SHOLO: Similarity-based Reduction Scheme for Efficient Data Transmission and Energy Saving in Multimedia Sensor Networks

Fouad Al Tfaily[a], Chady Abou Jaoude[b], Hassan Harb[c], Hussein Hazimeh[a], Hassan Kanj[c]
[a]*Department of Computer Science, Lebanese University, Beirut, Lebanon*
[b]*TICKET Lab, Faculty of Engineering, Antonine University, Baabda, Lebanon*
[c]*College of Engineering and Technology, American University of the Middle East, Kuwait*
Emails: [‡]fouad.tfaily@st.ul.edu.lb, [‡]chady.aboujaoude@ua.edu.lb, [‡]hassan.harb@aum.edu.kw,
[‡]hussein.hazimeh@ul.edu.lb, [‡]hassan.kanj@aum.edu.kw

***Abstract*—Nowadays, Wireless Multimedia Sensor Networks (WMSNs) plays a vital role in modern surveillance systems and enables innovative solutions across diverse sectors. In such networks, detecting and minimizing data redundancies has become crucial for both prolonging network lifespan and improving data quality. In this paper, we propose a two-layer reduction scheme, called as SHOLO, that efficiently detects and removes similarities in SHOrt and LOng term video data collected in WMSNs, thus conserve sensor energies and extending network lifetime. In the first layer, SHOLO detects and removes similarities among successive frames captured by video node during the same period time through two novel distance-based methods: Difference Hash (DH) and Pixel Intensity Threshold (PIT). In the second layer, SHOLO searches long-term similarities among frames collected in the same period to detect scene variation and zone dynamicity. Then, we introduced two similarity reduction methods in such layer: Intensity-based Difference (ID) and Block-based Difference (BD). We conducted extensive simulations using real-world video data set to validate the efficiency of the proposed framework. The results demonstrated that SHOLO can reduce up to 93.8% of collected video data, leading to tremendous energy savings and network lifetime compared to existing approaches.**

***Index Terms*—Wireless Multimedia Sensor Networks, Artificial Intelligence, Similarity Detection, Redundancy Reduction, Energy Saving, Data Transmission.**

I. Introduction

In the last years, wireless multimedia sensor networks (WMSNs) have rapidly grown as a resourceful means of collecting environmental data in different areas, including agriculture, surveillance and environmental monitoring [1, 2]. However, the enormous amount of video data generated by the WMSN presents significant challenges related to the consumption of energy resources and the utilization of available bandwidth [3]. Data reduction methods are key solutions to overcome such challenges by lowering the amount of transmitted data [4, 5].

In the literature, we can distinguish between two approaches of data reduction: compression-based and correlation-based. On one hand, the compression techniques are used to decrease the size of each frame inside a video before transmitting the data to the sink [6–8]. For instance, the authors of [6] proposed a distributed video transmission reduction algorithm (DiViTRA) in which compression operation is applied during capturing and communication operations. In the first operation, three algorithms (ORB, BF, and GMS) are introduced to adapt the capture frames of the video sensor, while, in the second operation, DiViTRA used two data reduction algorithms (principal component analysis and Huffman codding) to compress the size of frames sent to the sink node. On the other hand, spatio-temporal correlations among video sensors mostly occur due to the requirement of densely deployed nodes and the slow changes in the monitored area [9–11]. For instance, the authors of [11] introduced a temporal-based data reduction (TDR) technique to minimize the number of frames sent from each node to the sink. In their technique, each node calculates the similarity between video frames collected in each period; when a high similarity is detected, the node removes the current frame and adapts its sampling rate accordingly.

Although the current studies have substantially advanced the data reduction in WMSNs, two main limitations are still eminent: 1) complexity: they used complex algorithms which are not suitable to limited computation of the node. 2) Data quality: current techniques do not mostly benefit of various types of similarities existing in WMSN that could largely enhance the quality of data after being reduced. In this paper, we propose a similarity-based reduction technique called as SHOLO for minimizing data transmission and maximizing energy saving in WMSNs. SHOLO design relies on periodic data collection model, where video nodes monitor the target zone and submit the collected data in a periodic manner. Then, SHOLO consists of two layers: short-term similarity detection and long-term similarity detection. The first layer aims to detect and remove similarity among successive frames captured by a sensor node during the same period time, and it introduces two different distance-based methods: Difference Hash (DH) and Pixel Intensity Threshold (PIT). The second layer aims to find significant changes of the scene collected by each video sensor separately during a period time. We employed, at this layer, two temporal correlation based techniques: Intensity-based Difference (ID) and Block-based Difference (BD). All such methods apply statistical analysis and time series modeling to reveal the significant fluctuations that lead to reduce the data transmission while sustaining the event detection accuracy.

979-8-3503-9281-4/25/$31.00 ©2025 IEEE

II. SHOLO Architecture

The fundamental idea behind SHOLO is to develop an energy-saving system that encompasses data acquisition, pre-processing, and transmission while minimizing the redundant video frames and events collected by a sensor and maintaining useful information for decision-making.

A. Periodic Data Collection Model

We consider that the network consists of n nodes, e.g. $\mathcal{N} = \{N_1, N_2, \ldots, N_n\}$ where the camera of each node N_i has a field of view (FoV) to monitor a part of the surrounding area. FoV is represented by a vector of 4-tuple: $\{P; R_s; \overrightarrow{V}; \theta\}$ where P is the position of a node, R_s is its sensing range, $\overrightarrow{V}$ is the line of sight of the camera node or its sensing direction, and θ is the offset angle of the FoV. Furthermore, we employed a periodic collection model to ensure a continuous monitoring of the target zone and real-time events detection. In such model, each period time T is divided into τ slot times: $T = \{t_1, t_2, \ldots, t_\tau\}$. During such period, each node N_i will collect a video data of τ frames, $\mathcal{F}_i = \{F_1, F_2, \ldots, F_\tau\}$, where each frame $F_{j\in[1,\tau]}$ is collected during one time slot t_j. Each frame F_j is represented by a matrix of pixels $P_j[r][c]$, where r and c indicate the total number of pixels in each row and column respectively. At the end of the period, the frame vector generated by each node will be then used by our framework to assess the zone dynamicity, before sending it to the user. Figure 1 presents an example of the periodic data collection model applied in our framework.

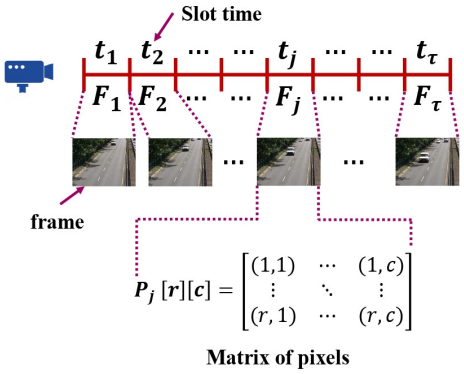


Figure 1. Illustration of periodic collection model.

B. SHOLO First Layer: Short-Term Similarity Detection

In WMSNs, there is an important need to continuously monitor a target to detect any potential events. This leads to a huge amount of data collected by each node that is mostly redundant. Hence, the first layer of SHOLO exploits the temporal correlation between two frames, e.g. F_{ij} and F_{ik}, collected by each video node N_i during the same period, then only sends non-redundant frames to the user. Therefore, we propose at this layer two different methods that use various distance metrics and feature extraction concept to compare frames and measure their degree of similarity.

1) Difference Hash (DH) Method: DH employs the perceptual hashing concept to calculate the similarity between two consecutive frames obtained by a sensor node. Perceptual hashing is a method that creates a compact representation of an image while keeping the perceptual properties intact. Its objective refers to generate the same hash values for perceptually similar images. Mathematically, a perceptual hash function, $PHF_k(\cdot)$, is a deterministic algorithm that has generated a k-bit perceptual hash $h = PHF_k(s) \in \{0,1\}^k$ from an input string s.

In this study, DH will use a specific perceptual hash function called the difference hash (e.g. $dHash$). Indeed, $dHash$ computes the hash value by subtracting the intensity values of adjacent pixels in the picture. Let F_{ij} be a frame with its corresponding pixel matrix $P_j[r][c]$. The $dHash$ function can be represented as follows:

$$dHash(F_{ij}) = \text{binary\_string}\big([F_{ij}(x,y) < F_{ij}(x,y+1) \text{ for } y \text{ in range}(c-1) \text{ for } x \text{ in range}(r)]\big) \quad (1)$$

where $F_{ij}(x,y)$ denotes the pixel intensity at position (x,y), and $binary\_string$ converts the boolean array to a binary string representation.

Furthermore, this layer implements Hamming distance to measure the similarity between two $dHash$ values of two frames. By definition, Hamming distance between two binary strings of equal lengths is the number of characters at which the corresponding bits differ. Thus, the Hamming distance($\mathcal{H}$) between two $dHash$ values, e.g. $dHash_j$ and $dHash_k$, for two frames respectively, F_{ij} and F_{ik}, is calculated as:

$$\mathcal{H}(dHash_j, dHash_k) = \sum_{i=1}^{d} |dHash_{j_i} - dHash_{k_i}| \quad (2)$$

where i represents the i-th bit of $dHash_{j_i}$ and $dHash_{k_i}$, respectively, and d is the length of the $dHash$ values.

Now, let consider that $MaxH$ be the maximum size of the hash values $dHash_{j_i}$ and $dHash_{k_i}$ as follows:

$$MaxH = \max(size(dHash_{j_i}), size(dHash_{j_i})) \quad (3)$$

where $size(dHash_{j_i})$ denotes the size of the hash value of the first frame. Then, the similarity ($Sim_{DH}(j,k)$) between two consecutive frames, F_{ij} and F_{ik}, collected by a sensor node is calculated using the Hamming distance $\mathcal{H}$ and $MaxH$ as follows:

$$Sim_{DH}(j,k) = \left(1 - \frac{\mathcal{H}(dHash_j, dHash_k)}{MaxH}\right) \times 100 \quad (4)$$

According to above equation (4), the similarity value between to frames will range between 0 and 100, where 0 indicates that both frames are completely dissimilar and 100 indicates a perfect frames match.

2) Pixel Intensity Threshold (PIT) Method: This method tries to match the intensity of the pixels of two matrices, $P_j[r][c]$ and $P_k[r][c]$, correspond to two consecutive frames, F_{ij} and F_{ik} respectively, based on a threshold. For the ease of comparison, the frames are first converted to grayscale then

PIT computes the absolute difference between the corresponding pixel intensities. Mathematically, the absolute difference $\mathcal{A}_{PIT}$ between two frames is calculated as:

$$\mathcal{A}_{PIT}(x,y) = |P_j(x,y) - P_k(x,y)| \tag{5}$$

where $P_j(x,y)$ and $P_k(x,y)$ denote the pixel intensities at position (x,y) in both frames respectively. Then, PIT compares each absolute difference value $\mathcal{A}_{PIT}(x,y)$ with a predefined threshold σ: if $\mathcal{A}_{PIT}(x,y)$ is less than or equal to σ, the observed pixels are considered similar, and dissimilar otherwise. Accordingly, the similarity between F_{ij} and F_{ik} is calculated as:

$$Sim_{PIT}(j,k) = \left(1 - \frac{\sum_{x=1}^{r}\sum_{y=1}^{c}[\mathcal{A}_{PIT}(x,y) > \sigma]}{r \times c}\right) \times 100 \tag{6}$$

where $[\cdot]$ denotes the Iverson bracket, which results in 1 if the condition inside the brackets is true and 0 otherwise.

3) Data Reduction Algorithm for SHOLO First Layer: Algorithm 1 shows the process of layer and how redundant video frames are deleted. The algorithm takes, as input, the set of frames collected by a node during a certain period along with the similarity threshold $\Omega \in [0,1]$. The node saves the first frame collected at the beginning of the period in its memory and sends simultaneously to its CH. Then, it computes the similarity between the saved frame and ones taken in the next slot times; check represents either the similarity function in the two methods (eqs. 4, 6). If a high similarity is detected, e.g. above threshold Ω, the next frame is considered redundant, thus it will be deleted by the node. Otherwise, e.g. a low similarity is noticed, the node saves the new frame for a later comparison and sends it to the user.

Algorithm 1 Data Reduction for SHOLO First Layer.

Require: Video node: N_i; Period time: $T = \{t_1, t_2, \ldots, t_\tau\}$; Video frames collected at T: $\mathcal{F}_i = \{F_1, F_2, \ldots, F_\tau\}$; Similarity threshold: Ω.
Ensure: Non-redundant video frames.
1: $Saved \leftarrow F_{i_1}$
2: Send(F_{i_1}) to CH
3: **for** each slot time $q \in$ [2,τ] **do**
4: **if** $check(F_{i_q}, Saved) \geq \Omega$ **then**
5: delete F_{i_q}
6: **else**
7: $Saved \leftarrow F_{i_q}$
8: Send($Saved$) to CH
9: **end if**
10: **end for**

C. SHOLO Second Layer: Long-Term Similarity Detection

Unlike layer 1, the second layer checks the similarity between scenes detected by a node during the same period. Indeed, if a significant number of scenes is detected during a period time, this indicates a high dynamicity of the zone thus the node should increase its capturing speed to detect all the scenes. Otherwise, zone slowly changes and few scenes are noticed, the node should decrease its capturing speed to avoid redundant frames and save battery. At this layer, we propose two methods to track scene changes during a period of time.

1) Intensity-based Difference (ID) Method: In this method, we compare frames based on their pixel intensity differences according to a predefined threshold. It goes through the frames in a subset of frames one by one and determines the similarity with a reference frame, the first one taken in the period. Consider two frames, the reference (F_{if}) and the j-th frame in the period (F_{ij}), ID calculates the absolute difference $\mathcal{A}_{ID}$ between F_{if} and F_{ij} as follows:

$$\mathcal{A}_{ID}(x,y) = |F_{if}(x,y) - F_{ij}(x,y)| \tag{7}$$

where $F_{if}(x,y)$ denotes the pixel intensity at position (x,y) in frames $F_{if}(x,y)$. Then, ID compares each absolute difference value $\mathcal{A}_{ID}(x,y)$ with a predefined threshold σ. If $\mathcal{A}_{ID}(x,y)$ is less than or equal to σ, the corresponding pixels are considered similar; and dissimilar otherwise.

Therefore, a scene is detected if a low similarity id detected between two frames as follows:

$$Sim_{ID}(f,i) = \left(1 - \frac{\sum_{x=1}^{r}\sum_{y=1}^{c}[\mathcal{A}_{ID}(x,y) < \sigma]}{r \times c}\right) \times 100 \tag{8}$$

where $[\cdot]$ denotes the Iverson bracket, which is evaluated to 1 if the condition inside the brackets is true and 0 otherwise, and r and c are the dimension of matrices of both frames.

Finally, ID iterates through the sequence of frames collected during a period and calculates the similarity for each frame F_ik. If $Sim_{ID}(f,i)$ falls below a predefined correlation threshold ε, ID identifies a significant change in the set and marks the frame F_i as a key frame. The process continues until all frames in the sequence have been analyzed.

2) Block-based Difference (BD) Method: BD extends the concept of temporal correlation analysis to a block-based approach. Instead of the process of comparing single pixels, BD divides the frames into smaller blocks and calculates the correlation between the corresponding blocks in consecutive frames. Moe specifically, this layer takes frames apart into blocks, and compares individual blocks to capture localized temporal variations and minimize the effect of minor pixel-level changes. The block-based approach helps to highlight essential changes in the video content very efficiently which reduces the need to forward unnecessary frames in WMSNs.

Let F_{if} be the reference frame and F_ij be the j-th frame in the video. BD technique divides each frame into non-overlapping blocks of size $b \times b$ pixels. Let $B_{if}(m,n)$ and $B_{ij}(m,n)$ denote the (m,n)-th block in frames F_{if} and F_{ij}, respectively, where $m \in 1,2,\ldots,c/b$ and $n \in 1,2,\ldots,r/b$. Then, for each block $B_{ij}(m,n)$ in frame F_{ij}, BD calculates the correlation coefficient $CC_{i_{fj}}(m,n)$ between $B_{if}(m,n)$ and $B_{ij}(m,n)$ as:

$$CC_{i_{fj}}(m,n) = \frac{1}{b^2 \times \sigma_{if}(m,n) \times \sigma_{ij}(m,n)} \times \sum_{x=1}^{b}\sum_{y=1}^{b}(B_{if}(m,n)(x,y) - \mu_{if}(m,n)) \times (B_{ij}(m,n)(x,y) - \mu_{ij}(m,n)) \quad (9)$$

where $\mu_{if}(m,n)$ and $\mu_{ij}(m,n)$ are the mean pixel intensities of blocks $B_{if}(m,n)$ and $B_{ij}(m,n)$, respectively, and $\sigma_{if}(m,n)$ and $\sigma_{ij}(m,n)$ are the standard deviations of pixel intensities in blocks $B_{if}(m,n)$ and $B_{ij}(m,n)$, respectively.

After that, BD calculates the average correlation coefficient $ACC_{i_{fj}}$ as follows:

$$ACC_{i_{fj}} = \frac{\sum_{m=1}^{c/b}\sum_{n=1}^{r/b} CC_{i_{fj}}(m,n)}{(c/b) \times (r/b)} \quad (10)$$

3) Capturing Rate Adaptation Algorithm for SHOLO Second Layer: In order to reduce the number of frames captured during each period, we propose to adapt the capturing frequency of each node based on the number of detected scenes in the period; less scenes are detected, more the number of frames should be reduced in the next period, and vice versa. This allows to dynamically adapt the capturing rate of a node according to the changes in the monitored zone. Consider the number of detected scenes detected by a node N_i, after applying any similarity detection methods (ID or BD) during a period time, is $\mathcal{S}_i$, then the new capturing rate ($\mathcal{T}_i$) in the next period time will be calculated as follows:

$$\mathcal{T}_i = \frac{\mathcal{S}_i \times 100}{\tau} \quad (11)$$

Mostly, WMSN applications are not equally important and can have different criticality according to the monitored zone. Thus, beside the number of detected scenes, the rate adaptation algorithm proposed at layer 2 also takes into consideration the application requirements. Accordingly, we represent the criticality of a WMSN application by a minimum capturing frequency of a node, e.g. $\mathcal{T}_i^{min}$; $\mathcal{T}_i^{min}$ takes a value between 0 and 100 indicating low and high criticality application levels respectively.

Algorithm 2 shows the entire process of capturing rate adaptation at the end of each period. The algorithm takes the number of frames at each period along with the application criticality level, and identifies the next capturing rate at the next period. After being deployed, the node captures the maximum number of frames during the period to avoid losing any scene or event (lines 1-5). Then, at the end of the period, the node calculates the number of detected scenes according to both methods proposed at layer 2, e.g. ID or BD, and accordingly computes the new capturing rate for the next period (lines 7 and 8). Particularly, if the new sampling rate goes below the minimum threshold defined for the application, the node sets the capturing rate to that threshold to maintain the quality of collected video (lines 9 and 10).

Algorithm 2 Capturing Rate Adaptation for Second Layer.

Require: Video node: N_i; Number of frames per period: τ; Application criticality level: $\mathcal{T}_i^{min}$.
Ensure: New capturing rate: $\mathcal{T}_i$.
1: $\mathcal{T}_i \leftarrow \tau$, // capturing rate is initialized to the maximum
2: **while** $Energy > 0$ **do**
3: **for** each period p **do**
4: takes frames at $\mathcal{T}_i$ rate
5: **end for**
6: **for** each end of period **do**
7: calculate $\mathcal{S}_i$ based on Eqs. 8, or 10
8: calculate $\mathcal{T}_i$ according to Eq. 11
9: **if** $\mathcal{T}_i < \mathcal{T}_i^{min}$ **then**
10: $\mathcal{T}_i \leftarrow \mathcal{T}_i^{min}$
11: **end if**
12: **end for**
13: **end while**

III. Performance Evaluation

This section demonstrates the performance of SHOLO by describing the used dataset, the simulation setup, and the benchmark comparisons with existing techniques.

A. Dataset Description

In our simulation, we utilized the CDnet dataset [12], which comprises 31 videos capturing both indoor and outdoor scenes featuring cars, trucks, and pedestrians under diverse scenarios. The dataset used in our model includes video sequences recorded under varying conditions, such as changes in illumination, moving backgrounds, and dynamic objects. These videos were captured using various camera types, ranging from low- to high resolution as well as thermal cameras. The spatial resolution of the videos in CDnet is set to 320×240, with video lengths varying between $1,000$ and $8,000$ frames.

B. Simulation Setup

The hardware setup for our implementation included an HPE ProLiant ML150 Gen9 server powered by a 64-bit, 6-core Intel Xeon CPU running at 1.7 GHz. The server was configured with a 240 GB SSD for primary storage, and an 8 TB HDD for additional capacity. The system operated on Windows Server 2012 R2, and our technique was developed using the Python framework. Table I presents the parameters used in SHOLO along with their values tested in the simulation.

Parameter	Description	Values
N	Number of video cameras	10, 20, 30
σ	Pixel similarity	fixed to 5
τ	Number of frames per period	10, 20, 30
Ω	Similarity threshold	0.7, 0.8, 0.9
$\mathcal{T}_i^{min}$	Application criticality	30, 50, 80

Table I
Simulation environments.

C. Sort-Term Similarity Detection Layer: Evaluation and Discussion

This section evaluates various methods (e.g. DH or PIT) proposed at the first layer of SHOLO in terms of similarity detection and latency. Firstly, Figure 2 shows the similarity detection of each method according to different metrics by varying the value of one of them and fixing the others in each subfigure. The following observations are eminent:

- PIT demonstrates better performance in terms of detecting similar video frames compared to DH. Subsequently, PIT enhances the similarity percentage up to 6.2% in the best case compared to DH.
- The similarity percentage increases with the increase of the frames number per period (τ). Particularly, Figure 2(b)) shows that the similarity rate of PIT is increased from 84.2% to 93.8% when τ's value is increased from 10 to 30. This is due to the temporal correlation between frames collected in the same period time, especially when the monitored zone changes slowly.
- The decrease of similarity threshold value (Ω) will increase the similarity between the frames video. Fro instance, the similarity percentage of PIT is decreased from 92.1% to 79.3% when Ω' value is increased from 0.7 to 0.9 (Figure 2(c)).

On the other hand, Figure 3 shows the performance comparison of the best results obtained in the first layer of SHOLO and two similarity-based detection models in the literature (e.g. DiViTRA [6] and TDR [11]) in terms of similarity and latency metrics. The following observations are noticed:

- From the similarity percentage point of view, DH outperforms the efficiency of DiViTRA and TDR up to 7.1% and 8.3% respectively. Whilst, PIT outperforms both existing techniques up to 5.5% and 8.8% respectively.
- From the latency point of view, DH decreases the latency up to 86.8% and 85.4% compared to DiViTRA and TDR. Whilst, PIT decreases the latency of the same methods by 85.3% and 83.9% respectively.

D. Long-Term Similarity Detection Layer: Evaluation and Discussion

This section evaluates various methods (e.g. ID and BD) proposed at the second layer of SHOLO in terms of reducing the capturing rate of cameras. Firstly, Figure 4 shows the camera capturing rate, for the first 20 periods, after applying each method when varying the application criticality level ($\mathcal{T}_i^{min}$) and fixing the frames per period to 20. The obtained results reveal the following observations:

- ID reduces the capturing rate of the camera more than BD method. This indicates that pixel-pairwise comparison between sequential frames are mostly high redundant, while block-wise comparison requires a significant changes in the environment to detect a scene.
- The capturing rate of the camera is increased when the application criticality increases. For instance, most of the capturing rates during the periods are adapted to less than 10 frames per period when $\mathcal{T}_i^{min}$ is 30% (Figure 4(a)) and to above 16 when $\mathcal{T}_i^{min}$ is 80% (Figure 4(b)).

Although our methods have significantly reduced the number of frames collected in each period, such reduction is subjected to frame loss. Thus, accuracy is considered as an essential metric when adapting capturing rate. Figure 5 shows the accuracy of each method by varying the application criticality and fixing the frames per period to 20. The obtained results show that:

- Both proposed methods show a good accuracy level, e.g. in range [73.4,92.1]. However, BD method demonstrates its superiority compared to ID due to more number of frames submitted by applying BD.
- the accuracy of all methods increases with the increase of the application criticality. This is because more frames will be submitted when the application criticality increases. For instance, the accuracy of BD increases from 84.1 to 92.1 when the application criticality increases from 30 to 80.

IV. Conclusion

This paper introduced SHOLO framework that aims to minimize the data redundancy and conserve sensor energy in WMSNs. SHOLO used multi-layer similarity reduction methods at camera sensors to detect short and long term similarities among collected video data. Extensive simulations using real-world video datasets demonstrated the framework efficiency, showing that SHOLO can reduce video data by up to 93.8% and save energy node compared to existing approaches.

References

[1] Ramiz Salama, Chadi Altrjman, and Fadi Al-Turjman. A survey of the architectures and protocols for wireless sensor networks and wireless multimedia sensor networks. *NEU journal for artificial intelligence and internet of things*, 2(3), 2023.

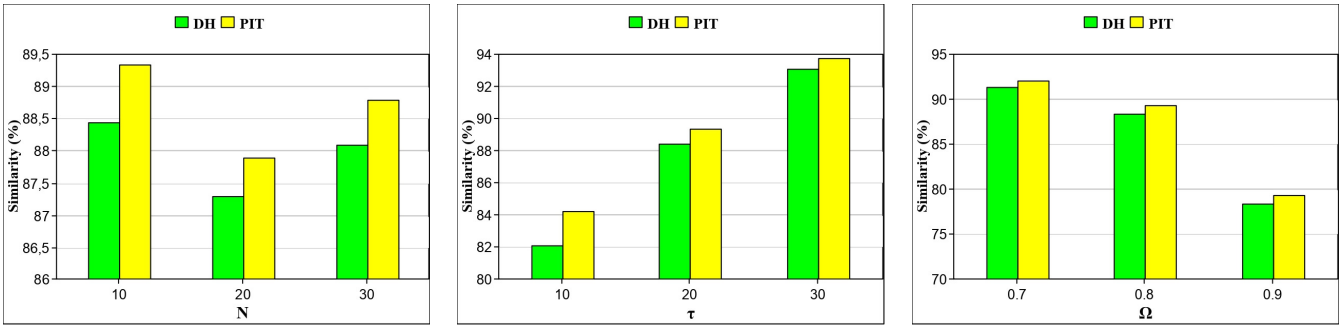
[2] Hassan Harb and Abdallah Makhoul. Energy-efficient scheduling strategies for minimizing big data collection in cluster-based sensor networks. *Peer-to-Peer Networking and Applications*, 12(3):620–634, 2019.

[3] Hassan Harb, Abdallah Makhoul, Ali Jaber, and Samar Tawbi. Energy efficient data collection in periodic sensor networks using spatio-temporal node correlation. *International Journal of Sensor Networks*, 29(1):1–15, 2019.

[4] Suman Ruchika. Exploring machine learning applications for enhancing security and privacy in multimedia iot: A comprehensive review. *Machine Learning in Multimedia*, pages 34–49, 2025.

[5] Nazli Tekin and Vehbi Cagri Gungor. Analysis of compressive sensing and energy harvesting for wireless multimedia sensor networks. *Ad Hoc Networks*, 103: 102164, 2020.

[6] Iman Kadhum Abbood and Ali Kadhum Idrees. Distributed video transmission reduction approach for energy saving in wmsns. *International Journal of Communication Systems*, page e5880.



(a) $\tau = 20$, $\Omega = 0.8$ (b) $N = 10$, $\Omega = 0.8$ (c) $N = 10$, $\tau = 20$

Figure 2. Similarity percentage after applying various methods proposed at SHOLO layer 1.

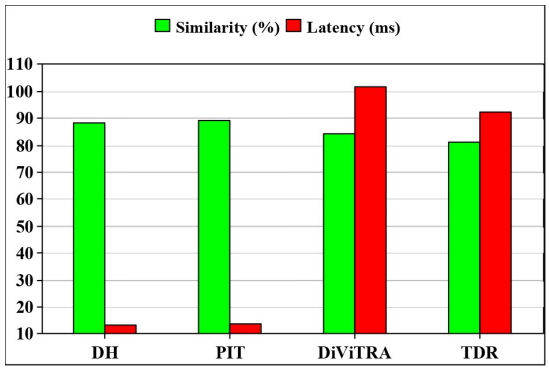
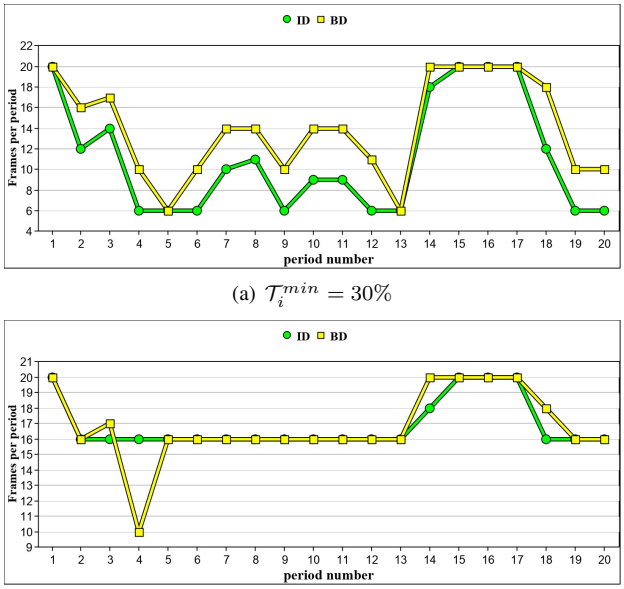


Figure 3. Similarity percentage vs latency: our methods vs state-of-the-art.



(a) $\mathcal{T}_i^{min} = 30\%$

(b) $\mathcal{T}_i^{min} = 80\%$

Figure 4. Capturing rate adaptation of a camera after applying various methods proposed at SHOLO layer 2. $\tau = 20$.

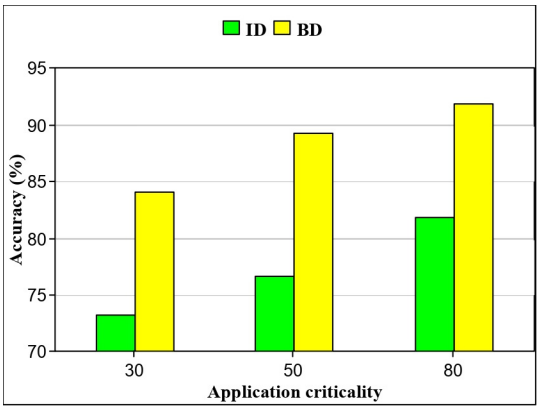


Figure 5. Accuracy percentage of various methods used at layer 2. $\tau = 20$.

[7] G Sudha and C Tharini. Energy-efficient image compression algorithm for wireless sensor networks in structural health monitoring applications. *Automatika*, 65(2):483–495, 2024.

[8] Abdallah Makhoul, David Laiymani, Hassan Harb, and Jacques M Bahi. An adaptive scheme for data collection and aggregation in periodic sensor networks. *International journal of sensor networks*, 18(1-2):62–74, 2015.

[9] Jana Koteich, Christian Salim, and Nathalie Mitton. Spatio-temporal data reduction technique in wvsn for smart agriculture. In *2022 18th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 345–350. IEEE, 2022.

[10] Sellamuthu Suseela, Ravi Krithiga, Muthusamy Revathi, Gajendran Sudhakaran, and Reddiyapalayam Murugeshan Bhavadharini. Energy aware optimal routing model for wireless multimedia sensor networks using modified voronoi assisted prioritized double deep q-learning. *Concurrency and Computation: Practice and Experience*, 36 (6):e7945, 2024.

[11] Jana Koteich, Christian Salim, and Nathalie Mitton. Image processing based data reduction technique in wvsn for smart agriculture. *Computing*, 105(12):2675–2698, 2023.

[12] Nil Goyette, Pierre-Marc Jodoin, Fatih Porikli, Janusz Konrad, and Prakash Ishwar. Changedetection. net: A new change detection benchmark dataset. In *2012 IEEE computer society conference on computer vision and pattern recognition workshops*, pages 1–8. IEEE, 2012.