frequency and ageing for the LFU and LRU replacement policies (interacting with the Fuzzy Cache TCRP to make the final replacement decision). In addition, this management will forward the replaced object to the 2nd-level cache via SVM Cache Replacement.

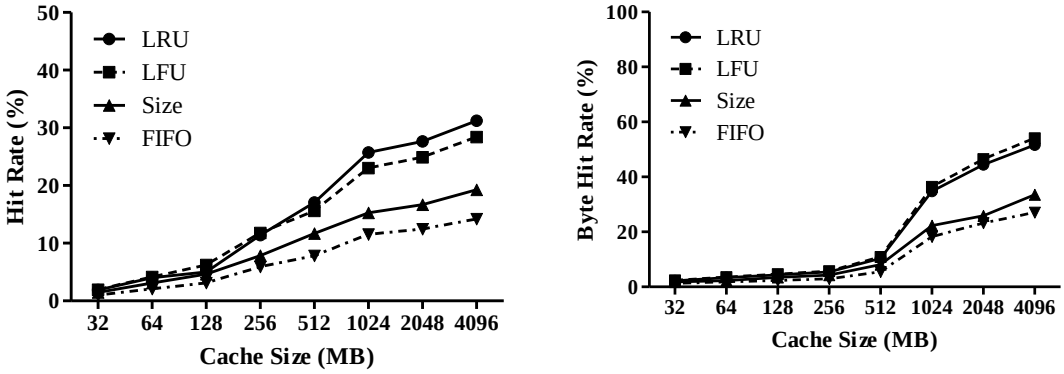
Fuzzy Cache TCRP (LFU/LRU): In case there is no free space remaining in the caching storage (1st level), this module is used to determine the actual caching replacement policy resulting in either LFU or LRU based on the fuzzy system criterion (discussed in detail later).

3.1.1. LFU and LRU Caching

Although there are several classic caching schemes, this research only considers four well-known caching policies: FIFO, SIZE, LRU, and LFU. Other approaches are also applicable; however, these approaches are the simplest, have the lowest computational complexity, and are suitable for real-time or online communications (Kjernsmo, K., 2015). FIFO was one of the first simplified caching policies and uses an arrival time as a criterion to replace new objects. Given other considerations, especially different file type characteristics, SIZE is also an important replacement scheme based on the object size for allocating space for new objects, e.g., replacing objects with the largest or smallest file size.

Considering user behaviors, LRU and LFU are both promising techniques for the repetition of access patterns. LRU considers the time to access a particular object in the final replacement decision. In other words, recent objects will be frequently used. For the other aspect, LFU assumes that a larger number of accessible objects means a higher probability of being cached. In other words, low-frequency objects will be selected for the final replacement.

Our inspiration for choosing LRU or LFU is based on our intensive evaluation of video caching, as in the examples shown in Figure 2. Note that the trace (access pattern) was retrieved from a well-known dataset, NLANR (Boulder, CO, USA - BO2), with a 30-day duration. These figures show the HR and BHR for four different policies while varying the cache size, i.e., from 32 to 4096 MB. The results show that LRU outperforms the other criteria, especially with large cache sizes in terms of the HR, with the performances being in the order of LFU, Size, and FIFO. However, the performance of LFU is best in terms of high precision based on the BHR, where the performance is in the order of LRU, Size, and FIFO.



(a) % Hit Rate (b) % Byte Hit Rate

Figure 2: Video Caching Performance Using BO2 Dataset

3.1.2. Fuzzy LFU and LRU Caching

A Fuzzy Logic System (FLS) is an SC technique that provides the logical reasoning when there is an approximation as opposed to an exact solution. Normally, the minimum and maximum are defined as 0 and 1; however, the actual value will fall in between with some amount of fuzziness. There are four main stages corresponding to FL Mamdani as follows (Sabeghi, M., 2006). Note that another set of evaluations was also performed using FL Sugeno, but a lower performance was found.

Fuzzifier: This step is used to transform the input data (here, the video object characteristics), i.e., frequency (object access rate), duration (latency required to access the video object from the VoD server), and size (actual video object size). This operation will normalize these features into numerical formats and then map to the defined membership functions to determine the crossing points of each function (corresponding to fuzzy rules). In this research, a triangular function was selected in addition to other functions, e.g., Gaussian, trapezoidal, generalized bell, and sigmoid functions (Bartere, M.M., 2012), as shown in Figures 3 to 5. Note that another experiment was also performed using these functions; however, the triangular function was found to be superior.

Fuzzy rule: This step is used to generate the mapping between the input and output given its membership function (if/then). In this scheme, the rule construction procedure follows Table 1. Here, there are twelve rules, including Very High (VHI), High (HIG), Medium (MED), Low (LOW), and Very Low (VLO) (Klir, G., 1995).

Fuzzy Inference Engine: This process is used to derive the output (weight) given the inputs, i.e., duration, size, and frequency; and rule-based systems (see Figure 6) and to apply an aggregation method, i.e., the intersection operation. The output here is the caching probability (see also Figure 7).

Defuzzifier: This step is used to compute the final output based on the Center of Gravity (CoG) concept over the derived output (weight) (see Figure 7). This output (ranging between 0 and 1) will be provided the bound to make it either 0 or 1, corresponding to the final replacement policy, i.e., either LFU or LRU, respectively. Note that in this research, based on our intensive evaluation, all fuzzy parameters will be normalized in a range, i.e., frequency (between 0 and 30), duration (between 0 and 1,000 milliseconds) based on the recommendation provided by (Wust, C.C., 2004), and object size (between 0 and 4.295×109 or 4 GB).

Table 1: Examples of Fuzzy Rules

If Duration	And Size	And Frequency	Then Stage
VHI	MED	LOW	VHI
HIG	HIG	LOW	VHI
VHI	HIG	MED	VHI
VHI	HIG	LOW	VHI
HIG	HIG	LOW	HIG
HIG	LOW	MED	MED
VHI	MED	MED	HIG
HIG	HIG	MED	HIG
VHI	HIG	HIG	LOW
HIG	HIG	HIG	LOW
MED	HIG	LOW	HIG
HIG	MED	MED	MED

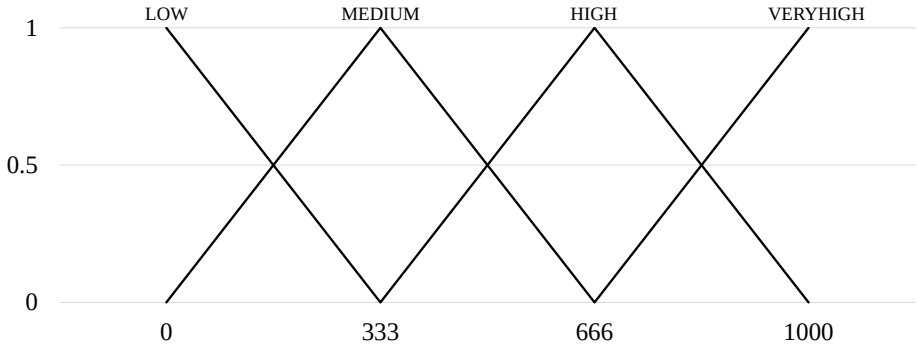


Figure 3: A Fuzzy Membership Function (Duration)

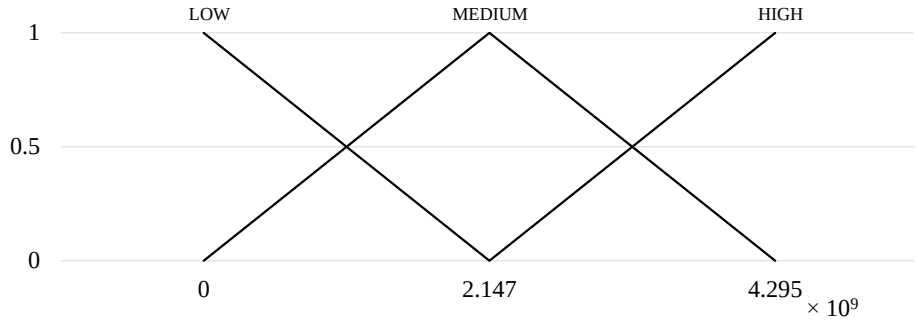


Figure 4: A Fuzzy Membership Function (Size)

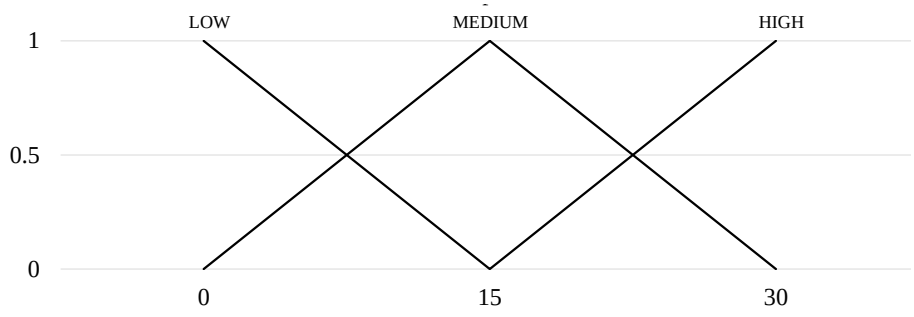


Figure 5: A Fuzzy Membership Function (Frequency)

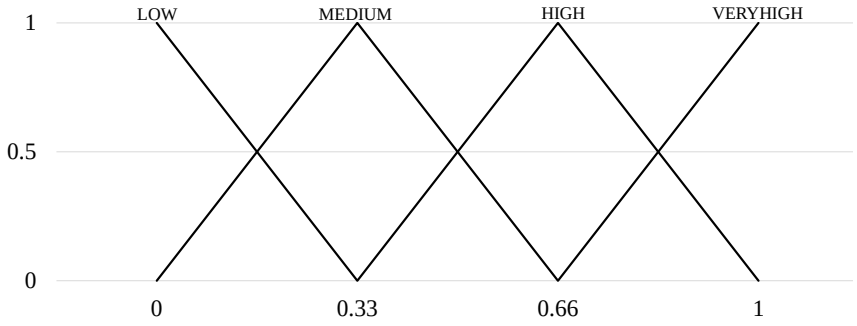


Figure 6: A Triangular Function of the Output Stage (Stage)

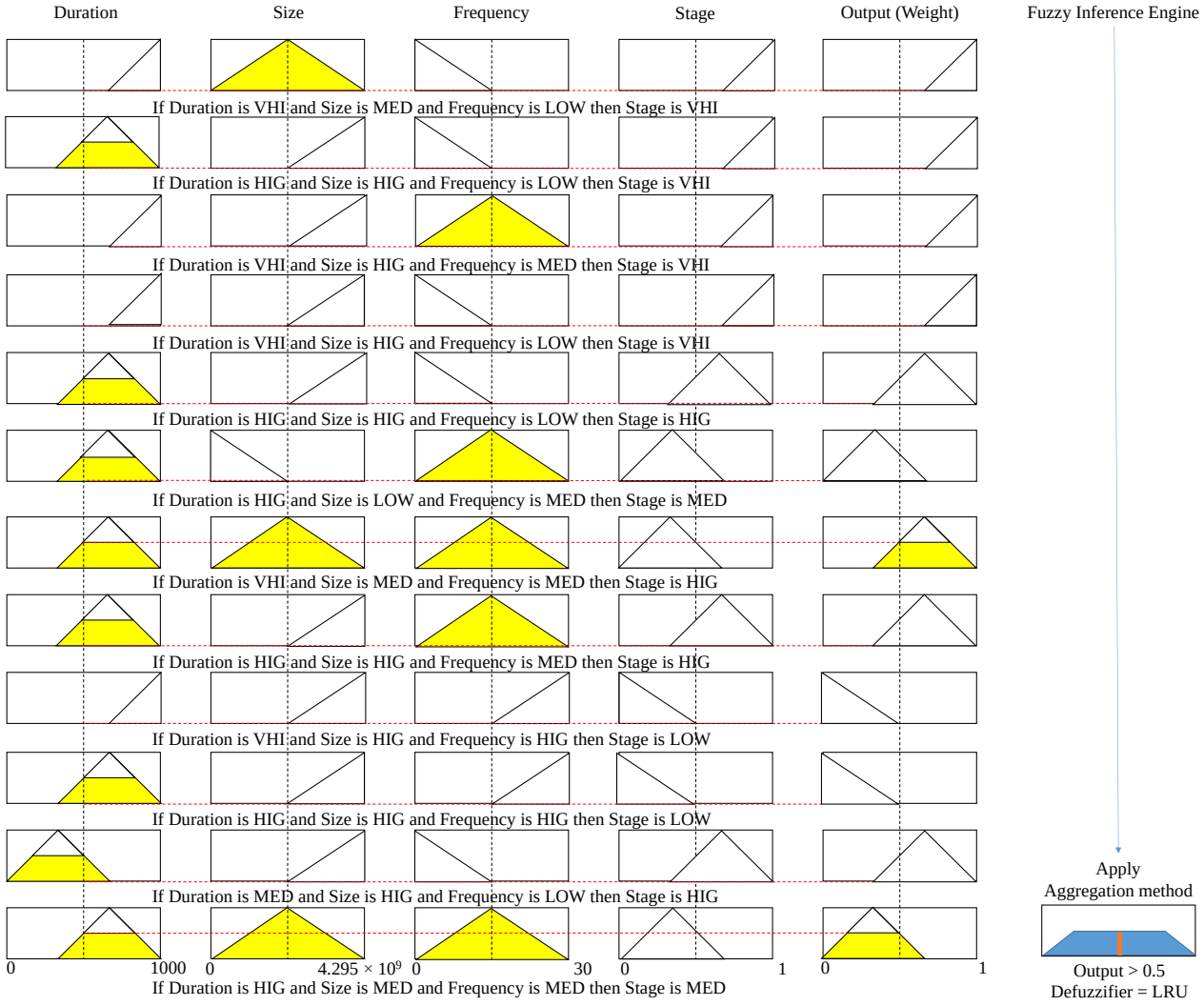


Figure 7: Defuzzifier: Example

3.2. Second-Level Caching

There are three main components of this level as follows: the SVM Cache Engine, the SVM Cache Management, and the SVM Cache Replacement.

SVM Cache Engine: This component works with the TCRP Cache Engine; if a cache miss occurs, this component primarily retrieves the video object from SVM Cache Management (2nd-level caching storage) back to the 1st-level cache (for cache synchronization). This module also returns "miss" if no such object is available.

SVM Cache Management: The primary function of this module is as database management - storing, replacing, and retrieving. Given a request from the previous component, the actual video object will be delivered. In addition, this module will store the video object determined by SVM Cache Replacement in a sequential manner. Note that this module also keeps the collection of access patterns via video traces to facilitate the SVM training process based on (weekly) time periods such as 7, 14, or 21 days (discussed in the next section).

SVM Cache Replacement: This module is primarily used to determine the final replacement decision of the object sent via Fuzzy Cache TCRP (LFU/LRU). There are two main tasks for this component. First, if there is free space in the 2nd-level caching storage, the object will be stored; otherwise, it will make a replacement decision based on the second task: SVM classification. This classification consists of two phases: training and testing.

The training process acquires knowledge based on the caching history (video traces from the storage management) to construct an SVM training model, which will be used for a later testing phase to determine whether content should be cached (determined by Replacement Probability - RP), i.e., 1 = cache and 0 otherwise. Noting that only objects with RP=1 will be used for replacement, we will also collect the access pattern in both stages (both 1 and 0) in the trace, which will be properly used during the training process.

One of the key classification models is via SVM, whose precision also depends on the training datasets. Since the nature of user access patterns for video objects is periodically updated, this model also requires an update of the training knowledge. Here, the training period is based on the round sequential of the caching storage. Again, especially at the beginning, if all objects in the storage have already been replaced, (re)training will be performed. However, the actual period used to determine the training cycle is based on a weekly duration corresponding to the user access behavior.

3.2.1. SVM for Video Caching

Considering supervised learning methods, the SVM represents a promising classification technique. In general, the SVM will determine two classes, making the SVM a non-probabilistic binary linear classifier (either -1 or +1) (Kumar, P.V., 2014). In this research, each class represents the caching stage, determining whether to cache (to replace/ignore). As shown in Figures 8 and 9, the SVM represents a set of points in space mapped to separate categories based on the highest margin. In these figures, linear classification is typically applied given the transformation procedure, i.e., from an *N*-dimensional input vector *X* to feature vectors that provide the objective function *D(X)*, as stated in Equation (1) below.

$$D(X) = W \cdot X + W0 = sign\left\{\sum_{j=1}^{m} w_j\, g(x) + b\right\} \quad (1)$$

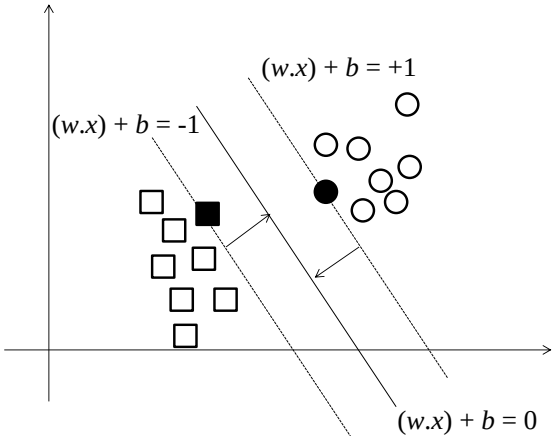


Figure 8: SVM Linear Classification

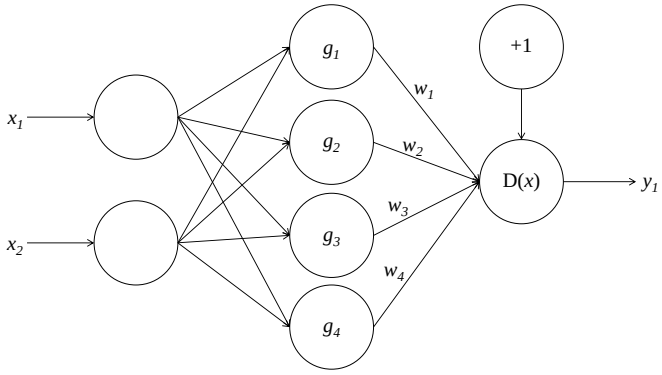


Figure 9: SVM Structure

where the input $X = \{x_1, \ldots, x_N\}$ will be fed into the model. Examples of such inputs are the HTTP request method (R), URL_ID (U), size (S), duration (D), geographical distance (G), and similarity (SIM) (discussed in the next section). A set of weights $W = \{w_1, \ldots, w_m\}$ are provided such that N is the total number of input nodes, m is the number of kernel function nodes, and b is the bias. Lagrange multipliers will be applied in terms of α (a scaling factor that is the gradient of the function used to find the largest or smallest value) to derive the weight. y_i is the target value in the range of [-1, +1] (either should be cached or else), and $g(x)$ is an SVM kernel function, e.g., linear, polynomial, RBF, or sigmoid.

Algorithm 1: SVM Training (VoD Caching)

Input: $X(R, U, S, D, G, SIM)_N$, Y_N

Output: w_m, b

1 Apply Kernel Function $g(X) = tanh(\gamma X^T X[j] + r)$

2 Derive Weights (w) and Bias (b) based on Lagrange Multipliers

Algorithm 2: SVM Testing (VoD Caching)

Input: $x(R, U, S, D, G, SIM)$, w_m, b

Output: RP

1 Apply Kernel Function $g(x) = tanh(\gamma x^T x[j] + r)$

2 Compute $RP \leftarrow sign\{\sum_{j=1}^{m} w_j\, g(x) + b\}$

Algorithms 1 and 2 show the SVM classification in both the training and testing stages. For training, six main features (R, U, S, D, G, and SIM) were selected as inputs from the access pattern to construct the training model. Note that the last two features will be discussed in the next section. In Algorithm 1, line 1 shows the determination from the inputs when applying a specific kernel function; here, this function is a sigmoidal function. In addition, gamma (γ) is $\frac{1}{2\sigma^2}$, where σ is a free adjustable parameter; based on the implementation of the SVM (from LIBSVM), this parameter was configured to 1 (Chang, C.C., 2011).

The SVM training process generates a result in terms of the weight (w) and bias (b) in the form of Lagrange multipliers. Similar steps were also applied for the testing shown in Algorithm 2. Only the test's individual input (x) (here, we only feed one line of input) will be applied to the kernel function over the training sets (X). Line 2 shows the application subsequently using Equation (1) to compute the caching replacement probability (RP), as shown in Algorithm 2.

3.2.2. SVM Classification Feature for VoD Caching

In general, the precision classification of SC techniques primarily depends on the feature characteristics; here, these characteristics can be selected from some variables related to metadata within the video access pattern (trace). Six main features were selected as follows. Note that each feature will be transformed into numerical formats.

First, the HTTP Request method (R) is the status of the HTTP for making a web request, i.e., GET (web request or pulling the object), POST and PUT (pushing the object), and DELETE (deleting the object). Here, the numerical transformations are 1, 2, 3, and 4, respectively. Note that in this research, the only object in the evaluation is a video.

Second, URL_ID (U) is the actual web/server address. The numerical assignment will start from 1 (first object) and be incremented by 1 for the remaining assignments.

Third, Size (S) is the actual size of the video object, ranging between 1 and 4 GB (or the maximum cache size).

Fourth, Duration (D) is the delay latency used to access the video object from the VoD server in milliseconds (ms). Note that if this value is high, the network distance (or the available bandwidth at a particular time) is also high. Therefore, the user satisfaction may not be sufficient, and they may stop the query or even change their interests.

Fifth, Geographical Distance (G) is the approximate physical distance from the requester (here, the VoD caching server) to the original VoD server. The lookup is performed from the source to destination locations (country) using IP Geolocation from GeoIP (GeoLite) (GeoLite legacy databases, 2016)with Geo Dist (Mayer, T., 2011). Equations 2 to 6 show examples of distance derivations. Not that the rationale behind using this G metric is based on the nature of the fluctuating bandwidth consumption during the working period, i.e., day or night (also fluctuating response time but with a fixed geographical distance).

$$dlon = lon_2 - lon_1 \quad (2)$$

$$dlat = lat_2 - lat_1 \quad (3)$$

$$a = \left(\sin\frac{dlat}{2}\right)^2 + \cos lat_1 \times \cos lat_2 \times \left(\sin\frac{dlon}{2}\right)^2 \quad (4)$$

$$c = 2 \times \text{atan2}\left(\sqrt{a}, \sqrt{1-a}\right) \quad (5)$$

$$d = R \times c \quad (6)$$

Where lat_1 and lon_1 are the latitude and longitude of the source country and lat_2 and lon_2 are those of the destination. R is the mean radius of the Earth (approximately 6,371 km). Again, this distance is an implication of a physical characteristic for a particular video object.

Finally, Similarity (SIM) is the sixth feature used to improve the classification precision. The motivation here is that, based on the video characteristics, it is difficult to determine the object similarity between the stored file with the computational complexity constraint. Therefore, to avoid byte-to-byte comparisons between video objects, MD5 hashing was selected to achieve high-precision object similarity determination. Note that this research is not limited to MD5; other hashing techniques can also be applied, including SHA, given the characteristics, including hash size and time complexity trade-off.

Pioneered by R. Rivest, MD5 is a promising hashing technique (The MD5 Message-Digest Algorithm, 2020) that is mostly used to correctly confirm file transfers over the Internet, similar to that used in Peer-to-Peer networks (Nurminen, J.K., 2013). The main advantage of MD5 is that it can quickly generate 128-bit hash values. Here, the results of MD5 were used to identify video object similarity in the cache because the video object size is large. To simplify the object comparison, the summation of MD5 chunks was used and limited to only the first megabyte of each object.

3.2.3. Caching Replacement with Periodic Training

Based on our observations, the VoD access pattern follows human behavior; in general, users/subscribers have watched a video in parts or via other media sources, such as at various events that are broadcast worldwide, e.g., Olympic broadcasts or presidential speeches. Here, user behavior is likely to exhibit similar characteristics on a weekly timeframe, i.e., watching video content such as cartoons, a series, or TV programs broadcasting an episode. Users also tend to watch videos of a previous episode beforehand to become familiar with the content before watching a current episode, similar to users also watching video/movie trailers beforehand.

Thus, this research introduced the concept of sliding windows to determine a scope of knowledge for training. In other words, a specific set of access patterns will periodically be used to determine training information to be used to achieve higher replacement probability precision given the constraints on caching storage size. As an example, shown in Figure 10, the selection of a 14-day period was used for the training model collected from a recent access, ignoring other accesses (old trace) (see the evaluation section for verification of this hypothesis).

Figure 10: Time Line Training Period

4 Performance Evaluation

To confirm the practicality of our approach, this section presents our evaluation, including a discussion of our hybrid model's performance, HF-SVM, in addition to a performance comparison against other existing candidate methods, such as SVM-LFU (Sathiyamoorthi, V., 2014) and SVM-LRU (Ali, W., 2012), as well as the other two traditional caching schemes, LFU and LRU (Andjarwirawan, J., 2015).

4.1. Data Pre-processing

To reflect the real usage of user access, IRCache (NLANR IRCache Sanitized Access Log, 2016), a well-known proxy log from the National Lab of Applied Network Research (NLANR) that includes the access trace from five main proxy servers across the U.S. (i.e., UC, BO2, SD, SV, and NY), was used for the web caching performance evaluation, in particular, the video object trace.

Note that most related evaluations of caching aspects have used these traces (NLANR IRCache Sanitized Access Log, 2016); among others, the key limitation of the proxy model is the deployment, which may affect commercial operation (Configuring the secure Gateway, 2016). In addition, since the actual access may include billions of objects, in this study, the trace was limited to 30 days from the first quarter of 2016 filtered by media type (MIME type) to only include video. Table 2 presents details about the dataset from IRCache.

Table 2: VoD Caching Dataset (from web trace IRCache)

Proxy Dataset	Proxy Server	Location	#Records	Duration (days)
UC	uc.us.ircache.net	Urbana-Champaign, IL	880,549	30
BO2	bo.us.ircache.net	Boulder, CO	859,703	30
SD	sd.us.ircache.net	Silicon Valley, CA	877,350	30
SV	sv.us.ircache.net	San Diego, CA	680,765	30
NY	ny.us.ircache.net	New York, NY	904,584	30

The trace files from IRCache record all requested information used to make a caching decision. The files include seven main attributes: the timestamp (with the socket status as closed) in milliseconds, the client address (IP address of the requester to the proxy server), the tag and HTTP code (the status of accessible codes, i.e., hit or miss), the size of the web object in bytes, the request method (e.g., GET, POST, or PUT), the URL, and the MIME type (e.g., html, audio, and video; here, the focus is only on video).

4.2. Performance Measurement Metric

There are two main metrics used in this research: Classification Precision and Cache Replacement Rate.

Classification Precision: As a unit for testing, the Corrected Classification Rate (CCR) (Jang, H., 2014) was selected to measure the classification accuracy, i.e., whether a particular video object should be cached. The metric will only apply to the intelligent system, aka SVM, for training/testing processes, as stated in Equation (10):

$$CCR = \frac{TP + TN}{TP + FP + FN + TN}(\%) \quad (10)$$

where True Positive (TP) is a classification result where positive training data (to be cached) are evaluated as positive (cached), True Negative (TN) is a classification result where negative training data (not to be cached) are evaluated as negative (not cached), False Positive (FP) is a classification result where negative training data are evaluated as positive, and False Negative (FN) is the classification result where positive training data are evaluated as negative.

Cache Replacement Rate: For system/overall testing, the HR and the BHR are commonly used to determine the overall caching efficiency. The HR is the ratio of the number of web objects that the proxy server can deliver directly back to the client (#cache hits/#requests), and its corresponding size is given by the BHR (#bytes of cache hits/#bytes requested) (Podlipnig, S., 2003).

4.3. Simulation Setup and Configuration

There are two main scenarios used to demonstrate the performance of our hybrid model corresponding to the two main metrics. Our testbed was run on a system with Linux Ubuntu 12.04 LTS with an Intel(R) Core (TM) Quad Q8400, 2.66 GHz, 8 GB DDR-SDRAM, and a 500 GB 5400 rpm hard disk.

Our first scenario was set up to reflect the usage of the training period with the assumption that the nature of video access considers a specific period. Users tend to watch videos regularly during a week; therefore, we varied the factor of the period in the order of 1, 2, and 3 weeks. Here, the SVM (sigmoid) was used as the classification model (Ali, W., 2012) due to its superior performance. Here, a well-known SVM library (LIBSVM) based on a C++ implementation (Chang, C.C., 2011) was used to evaluate the five main datasets from NLANR.

The second scenario was conducted to evaluate the overall performance. At this stage, the results from the first experiment (suitable period determination) were used to measure the actual caching efficiency in terms of the HR and the BHR. Again, the two candidates, SVM-LRU (Ali, W., 2012) and SVM-LFU (Sathiyamoorthi, V., 2014), were used to conduct a comparative performance evaluation with our hybrid model (HF-SVM), therein also including two other traditional caching schemes, i.e., LRU and LFU.

A well-known caching emulation for real-world traces, Web Traff (Markatchev, N., 2002), was selected; this is also for model justification proposes (Sathiyamoorthi, V., 2014) (Ali, W., 2012). A fuzzy logic tool embedded in the Octave library (Eaton, J.W., 2016) was used, with Fuzzy Mamdani as the interference engine. The traces from five main web access datasets were evaluated with cache sizes of 2k, with k ranging from 5 to 12.

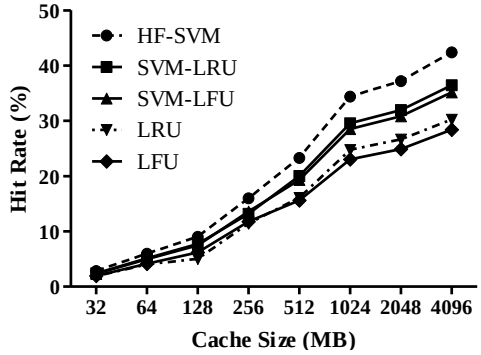
4.4. Simulation Results and Discussion

Two main experiments are discussed here. We compared the performance of our proposed technique. First, Table 3 shows the CCRs of the SVM classification with different training periods on five datasets; higher scores indicate better performance. The classification performance of a 14-days training period is on average outstanding (greater than 91.55%) compared with other training periods - 7 and 21 days, with values of approximately 86.74 to 90.80% and 89.77 to 93.32%.

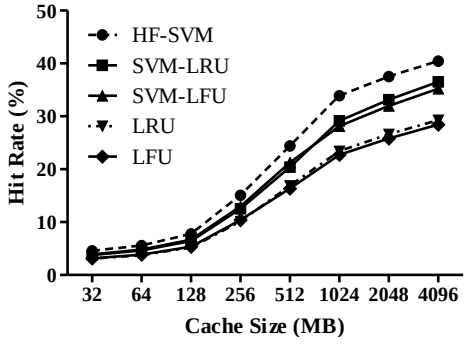
Table 3: CCRs of Different Training Periods

Dataset	CCR (%)		
	7 Days	14 Days	21 Days
Bo2	90.44	**94.38**	92.47
NY	86.74	**91.82**	89.77
SD	90.80	**94.76**	93.32
SV	87.29	**91.55**	90.43
UC	88.65	**92.81**	90.84

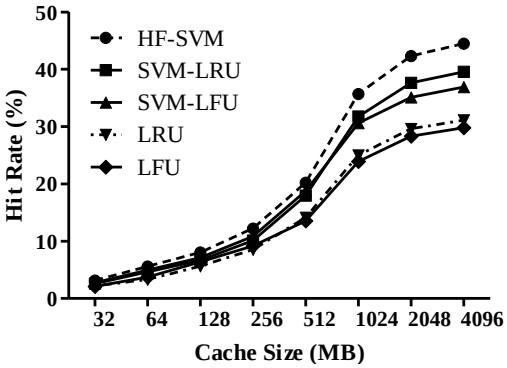
Figures 11 and 12 show the evaluation results of the second scenario on the 5 main datasets (either HR or BHR). In general, as shown in Figure 11, the larger the cache size, the higher the HR, i.e., from approximately 41.41% to less than 4.11%. On average, the performance (HR) of HF-SVM is outstanding on all datasets, i.e., approximately 4.11% to 41.41% (from 32 to 4096 MB); the remaining performances are in the order SVM-LRU (3.48% to 36.46%), SVM-LFU (3.51% to 34.70%), LRU (2.97% to 29.42%), and LFU (2.83% to 28.01%).



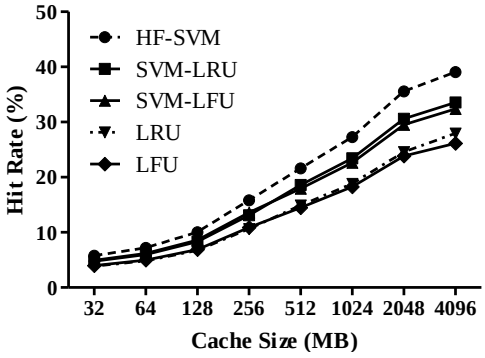
(a) BO2 Dataset



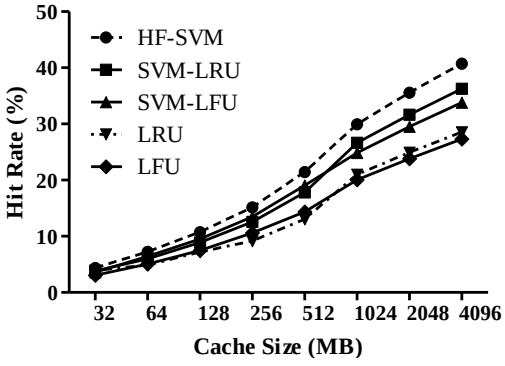
(b) NY Dataset



(c) SD Dataset



(d) SV Dataset

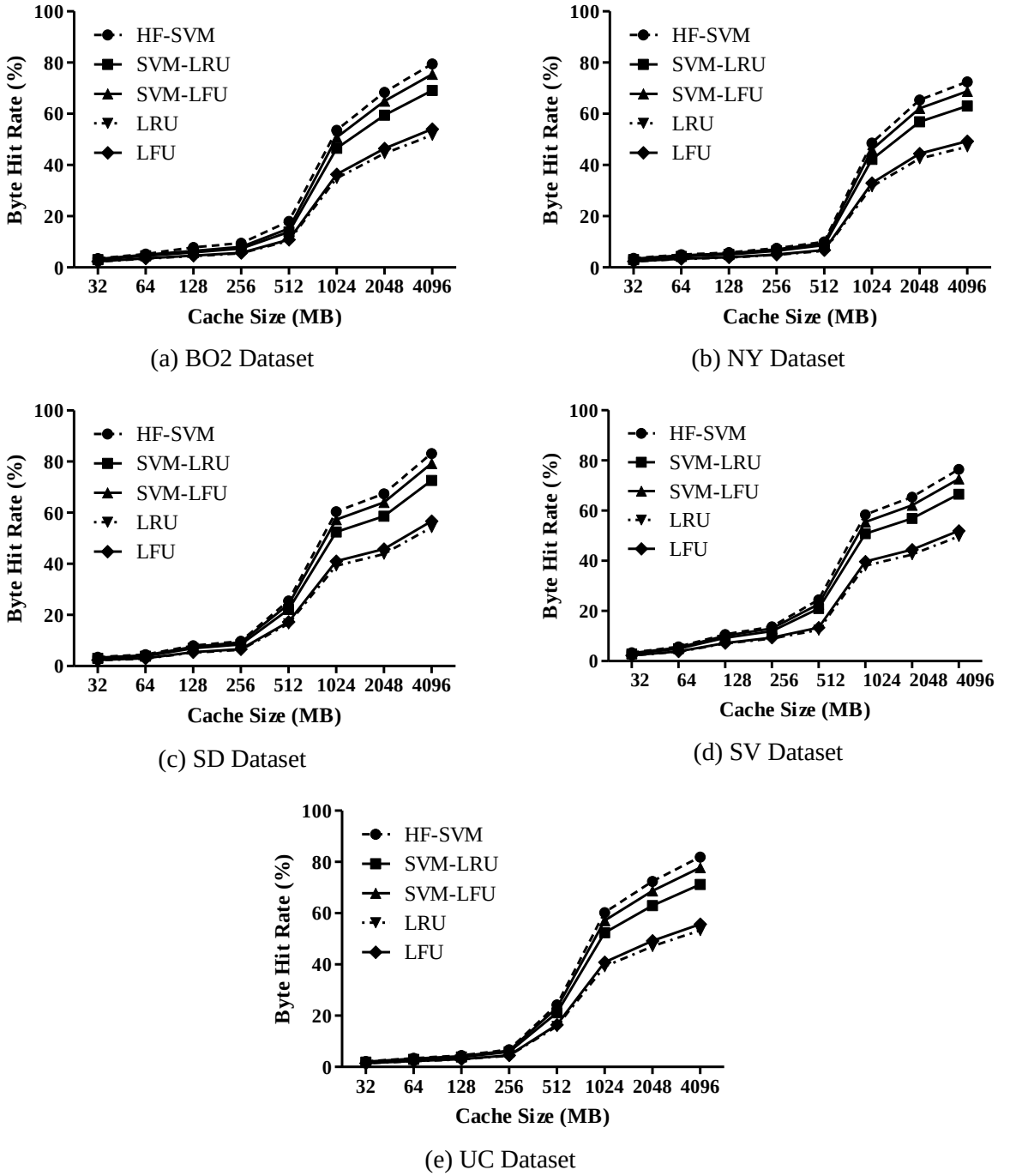


(e) UC Dataset

Figure 11: VoD Caching Performance on Different Datasets: % Hit Rate (HR) vs. Cache Size (MB)

In other words, the efficiency improvement over the others of HF-SVM is in order of 15.60%, 18.14%, 44.14%, and 47.17% for SVM-LRU, SVM-LFU, LRU, and LFU, respectively. Considering HF-SVM, the average HRs on the SD and BO2 datasets are the highest (almost 50% with a 4096 MB cache) but approximately 40% on the other datasets. The other caching schemes follow this trend but with lower HRs. It is also noted that with a small cache size (less than 128 MB), there is no significant difference in terms of HRs among all schemes; however, with a larger cache, the performance is significantly improved with the effect of the SVM.

Similar to the HRs, the larger the cache size, the higher the BHRs (ranging from 3.19% to 78.68%). Figure 12 shows the BHRs. In general, HF-SVM outperforms the other techniques, i.e., from 3.19% to 78.68%, with the other performances in the order of SVM-LFU (3.03% to 74.79%), SVM-LRU (2.77% to 68.50%), LFU (2.17% to 53.54%), and LRU (2.07% to 51.17%). This is also the rationale for using a hybrid fuzzy technique based on LFU and LRU to achieve the highest performance of HF-SVM.



(a) BO2 Dataset

(b) NY Dataset

(c) SD Dataset

(d) SV Dataset

(e) UC Dataset

Figure 12: VoD Caching Performance on Different Datasets: % Byte Hit Rate (BHR) vs. Cache Size (MB)

In other words, the efficiency improvement of HF-SVM is better than the improvements of 5.62%, 15.32%, 48.06%, and 54.79% for SVM-LFU, SVM-LRU, LFU, and LRU, respectively. Similarly, the effects of the intelligent scheme, the SVM, become clearer with larger cache sizes (128 MB and greater). On average, considering HF-SVM, the BHRs on the SD and UC datasets are higher, i.e., more than 80% with a 4096 MB cache size; however, for the other datasets, the BHRs are approximately 60% to 80%. The other caching schemes still follow this trend but achieve lower BHRs.

4.5. Complexity Analysis

Our proposed method consists of two levels: TCRP and fuzzy LFU/LRU in the first level of cache replacement and SVM in the second level, all of which are designed to run continuously. For this reason, the complexity of the algorithm can be analyzed using Big O notation, whose details are in two steps as follows:

First, in general, the time complexity of TCRP cache replacement algorithms (Lee, D., 2001) can be analyzed based on the number of cache hits and misses, and the cost of updating the cache when new data is added or when old data is removed. For example, the LRU algorithm keeps a queue of the most recently used items in the cache. When there is a cache miss, the least recently used item is kicked out. LRU's time complexity can be approximated by the cost of a cache hit/miss, i.e., O(*1*), because the item can be quickly retrieved from the cache. The time complexity of the LFU algorithm is O ($log_2 n$), but it is determined by the frequency of access to the cache items rather than the recency of access.

According to Y. H. Kim et al. (Kim, Y.H., 2000), FLS has been analyzed for complexity, which is divided into three components, i.e., fuzzifier, fuzzy inference engine, and defuzzifier. Assuming that each input variable has 3 fuzzy sets, and each fuzzy set has 3 membership functions, the number of operations required to evaluate the fuzzy set can be approximated as follows:

Each input variable requires three comparisons and three multiplications to evaluate the membership functions of the three fuzzy sets. Three input variables require 18 operations to evaluate membership functions: 3 × (3 comparisons + 3 multiplications). Each rule requires two comparisons to apply the logical operator (e.g., AND and OR) to the membership values of the input variables, such that 12 rules × 2 comparisons = 24 logical operator operations. We then calculated a weighted average, or maximum value, from the membership values to get the fuzzy set output. For each output variable, a weighted average requires 27 multiplications and 26 additions to compute the numerator and denominator.

Since there is only one output variable, aggregating membership values requires 27 × (3 multiplications + 2 additions) = 189 operations. The total number of operations necessary to evaluate the fuzzy set is therefore approximately 18 + 24 + 189 = 231.

To calculate the approximate Big O complexity of this fuzzy set using the complexity analysis according to (Kim, Y.H., 2000), assume *NMF* is the number of membership functions and *NV* is the number of input variables. Big O complexity is the maximum value of the process fuzzy interference engine, which is O(NMF^{NV}).

Second, the implementation and algorithm used determine the Big O complexity of SVM (Chang, C.C., 2011) (Chapelle, O., 2010). SVM algorithms can be described by the number of support vectors, training examples, and dimensions in the feature space. In this study, we used a second level replacement cache engine to implement a linear SVM classifier. The linear SVM has a complexity of O(*nd*), where *n* is the number of training examples and *d* is the dimensionality of the feature space. This is because the linear SVM solves a linear optimization problem involving only the training examples and the weight

vector, and the complexity of solving a linear optimization problem increases proportionally to the number of variables.

To summarize, Table 4 shows the complexity of our method, HF-SVM, against the other algorithms for comparison purposes, i.e., SVM-LRU, SVM-LFU, LRU, and LFU.

Table 4: Complexity Analysis

Cache Replacement Scheme	Complexity
HF-SVM	$O(nd) + O(NMF^{NV})$ + Complexity of TCRP ($O(1)$ or $O(log_2 n)$)
SVM-LRU	$O(nd) + O(1)$
SVM-LFU	$O(nd) + O(log_2 n)$
LRU	$O(1)$
LFU	$O(log_2 n)$

5 Conclusion and Future Work

In this research, an investigation was conducted to integrate traditional and intelligent caching schemes as two-level approaches, in particular for video caching in order to achieve fast user responses and high caching precision by leveraging the advantages of each scheme.

Considering the first level, intensive evaluations were conducted to select the two best traditional algorithms - LRU and LFU. The obtained HRs were 31.23% and 28.38% for LRU and LFU, respectively, and the BHRs were 51.64% and 54.02% for LRU and LFU, respectively. Since the former technique is superior in terms of HR and the latter is superior in terms of BHR, this research proposed an additional weight derived from a fuzzy logic system as the replacement selection criteria, called Fuzzy TCRP or Fuzzy LRU/LFU. This stage was used to support real-time video access as a result of its key benefits of simplicity (fast response).

Considering the constraint of cache size (first level used to support a fast response), the second level primarily focuses on the caching replacement precision, therein leveraging the larger cache size and lack of a real-time requirement. Here, an SVM was proposed as an intelligent SC scheme to make a decision as to whether the video object should be cached in the storage based on knowledge of historical access. In addition, six main features were selected for SVM classification, HTTP request method, URL_ID, size, duration, geographical distance, and similarity, in addition to an adjustable training period for the sliding window concept, properly based on human behaviors to access video contents.

Our two-level caching model, Fuzzy LFU/LRU and SVM (called HF-SVM), was evaluated against SVM-LFU, SVM-LRU, LFU, and LRU using video traces from five main datasets from NLANR for 30-day periods. The results showed that HF-SVM is superior in terms of both HR and BHR, i.e., average values of 44.45% and 83.16%. HF-SVM achieved a better caching efficiency compared with the efficiencies of 47.17%, 44.14%, 18.14%, and 15.60%, for SVM-LFU, SVM-LRU, LFU, and LRU, respectively, in terms of HRs and 48.06%, 54.79%, 5.62%, and 15.32%, respectively, in terms of BHRs.

Noting that although HF-SVM is a superior video caching replacement system, the trade-off is the increase of computational complexity. Additional investigations, assumptions, and constraints should also be explored including large-scale datasets such as for a year trace but also including large-scale caching storage. Other hybrid schemes and optimizations for soft computing should also be investigated. The practicality of HF-SVM embedded in a real caching system, such as Squid, should be investigated further. These topics are all subjects of on-going research considering commercial deployment and hardware cost constraints.

Acknowledgements: This work was funded by grants from Khon Kaen University via the ASEAN and GMS Countries' Personnel Programs 2016-2019 and by an interdisciplinary grant (CSKKU2559) from the Department of Computer Science, Khon Kaen University; by the Research Affairs and Graduate School, Khon Kaen University, Thailand.

References

[1] Aimtongkham, P., So-In, C., & Sanguanpong, S. (2016). A novel web caching scheme using hybrid least frequently used and support vector machine. *In IEEE 13th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 1-6.

[2] Albana, A.A., Suliman, S., & Ahmed, W.A. (2015). Intelligent web objects prediction approach in web proxy cache using supervised machine learning and feature selection. *International Journal of Advances in Soft Computing & Its Applications*, *7*(3), 147-168.

[3] Ali, W., & Shamsuddin, S.M. (2009). Intelligent client-side web caching scheme based on least recently used algorithm and neuro-fuzzy system. *In Advances in Neural Networks–ISNN 2009: 6th International Symposium on Neural Networks, ISNN 2009 Wuhan, China, Proceedings, Part II 6*, 70-79. Springer Berlin Heidelberg.

[4] Ali, W., Shamsuddin, S.M., & Ismail, A.S. (2011). A survey of web caching and prefetching. *International Journal of Advances in Soft Computing and its Applications*, *3*(1), 18-44.

[5] Ali, W., Shamsuddin, S.M., & Ismail, A.S. (2012). Intelligent web proxy caching approaches based on machine learning techniques. *Decision Support Systems*, *53*(3), 565-579.

[6] Andjarwirawan, J., Gunawan, I., & Kusumo, E.B. (2015). *Varnish web Cache application evaluation*, 404-410. Springer Berlin Heidelberg.

[7] Bartere, M.M., & Ingole, P.V. (2012). A Survey on Applications of Genetic Algorithms and Fuzzy Logic in Caching. *International Journal of Science and Engineering Investigations*, *1*(2), 5-7.

[8] Baskaran, K.R., & Kalaiarasan, C. (2016). Improved performance by combining web pre-fetching using clustering with web caching based on SVM learning method. *International Journal of Computers Communications & Control*, *11*(2), 67-178.

[9] Bhaumik, H., Bhattacharyya, S., Nath, M.D., & Chakraborty, S. (2016). Hybrid soft computing approaches to content based video retrieval: A brief review. *Applied Soft Computing*, *46*, 1008-1029.

[10] Calzarossa, M.C., & Valli, G. (2003). A fuzzy Algorithm for web caching. *Simulation Series*, *35*(4), 630-636.

[11] Chakareski, J. (2014). Social video caching, *Signal Processing: Image Communication*, *29*(4), 462-471.

[12] Chang, C.C., & Lin, C.J. (2011). LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, *2*(3), 1-27.

[13] Chapelle, O., & Keerthi, S.S. (2010). Efficient algorithms for ranking with SVMs. *Information retrieval*, *13*, 201-215.

[14] Cisco System VNI: Forecast and Trends, 2017-2022, January 2022, https://twiki.cern.ch/twiki/pub/HEPIX/TechwatchNetwork/HtwNetworkDocuments/white-paper-c11-741490.pdf

[15] Claeys, M., Bouten, N., De Vleeschauwer, D., Van Leekwijck, W., Latré, S., & De Turck, F. (2016). Cooperative announcement-based caching for video-on-demand streaming. *IEEE Transactions on Network and Service Management*, *13*(2), 308-321.

[16] Cobb, J., & ElAarag, H. (2008). Web proxy cache replacement scheme based on back-propagation neural network. *Journal of Systems and Software*, *81*(9), 1539-1558.

[17] Configuring the secure Gateway or secure Gateway proxy, 2016, https://docs.citrix.com/en-us/xenapp-and-xendesktop/xenapp-6-5/xenapp65-w2k8-wrapper/sg-presentation-server-v2/sg-configuring-sg-v2.html
[18] Eaton, J.W., Bateman, D., Hauberg, S., & Wehbring, R. (2016). GNU Octave version 3.0. 1 manual: a high-level interactive language for numerical computations CreateSpace Independent Publishing Platform.
[19] GeoLite legacy databases, 2016, http://dev.maxmind.com/ geoip/ legacy/geolite
[20] ITU International bandwidth usage, January 2022, https://www.itu.int/itu-d/ reports/statistics/2022/11/24/ff22-international-bandwidth-usage/
[21] ITU Measuring digital development: Facts and figures 2022, January 2022, https://www.itu.int/dms_pub/itu-d/opb/ind/d-ind-ict_mdd-2022-pdf-e.pdf
[22] Jang, H., & Topal, E. (2014). A review of soft computing technology applications in several mining problems. *Applied Soft Computing*, *22*, 638-651.
[23] Kim, Y.H., Ahn, S.C., & Kwon, W.H. (2000). Computational complexity of general fuzzy logic control and its simplification for a loop controller. *Fuzzy Sets and Systems*, *111*(2), 215-224.
[24] Kjernsmo, K. (2015). A survey of http caching implementations on the open semantic web. *In The Semantic Web. Latest Advances and New Domains: 12th European Semantic Web Conference, ESWC, Portoroz, Slovenia. Proceedings 12*, 286-301. Springer International Publishing.
[25] Klir, G., & Yuan, B. (1995). *Fuzzy sets and fuzzy logic*, *4*, 1-12. New Jersey: Prentice hall.
[26] Kumar, P.V., & Reddy, V.R. (2014). Novel Web Proxy Cache Replacement Algorithms using Machine Learning Techniques for Performance Enhancement. *International Journal of Engineering Sciences & Research Technology*, *3*(1), 339-346.
[27] Lee, D., Choi, J., Kim, J.H., Noh, S.H., Min, S.L., Cho, Y., & Kim, C.S. (2001). LRFU: A spectrum of policies that subsumes the least recently used and least frequently used policies. *IEEE transactions on Computers*, *50*(12), 1352-1361.
[28] Markatchev, N., & Williamson, C. (2002). Webtraff: A GUI for web proxy cache workload modeling and analysis. *In Proceedings. 10th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*, 356-363.
[29] Mayer, T., & Zignago, S. (2011). Notes on CEPII's distances measures: The GeoDist database.
[30] Mittal, S. (2016). A survey of cache bypassing techniques. *Journal of Low Power Electronics and Applications*, *6*(2), 5.
[31] Nielsen Video on Demand https://www.nielsen.com/wp-content/uploads/sites/2/2019/04/ Nielsen20Global20Video-on-Demand20Report20DIGITAL20FINAL-1.pdf
[32] NLANR IRCache Sanitized Access Log, 2016, http://www.ircache.net
[33] Nurminen, J.K., Meyn, A.J., Jalonen, E., Raivio, Y., & Marrero, R.G. (2013). P2P media streaming with HTML5 and WebRTC. *In IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 63-64.
[34] Ofcom The communications market report, August 2016. http://stakeholders.ofcom.org.uk/ binaries/research/cmr/cmr15/CMR_UK_2015.pdf
[35] Olmos, F., Kauffmann, B., Simonian, A., & Carlinet, Y. (2014). Catalog dynamics: Impact of content publishing and perishing on the performance of a LRU cache. *In IEEE 26th International Teletraffic Congress (ITC)*, 1-9.
[36] Podlipnig, S., & Böszörmenyi, L. (2003). A survey of web cache replacement strategies. *ACM Computing Surveys (CSUR)*, *35*(4), 374-398.
[37] Raghavendra, R., & Belding, E.M. (2010). Characterizing high-bandwidth real-time video traffic in residential broadband networks. *In IEEE 8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, 597-602.

[38] Ramadan, E., Narayanan, A., Dayalan, U.K., Fezeu, R.A., Qian, F., & Zhang, Z.L. (2021). Case for 5g-aware video streaming applications. *In Proceedings of the 1st Workshop on 5G Measurements, Modeling, and Use Cases*, 27-34.
[39] Sabeghi, M., & Yaghmaee, M.H. (2006). Using fuzzy logic to improve cache replacement decisions. *Int J Comput Sci Network Security*, *6*(3A), 182-188.
[40] Sathiyamoorthi, V., & Ramya, P. (2014). Enhancing Proxy Based Web Caching System using Clustering based Pre-fetching with Machine Learning, 3(7), 463-469.
[41] So-In, C., Permpol, S., & Rujirakul, K. (2016). Soft computing-based localizations in wireless sensor networks. *Pervasive and Mobile Computing*, *29*, 17-37.
[42] The MD5 Message-Digest Algorithm, 2020, https://tools.ietf.org/html/rfc132,
[43] Vakali, A. (2002). Evolutionary techniques for web caching. *Distributed and Parallel Databases*, *11*, 93-116.
[44] Valenza, F., & Cheminod, M. (2020). An Optimized Firewall Anomaly Resolution. *Journal of Internet Services and Information Security*, *10*(1), 22-37.
[45] Wang, C., Li, X., Zhou, X., Wang, A., & Nedjah, N. (2016). Soft computing in big data intelligent transportation systems. *Applied Soft Computing*, *38*, 1099-1108.
[46] Wust, C.C., Steffens, L., Bril, R.J., & Verhaegh, W.F. (2004). Qos control strategies for high-quality video processing. *In IEEE Proceedings. 16th Euromicro Conference on Real-Time Systems, 2004. ECRTS*, 3-12.
[47] Zhao, H., Zheng, Q., Zhang, W., & Li, H. (2016). MSC: a multi-version shared caching for multi-bitrate VoD services. *Multimedia Tools and Applications*, *75*, 1923-1945.

Authors Biography

Sovannarith Heng received the B.S. degree from the Royal University of Phnom Penh, in 2005, and the M.S. degree from the Ateneo de Manila University, in 2010. He is currently pursuing the Ph.D. degree with the Department of Computer Science, Khon Kaen University, Thailand. His research interests include image processing, wireless multimedia sensor networks, computer networks, and distributed systems.
Email: sovannarith@rupp.edu.kh, Orcid: https://orcid.org/0000-0002-8649-1079

Phet Aimtongkham is currently a lecturer and researcher at the Department of Computer Science, Faculty of Science, Khon Kaen University, Thailand. In 2020, he has recent received a Ph.D. in Information Technology from the Department of Computer Science, Khon Kaen University, Thailand, where he also received the B.S. and M.S. degrees in 2013 and 2017, respectively. His research interests include computer networking, Internet of Things (IoTs), multimedia networks, cybersecurity and machine learning.
Email: phetim@kku.ac.th Orcid: https://orcid.org/0000-0001-5289-1149

Chakchai So-In

Chakchai So-In is a Professor of Computer Science in the Department of Computer Science, Khon Kaen University, KK, TH. He received B.Eng./M.Eng. degrees from Kasetsart University, BKK, TH in 1999/2001 and M.S./Ph.D. degrees from Washington University in St. Louis, MO, USA in 2006/2010, all in Computer Engineering. He has interned with Cisco Networking Academy (CNAP-NTU, SG), Cisco Systems (Silicon Valley, USA), WiMAX Forums (USA), and Bell Labs (Alcatel-Lucent, USA). His research interests include computer networking and the internet, wireless and mobile networking, the Internet of Things, wireless sensor networks, signal processing, cybersecurity, cyber-physical systems, and applied intelligent systems. He has served as an associate editor for IEEE Access, PLOS ONE, Wireless Networks, WCMC, PeerJ (CS), and ECTI-CIT and as a committee member/reviewer for many journals/publishers such as IEEE, Elsevier, Springer, Wiley, IET, Inderscience, IEICE, and ETRI; and conferences such as GLOBECOM, ICC, VTC, WCNC, ICNP, ICNC, and PIMRC. He has authored/coauthored over 100 international (technical) publications, including some in IEEE JSAC, IEEE TCCN, IEEE/CAA, IEEE Commun./Wireless Commun. Mags, IEEE IoT J., IEEE System J., COMNET, MONET, and ESWA; and 10 books, including Mobile & Wireless Nets with IoT, Computer Network Lab., and Network Security Lab. He is also a senior member of IEEE and ACM.
Email: chakso@kku.ac.th, Orcid: https://orcid.org/0000-0003-1026-191X