Smart Blue Light Pole-based Real-Time Crowd Counting for Smart Campuses

Yitao Chen*, Kaiqi Zhao†, Krishna Gundu*, Zohair Zaidi*, Ming Zhao*
*Arizona State University
{ychen404, kgundu1, zohair.zaidi, mingzhao}@asu.edu
†Oakland University
kaiqizhao@oakland.edu

Abstract*—Crowd counting on a smart campus can provide valuable insights into pedestrian behaviors and is critical to making decisions about campus designs and improvements. Conventional image or video based crowd-counting approaches have serious privacy risks, while cloud-based solutions also face challenges in providing real-time responses. This paper proposes Height-AWare Human Classifier for Crowd Counting (HAWC-CC), a real-time, cost-effective, and privacy-protected smart blue light pole-based crowd-counting framework for smart campuses, using Light Detection and Range (LiDAR) sensors and computing devices installed on the blue light poles to collect traffic data and analyze patterns in real time while protecting data privacy. HAWC-CC is built upon two novel methods. The first is an adaptive clustering approach that dynamically adjusts to point cloud structure and density for accurate and efficient clustering. The second method is a Height-AWare Human Classifier (HAWC) that projects 3D point clouds into height-augmented multiple 2D views and uses a lightweight CNN to detect humans from the projected views. HAWC-CC and its quantized version are then implemented on Nvidia Jetson and Coral TPU for real-world deployment on the blue light poles. A comprehensive 3D LiDAR dataset is collected and curated for single-person detection and crowd counting. An extensive evaluation shows that HAWC-CC significantly outperforms the representative related works (PointNet-CC, AutoEncoder-CC, and OC-SVM-CC) by up to 86.61% in Mean Absolute Error (MAE) and 90.44% in Mean Squared Error (MSE). Additionally, HAWC-CC is the only framework that meets the real-time requirements for LiDAR-based crowd counting, processing one LiDAR sample in 17.42 ms. HAWC demonstrates the highest robustness to limited training data, achieving a notable accuracy of 90.29% with only 0.1% of the training data. HAWC-CC outperforms SOTA RGB image-based solutions in high-density settings, achieving 97.64% accuracy.

***Index Terms*—Human Classification, Crowd Counting, Smart Campus, 3D Point Clouds**

I. Introduction

We present a smart blue light pole-based smart campus (Figure 1), where each pole is equipped with a variety of sensors and computing devices. The smart blue light poles are connected to the private campus network for secure communications with the private campus cloud backend. Among the many exciting smart campus applications that can be enabled by such a smart blue light pole-based infrastructure, in this work, we focus on real-time crowd counting. Crowd counting can provide valuable data on pedestrian behaviors, including popular routes, peak times, and common gathering areas. This information is critical to making decisions about campus design and management. Real-time crowd counting is essential for the instant detection of unusual crowding, indicating potential safety hazards or emergencies, and identifying crowded areas during emergencies, such as fires or security threats.

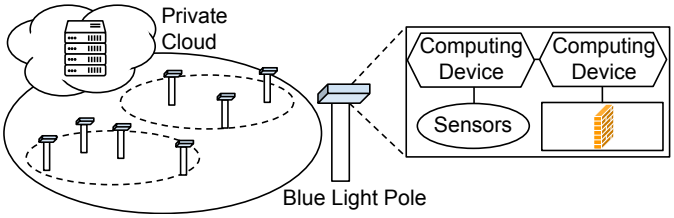


Fig. 1: Overall diagram of a smart blue light pole-based smart campus, each equipped with sensors and computing devices, communicating securely with a private cloud backend.

To develop crowd counting, we need to address the following challenges. 1) *Privacy Risks*: Conventional approaches based on camera images or videos [1]–[4] pose privacy risks due to potential exposure of human-identifiable information. Advanced vision algorithms can extract identifiable details, such as facial features, even from small images. Furthermore, the aggregation of camera data over time or across multiple locations can build detailed movement profiles of individuals, exposing sensitive personal information such as habitual routes and activity patterns, posing significant privacy risks [5]. 2) *Real-time requirements*: State-of-the-art camera-based approaches leverage computationally intensive architectures, such as transformer-based TransCrowd [1] and attention-based ASNet [3], to achieve high accuracy. While cloud can handle such computation, it is difficult to deliver real-time responses, and sharing camera data to cloud also introduces privacy risks.

We propose Height-AWare Human Classifier for Crowd Counting (HAWC-CC), leveraging the Light Detection and Ranging (LiDAR) sensors and computing devices on blue light poles to collect 3D traffic data and analyze patterns in real time while protecting privacy by keeping data local. *To the best of our knowledge, HAWC-CC is the first-of-its-kind real-time, cost-effective, privacy-protected crowd-counting system, successfully deployed on campus for practical use.*

First, HAWC-CC employs LiDAR-based human detection to perform privacy-preserving crowd counting. It leverages LiDAR sensors to generate 3D point clouds (set of points

with cloud-like structure), which reveal significantly less personally identifiable information (PII) than camera images [6]. Additionally, LiDAR provides precise object positions and shapes regardless of lighting conditions.

To enable large-scale campus deployment, HAWC-CC uses a cost-effective 32-channel LiDAR sensor [7]. These sensors produce fewer points than professional-grade sensors. The reduction in points is particularly pronounced with increasing distance due to the diminishing surface area for light reflection. To address this limitation, we propose a noise-controlled up-sampling method that preserves key human features for the subsequent CNN-based detection.

Second, HAWC-CC overcomes the challenges of variable point cloud density and efficient 3D point clouds processing for resource-constrained computing devices deployed on the blue light poles, enabling real-time LiDAR data processing and crowd counting. Specifically, HAWC-CC is built upon two novel methods. The first is an adaptive clustering approach that dynamically adjusts to point cloud structure and density for accurate and efficient clustering. The second is a Height-AWare Human Classifier (HAWC), which projects 3D point clouds into height-augmented multiple 2D views and uses a lightweight CNN to efficiently detect humans from these projected views.

Third, we collected and curated comprehensive 3D LiDAR datasets to address the scarcity of datasets from large outdoor environments. Unlike the abundance of datasets from RGB and RGB-D cameras, available 3D LiDAR datasets [8], [9] are limited and mainly collected in indoor environments for mobile service robots. Our dataset introduces unique properties that are valuable for 3D LiDAR-based human detection and crowd counting in outdoor environments. The first dataset contains exclusively single-person captures for evaluating single-person detection accuracy. The second dataset includes multiple-people captures for evaluating crowd counting accuracy. Each dataset consists of 15,028 LiDAR point clouds collected from 2021 to 2022.

Fourth, we implemented our proposed algorithms and models on two representative computing devices, Nvidia Jetson Nano and Coral TPU, and deployed them on blue light poles. To further improve the real-time performance of HAWC-CC, we employed post-training quantization to reduce model complexity. During deployment, we carefully monitored the environmental factors, such as temperature, to ensure reliable device operations.

Finally, we extensively evaluated the performance of HAWC-CC against three representative related frameworks: PointNet-CC [10], AutoEncoder-CC [11], and OC-SVM-CC [12]. HAWC-CC consistently achieves the highest accuracy for crowd counting with the lowest Mean Absolute Error (MAE) and Mean Squared Error (MSE), in both full precision and 8-bit quantization, outperforming the baselines by substantial margins of up to 86.61% in MAE and 90.44% in MSE. Additionally, HAWC-CC is the only framework that meets the real-time processing requirement of LiDAR-based crowd counting, processing the LiDAR samples at a rate of 17.42ms per sample. Furthermore, the proposed HAWC achieves a test accuracy of 99.97% for single-person human detection, significantly outperforming alternative classifiers used in the related frameworks by up to 51.37%. HAWC also demonstrates the highest robustness to limited training data, achieving a notable accuracy of 90.29% with only 0.1% of the training data, whereas PointNet's accuracy decreases to 75.82%, and AutoEncoder's even drops to 12.44%. HAWC-CC significantly outperforms SOTA RGB image-based methods in high-density settings by up to 20.54%, achieving an impressive 97.64% accuracy with an MAE of 5.9, compared to 90.9% by Su et al. [13], 77.1% by Liu et al. [14], and 86.27% by Hao et al. [15].

The paper is organized as follows: Section II introduces the background and related works; Section III describes the system overview; Sections IV and V explain the proposed adaptive clustering method and height-aware human classifier; Section VII presents the experimental results and discusses the real-world deployment issues; and Section VIII concludes the paper.

II. Background and Related Works

Outdoor crowd counting helps improve traffic control, road design, and facility management by distinguishing humans from other objects to understand traffic flow. Researchers have explored the following main crowd counting approaches.

A. Image-based Methods

Density estimation is the mainstream approach for image-based crowd counting, where a model generates a density map representing people density at each pixel, and the total count is obtained by summing pixel values. Techniques to improve density map accuracy include multi-column networks [16], which extract multi-scale features through parallel branches with different filter sizes, and attention mechanisms like ASNet [3], which dynamically selects the appropriate scale for each region. Recently, transformer-based methods [1] have further improved crowd counting performance.

However, these techniques face challenges: 1) loss of spatial information. Density maps lose precise localization of individuals; 2) lighting sensitivity. Density maps rely on images that degrade in poor lighting conditions; and 3) privacy concerns. Density maps based on images can reveal PII. In comparison, LiDAR retains spatial precision, overcomes lighting issues, and preserves privacy, making it a superior alternative for robust and privacy-preserving crowd counting.

B. Wi-Fi, Sonar, and Sound-based Methods

Wi-Fi-based methods [17] track personal devices, raising privacy concerns by collecting unique identifiers like MAC addresses, even if anonymized. Additionally, these systems offer lower resolution than LiDAR-based solutions. RF-based systems [18] rely on users carrying RF-ID tags, limiting their scalability for general crowd counting. Sonar-based methods [19] suffer from rapid attenuation in air and lack the fine spatial details of LiDAR, making them unsuitable for large outdoor spaces. Sound-based methods [20] are cost-effective but struggle with environmental noise and provide

lower accuracy and granularity than LiDAR, especially in outdoor environments.

C. LiDAR-based Methods

There is currently no cost-effective, LiDAR-only crowd-counting framework for outdoor environments like campuses, where challenges such as weather conditions, privacy concerns, costs, real-time requirements, and deployment complexities arise. HAWC-CC is the first-of-its-kind framework designed to address these issues and has been successfully deployed on campus for real-world use.

3D LiDAR-based methods leverage human classifiers for crowd counting. But designing a classifier that works well on real-life data and satisfies real-time needs for human detection is non-trivial. Human classifiers can be broadly categorized into two types: non-CNN-based and CNN-based classifiers.

Non-CNN-based Human Classifier. Schölkopf et al. [12] addressed human detection as a single-class classification problem and proposed the One-Class Support Vector Machine (OC-SVM) to solve it. Unlike conventional SVM, which maximizes separation between two classes, OC-SVM treats the origin in the projected space as an outlier and finds a hyperplane that maximizes separation from it.

CNN-based Classifier. CNN-based classifiers can be classified into two categories based on their techniques for handling 3D LiDAR point clouds. *2D-CNN-based Methods* convert 3D point clouds into 2D views, e.g., bird-eye-view (BEV) [21], range-view (RV) [22], and Density-SetAbstraction (Density-SA) [23]. These views are then classified using a 2D CNN. However, these projection methods fail to capture the important spatial details. For example, BEV [21] lacks vertical information, making it less effective for small objects like pedestrians, and often relies on other modalities, such as images, to provide additional information [24]. Existing multi-view methods [22], [23] address this by generating several 2D views to improve spatial coverage. However, these views are typically predefined and do not emphasize the semantic importance of features for human detection. In comparison, we argue that height is the most critical feature for human detection and propose a height-aware projection method that augments height information in the views. Our evaluation shows that the proposed method outperforms all these alternatives (see Section VII-B).

3D-CNN-based Methods do not have the projection issue as they directly process 3D point clouds. Qi et al. [10] designed PointNet, the state-of-the-art neural network for classifying 3D point clouds, leveraging read-process-write networks with attention mechanisms to consume unordered input [25]. However, PointNet is too large and too slow to run on edge devices, and it is not robust to limited training data (see Section VII-B for details). In comparison, our proposed HAWC is more accurate, robust, and much faster than PointNet.

III. System Overview

LiDAR-integrated smart blue light pole infrastructure. Figure 2 shows a smart blue light pole with a LiDAR sensor mounted atop the blue alarm light. The pole's compartment

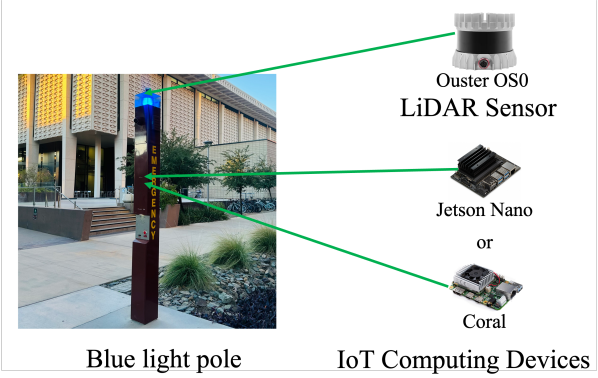


Fig. 2: Blue light pole with LiDAR sensor and computing devices.

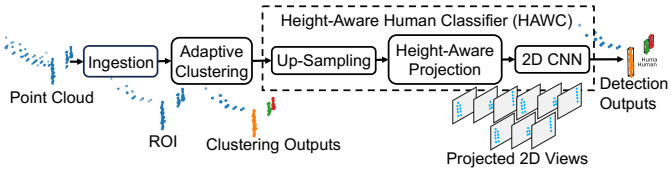


Fig. 3: Pipeline overview of HAWC-CC for crowd counting.

houses computing devices and protects them from heat and rain. To balance cost and performance, we select the Ouster OS0 32-channel LiDAR sensor [7] for data capture. For computing, we employ the Coral Dev Board and Nvidia Jetson Nano, both widely used lightweight Internet-of-Things (IoT) devices. The LiDAR sensor captures real-time 3D point cloud data from passersby, which is securely transmitted to the computing devices housed in the pole's compartment for crowd counting.

3D LiDAR data capture. HAWC-CC defines its region of interest (ROI) and captures three-dimensional points $p_i(x_i, y_i, z_i) \in \mathbb{R}^3$ within it. The x-axis of the ROI restricts the distance from 12 meters to 35 meters from the LiDAR sensor, ensuring that individuals within this range are captured without being affected by shadows cast by the pole. Meanwhile, individuals beyond 35 meters are excluded since they produce weak reflective signals. The y-axis covers the entire 5-meter-wide walkway, providing comprehensive coverage for data collection within the ROI. Instead of collecting point cloud data from a full 360-degree LiDAR scan, HAWC-CC targets approximately 90 degrees, specifically targeting the walkways connecting popular campus areas. This targeted approach discards irrelevant data, improving processing efficiency and accuracy by concentrating only on the most frequently used paths.

Ingestion of high-quality 3D point clouds. HAWC-CC improves the quality of captured 3D point clouds by removing the noise from ground reflections or objects beyond the sensor's effective range, which can degrade classifier performance in real-world LiDAR data. HAWC-CC employs a rule-based ground segmentation technique to reduce z-axis noise, targeting ground noise from objects like pulleys commonly found on campus. Let z_i denote a point's z-axis coordinate and z_{min} the minimum z value. The LiDAR sensor, mounted on the top of a three-meter-tall smart blue light pole, has a detection range of 0

to -3 meters along the z-axis. Empirical observations indicate that ground noise extends up to 0.4 meters. Thus, HAWC-CC retains the points $P_i = \{(x_i, y_i, z_i) \in \mathbb{R}^3 | i = 1, \ldots, N; z_i \geq z_{min}\}$ and sets z_{min} to -2.6 meters.

Processing 3D point clouds for crowd counting. Figure 3 provides an overview of HAWC-CC. First, it filters ground-reflected noise in the point clouds. Second, it partitions the point clouds into distinct clusters using the proposed adaptive clustering method. Third, the height-aware human classifier (HAWC) processes these clusters to classify them as "Human" or "Object". The crowd count equals the total number of clusters identified as "Human". For fast processing and low memory usage, the convolutional network within HAWC is quantized to 8-bit using post-training quantization. The following sections will further discuss each step in detail.

IV. Adaptive Clustering

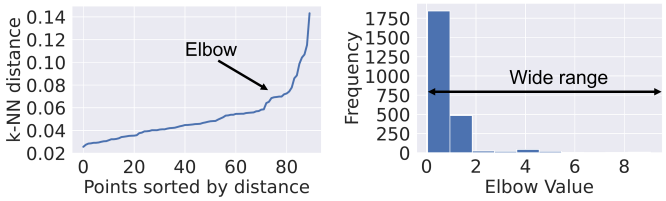
A core mechanism of HAWC-CC is to partition the point clouds into distinct clusters. The mechanism serves two purposes: 1) noise reduction—even after preprocessing, the point cloud data may still contain noise, and clustering helps reduce this noise by grouping similar data points; 2) handling multiple objects of interest. Clustering is essential for scenes with multiple objects, as it helps separate each object in the point cloud, facilitating the detection.

Clustering different instances of humans. HAWC-CC employs the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm to group point clouds belonging to different human instances. DBSCAN structures the data to represent the underlying density accurately [26], ensuring that points belonging to the same human instance are clustered together. We also tested other popular clustering methods, e.g., hierarchical clustering [27], Gaussian mixture clustering [28], and k-means clustering [29], but found them less favorable. Gaussian mixture clustering and k-means clustering assume a parametric distribution and typically create clusters with convex shapes, which are less suitable for capturing complex, non-convex patterns in the data. Hierarchical clustering often attributes bounding boxes of the same object to separate clusters. In contrast, DBSCAN can detect clusters with arbitrary shapes, including non-convex shapes, and effectively handle low-density regions by classifying them as outliers [30].

Adaptive clustering. As a prerequisite for applying DBSCAN, HAWC-CC must specify ϵ, which defines the neighborhood range for each point. A good ϵ is crucial for accurate clustering, but selecting an appropriate value is challenging due to the lack of automatic method.

To determine ϵ, we plot k-nearest neighbor distances in decreasing order and identify the "elbow" point, indicating the transition from points within clusters (with smaller distances) to noise points (with larger distances). The plot's "elbow" point denotes this sample's optimal ϵ. Figure 4a shows the k-nearest neighbor distance plot for a single training sample and its "elbow" point indicates an optimal ϵ value of 0.069.

Due to the unstructured and unordered nature of point clouds, each sample may have a distinct optimal ϵ. Our analysis of



(a) k-NN distance plot. (b) Elbow values distribution.

Fig. 4: Illustration of elbow point and the wide range of its value.

optimal ϵ values across the training set (Figure 4b) reveals a wide range from 0.04 to 9.06, with 0.08 predominating. This variability poses challenges, as using a fixed ϵ can result in suboptimal performance. Furthermore, the range of ϵ may exceed that observed in the training set when processing new data after HAWC-CC deployment.

To address the aforementioned problems, we propose adaptive clustering approach that dynamically calculates the optimal ϵ for each LiDAR capture by identifying the "elbow" point in the k-nearest neighbor plot. This optimal ϵ is then applied in the DBSCAN to improve clustering quality. Specifically, HAWC-CC first calculates the distances between each point p_i and its nearest neighbors, and then sorts these distances in ascending order: $D_i = \{d_{i,k}\}_{k=1}^{k=n}$, where $d_{i,k}$ is the distance between point p_i and its k-th nearest neighbor, and n denotes the number of nearest neighbors.

Second, HAWC-CC performs the KneeLocator algorithm on the sorted distance vector D_i to determine the "elbow" point as follows: $k_{\text{elbow}} = \arg\max_i \left(\frac{d_{i+1,k} - d_{i,k}}{d_{i,k}} \right)$. The optimal ϵ is determined as the distance value at this point: $\epsilon_{\text{optimal}} = d_{k_{\text{elbow}},k}$. Finally, HAWC-CC identifies core points C as those having at least m neighbors within the $\epsilon_{\text{optimal}}$ range, as follows: $C = \{p_i \mid |\{p_j \mid \text{distance}(p_i, p_j) \leq \epsilon_{\text{optimal}}\}| \geq m\}$.

HAWC-CC forms the final clusters by iterating through the points and assigning them to clusters based on their connectivity to core points. A point p_i belongs to cluster C_m if it is a core point or a neighbor of a core point within the $\epsilon_{\text{optimal}}$ range.

In summary, the proposed adaptive clustering method dynamically adjusts to the inherent structure and density of point clouds, resulting in more accurate clustering compared to both fixed-ϵ density-based clustering and other non-density-based methods (see Section VII-C for comparison results).

V. Height-Aware Human Classifier (HAWC)

A core component of HAWC-CC is classifying the 3D point clouds produced by adaptive clustering. The classification model should be: 1) lightweight to accommodate the resource constraints; 2) capable of real-time classification for smart campus applications; and 3) robust with limited labeled data.

Given these challenges, we propose a 2D-CNN-based classifier as an suitable choice for classifying 3D point clouds on edge devices. While non-CNN classifiers are lightweight, easy to train, and suitable for learning from scarce data, they rely on hand-crafted features, which is challenging to find. In contrast, CNNs directly learn features from raw data, demonstrating remarkable success across domains. To classify

3D point clouds, CNN-based classifiers either 1) process point clouds with 3D models like PointNet or 2) convert 3D point clouds into 2D views and process them with 2D CNNs. 3D models preserve all features and can achieve good accuracy, but they require extensive training data for effective feature extraction due to the sparse and unstructured nature of 3D point clouds. Processing in 3D space also demands significantly more computational power than 2D, often exceeding edge devices capabilities. Transformer-based and hybrid models face similar limitations in terms of complexity and latency. Therefore, we argue that a 2D-CNN-based approach can more effectively handle the challenges of edge resource constraints, real-time processing, and limited labeled data.

We propose a novel 2D-CNN-based classification method, **H**eight-**AW**are Human **C**lassifier (HAWC) to enhance the 3D point cloud classification. Specifically, first, HAWC employs the proposed noise-controlled up-sampling approach to standardize the size of 3D point clouds. It then uses a new height-aware projection method to transform 3D point clouds into multiple height-augmented 2D views. Finally, HAWC classifies these 2D views through a lightweight convolutional network.

Noise-controlled up-sampling. Given the challenge of identifying point significance and its potential impact on classification, we address CNNs' fixed-size input requirement for variable-sized 3D point clouds by standardizing point counts and employing a novel noise-controlled up-sampling approach.

Let $P_i = \{(x_j, y_j, z_j)\}_{j=1}^{j=N_i} \in \mathbb{R}^3$ represent points in the ith point cloud, where N_i is the number of points, and $\mathbb{R}^3$ represents the three-dimensional Euclidean space with each point (x_j, y_j, z_j) specifies its x-, y-, and z-coordinates. To determine the fixed-size input, HAWC first calculates the maximum number of points across all the point clouds in the training dataset: $N_{max} = max(\{N_i\}_{i=1}^{M})$, where M is the total number of point clouds in the entire training dataset. HAWC then calculates the up-sampling size N'_{max}, ensuring that it is slightly larger than N_{max} and a perfect square: $N'_{\text{max}} = \lceil \sqrt{N_{\text{max}}} \rceil^2$, meeting CNNs' square image requirement.

To maintain a consistent count of N'_{max} points across all training samples, we introduce a controlled noise level to each sample. Instead of sampling noise points from commonly used distributions such as Gaussian, we propose to sample the noise points from a distinct "Object" dataset, where humans are not present in the scene. Thus, to standardize the number of points to N'_{max} for the ith point cloud, HAWC samples additional points $Q_i = \{(x_k, y_k, z_k)\}_{k=1}^{k=N'_{max}-N_i} \in \mathbb{R}^3$ from the object data. Finally, HAWC produces the ith up-sampled point cloud: $P'_i = P_i \cup Q_i = \{(x_j, y_j, z_j)\}_{j=1}^{j=N'_{max}} \in \mathbb{R}^3$.

Figure 5 illustrates the noise-controlled up-sampling process for a single point cloud. The left figure shows a point cloud of a single human, the middle figure shows a point cloud from "Object" data, and the right figure shows the up-sampled point cloud. In practice, all "Object" data are pooled together, and the required number of point clouds are randomly selected from this pool to up-sample each "Human" point cloud. The proposed noise-controlled up-sampling approach effectively reduces the impact of noise on the learning process of HAWC and outperforms Gaussian distribution sampling (see Sec VII-B for results).

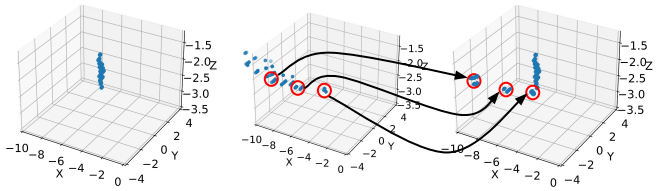
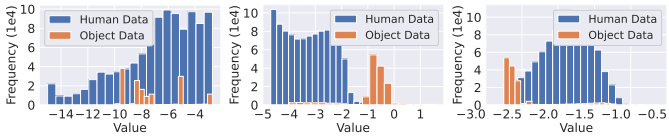


Fig. 5: Noise-controlled up-sampling: left shows a single human point cloud, the middle shows "Object" data, and the right shows the up-sampled point cloud.



(a) x-axis. (b) y-axis. (c) z-axis.

Fig. 6: Histogram of "Human" and "Object" data on (a) x-axis, (b) y-axis, and (c) z-axis.

To validate the low impact of additional noise on classification accuracy, we compare the distribution patterns of "Human" and "Object" data. Figure 6 shows histogram patterns for point cloud coordinates across all three axes (x, y, z). This visualization demonstrates that the "Human" data exhibits unique patterns that are markedly different from those of the "Object" data. The controlled noise from "Object" data is unlikely to compromise classification accuracy. The proposed noise-controlled up-sampling approach effectively reduces the noise impact of HAWC and outperforms Gaussian distribution sampling (see Sec VII-B for results).

Height-aware projection (HAP). After noise-controlled up-sampling, the point clouds attain a consistent size that can be converted into square views. In this section, we propose a simple yet effective height-aware projection (HAP) method to convert 3D point clouds into multiple 2D views.

First, HAP projects 3D point clouds into 2D space by directly generating three 2D views, including front view, top view, and side view, each with a dimension of $N'_{\text{max}} \times 2$. Specifically, for the ith point cloud, the xy plane projection generates the top view $P_i^{top} = \{p_i^j(x_j, y_j)\}_{j=1}^{j=N'_{\text{max}}} \in \mathbb{R}^2$, the yz plane projection produces the front view $P_i^{front} = \{p_i^j(y_j, z_j)\}_{j=1}^{j=N'_{\text{max}}} \in \mathbb{R}^2$, and the xz plane projection generates the side view $P_i^{side} = \{p_i^j(x_j, z_j)\}_{j=1}^{j=N'_{\text{max}}} \in \mathbb{R}^2$. HAP employs direct projection instead of the commonly used occupancy grid [31], since the occupancy grid is effective only with a large number of points. With fewer points, the occupancy grid struggles to distinguish valuable patterns from background noise.

Then, HAP calculates the variation in height (z-coordinate) among neighboring points in the 3D point cloud, as height is a significant feature for distinguishing human patterns. Specifically, for the i-th point cloud $P_i = \{p_i^j(x_j, y_j, z_j)\}_{j=1}^{j=N'_{\text{max}}} \in \mathbb{R}^3$, it first constructs a KD-tree $KDTree(P_i)$ to facilitate fast nearest-neighbor searches. Then, for each j^{th} point p_i^j

in P_i, it identifies the k-nearest neighbors through a single query to the KD-tree: $K_i^j = Query(KDTree(P_i), p_i^j, k)$, producing $K_i^j = \{p_i^j(x_j, y_j, z_j)\}_{j=1}^{j=k} \in \mathbb{R}^3$. Next, it extracts the z-coordinates of the neighbors K_i^j, generating a set of z-coordinates: $\{z_i^j\}_{j=1}^{j=k} \in \mathbb{R}^1$. After that, it calculates the height variation σ_i^j for the j-th point of i-th point cloud, as the standard deviation of these z-coordinates along the neighbors: $\sigma_i^j = \sqrt{\frac{1}{k}\sum_{j=1}^{k}(z_i^j - \bar{z}_i)^2}$, where k is the number of neighbors, $\bar{z}_i$ is the mean of the z-coordinates for the j-th point's neighbors.

Next, HAWC incorporates height variations into the projected views to augment the height features. Specifically, it stacks the height variations $\{\sigma_i^j\}_{j=1}^{j=N'_{\text{max}}} \in \mathbb{R}^1$ into the top view $P_i^{top} = \{p_i^j(x_j, y_j)\}_{j=1}^{j=N'_{\text{max}}} \in \mathbb{R}^2$, producing a height-integrated top view $P_i^{top'} = \{p_i^j(x_j, y_j, \sigma_i^j)\}_{j=1}^{j=N'_{\text{max}}} \in \mathbb{R}^3$. It then utilizes a reshape function to transform the top view from a size of $N'_{\text{max}} \times 3$ to $D \times D \times 3$, where $D = \sqrt{N'_{\text{max}}}$. Similarly, it also transforms the other two views from a size of $N'_{\text{max}} \times 2$ to $D \times D \times 2$. Finally, following the approach of stacking RGB channels in images, HAWC stacks all three projections to create a single input image of dimension $D \times D \times 7$, serving as the input for the 2D CNN model.

2D CNN. HAWC processes the stacked projections using a lightweight CNN model with three convolutional layers and two fully connected layers, which has a total of 62,114 parameters. Each convolutional layer includes batch normalization and a ReLu activation, using 3×3 kernel and a stride of 1.

VI. Edge Deployment

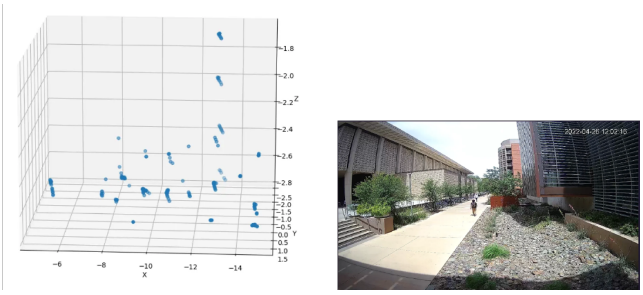
We deployed the proposed models on smart blue light poles for real-time and privacy-preserving crowd counting. We considered two complementary devices: the Coral Dev Board, providing an edge TPU for accelerating AI applications but with limited 1GB memory, and the Nvidia Jetson Nano, providing the Nvidia Maxwell GPU and 4GB of memory.

We used TensorFlow Lite on both devices for inference with TFLite models. We employed TensorFlow Lite Converter [32] to apply post-training quantization, converting floating-point models into integer format and reducing complexity for real-time inference on resource-constrained devices. Quantization is essential for accelerating the model on the Coral Dev Board's TPU, which supports only 8-bit integer models. To preserve accuracy, the converter calibrates the quantization range using a calibration dataset. We randomly selected 100 samples from our training data to serve this purpose.

VII. Evaluation

A. Methodology

3D LiDAR dataset. A good 3D LiDAR dataset for crowd counting should: 1) consist of clear cloud points from LiDAR sensor; 2) originate from real-world deployments, capturing spatiotemporal patterns of humans, and aligning with the deployment setup; 3) provide annotated human labels for accuracy evaluation; and 4) include metadata such as timestamps and sensor positions to support the analysis of dynamic crowd behaviors over time.



(a) Single-person LiDAR capture. (b) Single-person camera capture.

Fig. 7: Example visualization of captures from LiDAR and image sensor. These sensors face the same direction at a similar height.

However, existing public outdoor 3D point cloud datasets for crowd counting are limited and do not meet the above requirements. To address this gap, we collected and curated two LiDAR datasets using data captured from smart blue light poles deployed across the campus (see Section III). Each dataset contains 15,028 LiDAR samples collected over one year (2021 to 2022). We also collected camera images for ground truth validation, but we did not use camera data for inference to provide strong privacy protection. The first dataset exclusively contains single-person captures for evaluating single-person detection. We created the ground truth labels for the LiDAR samples using a Lasso selector to identify the most distinctive human pattern in each point cloud and verified them using the associated RGB images (see Figure 7). The second dataset includes multiple-person captures for evaluating crowd counting. We created the ground truth labels by manually counting humans in each LiDAR capture and verifying the counts using the associated RGB image. Each LiDAR sample, i.e., one point cloud, has a size of 324×3. We applied a random 80:20 training and test split across LiDAR samples.

Accuracy metrics. We measured the accuracy of both HAWC for single-person detection and HAWC-CC for crowd counting. For single-person detection, we evaluated HAWC's test accuracy, precision, recall, and F1 score. For crowd counting, we measured HAWC-CC's Mean Absolute Error (MAE) and Mean Squared Error (MSE), following the methodology of previous works on image-based crowd counting [2], [4]. For a sequence of N LiDAR point clouds, MAE is defined as follows: $MAE = \frac{1}{N}\sum_{i=1}^{N}\left|C_i - C_i^{GT}\right|$, where C_i represents the crowd count predicted by the model, and C_i^{GT} is the crowd count derived from the human-labeled ground- truth. MSE serves as a complementary metric to MAE and indicates the robustness of the predicted count [2], defined as follows: $MSE = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(C_i - C_i^{GT})^2}$.

Speed metric. For inference speed, we measured the end-to-end inference time to process a single LiDAR sample, reflecting real-world scenarios with continuous streaming data.

Baselines. For human detection, we compared HAWC to three prior works: PointNet [10], AutoEncoder [11], OC-SVM [33]. These designs cannot support crowd counting as they lack our

data processing and adaptive clustering techniques. In order to evaluate their performance for crowd counting, we integrated them into HAWC-CC by replacing HAWC and adding steps (e.g., feature extraction, up-sampling), resulting in PointNet-CC, AutoEncoder-CC, and OC-SVM-CC.

- *PointNet-CC*: Unlike HAWC, which converts 3D point clouds into multiple 2D views before classification, PointNet-CC directly process 3D point clouds using the state-of-the-art model, PointNet [10]. To handle PointNet's fixed-size input requirement, PointNet-CC uses our up-sampling step to increase the number of points in each point cloud generated from adaptive clustering. We used the original PointNet implementation, consisting of 64 layers and 747,947 parameters. It includes a classification network, which transforms inputs and aggregates features by max pooling, and a segmentation network, which concatenates global and local features for per-point scores.
- *AutoEncoder-CC*: AutoEncoder-CC performs feature extraction after adaptive clustering to obtain meaningful features, e.g., boundary regularity and circularity, for human detection [34]; it then utilizes AutoEncoder [11] for classification. The feature extraction divides each point cloud into slices (0.2-meter intervals, approximating human head length), and extract features from each slice. The AutoEncoder comprises a three-layer encoder, a bottleneck layer, a three-layer decoder, and an output layer. Following Liou et al [11], we used KeraTuner [35] to optimize number of neurons in the model with a grid search (16 to 128 neurons per layer), resulting in a model with 26,384 parameters.
- *OC-SVM-CC*: Similar to AutoEncoder-CC, OC-SVM-CC also performs feature extraction following adaptive clustering and then it utilizes OC-SVM [33] for classification. We considered OC-SVM because of its popularity in two-class nonlinear classification tasks, treating the origin of the high dimensional space as the only member of the second class.

Implementation details. We implemented HAWC-CC and all baselines using TensorFlow 2.12.0 and Python version 3.9.16. HAWC uses the Adam optimizer with a learning rate of 0.001 and a batch size of 32. PointNet and AutoEncoder use the Adam optimizer with batch sizes of 64 and 512, respectively, and a learning rate of 0.001. OC-SVM employs a Radial Basis Function kernel with a coefficient of $1/n$, where n is the number of input features. Both the training errors upper bound and the support vectors lower bound are set to 0.01.

B. Significance of Height-Aware Human Classifier

Height-**AW**are Human **C**lassifier (HAWC) is designed to distinguish between humans and objects, which is critical to the performance of crowd counting. In this section, we conduct a comprehensive evaluation of HAWC in comparison to alternative classifiers, including PointNet [10], AutoEncoder [11], and OC-SVM [33]. Specifically, we compare their accuracy, robustness to limited training data, and inference speed. Additionally, we perform an in-depth analysis of the effectiveness of HAWC's key components, such as noise-controlled object data sampling and height-aware projection.

Accuracy. Table I illustrates the test accuracy comparison for single-person classification between HAWC and the baselines in full-precision (32-bit floating points) and quantized 8-bit formats. In full precision, HAWC performs the best across all accuracy metrics, including test accuracy, F1 score, precision and recall. Specifically, HAWC achieves the highest test accuracy of 99.97%, significantly outperforming the baselines by 5.06% to 51.37%. Moreover, HAWC achieves F1 score, precision, and recall of 1.0, reflecting its exceptional ability to distinguish between humans and objects without any errors.

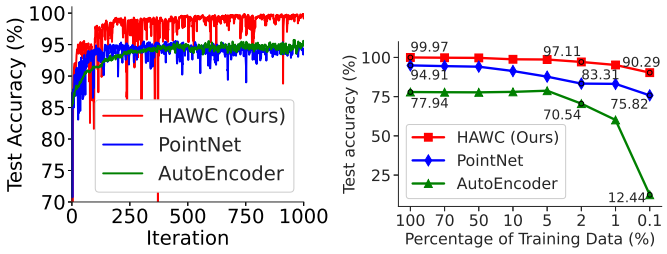
After 8-bit quantization, HAWC maintains the highest accuracy among all the baselines. Specifically, 8-bit HAWC achieves 99.53% test accuracy, outperforming 8-bit PointNet and 8-bit AutoEncoder by 9.94% and 26.18%, respectively. HAWC also shows the least accuracy degradation from quantization. Quantization reduces HAWC's test accuracy by only 0.44%, significantly lower than the accuracy loss observed in PointNet (5.32%) and AutoEncoder (4.59%). OC-SVM is excluded from the quantized accuracy comparison because its reliance on support vectors and kernel methods, which require precise positioning and transformations, makes it incompatible with reduced bit widths. It is noteworthy that the impact of quantization on networks for LiDAR-based 3D classification tasks is recognized as a significant challenge [36]. Our work is among the first to provide a comprehensive analysis of the state-of-the-art methods, and our proposed classification method can achieve the lowest and negligible quantization accuracy loss.

As observed, PointNet performs worse than HAWC, with lower test accuracy (94.91% vs. 99.97%) and F1 scores (0.89 vs. 1.0) in full precision, and lower test accuracy (89.59% vs. 99.53%) in 8 bitwidth, despite its state-of-the-art accuracy in complex 3D tasks [10]. Its complex architecture, tailored for multi-category object classification and detailed segmentation, leads to overfitting in binary classification tasks, reducing its generalization capability. Moreover, PointNet's direct processing of raw 3D point clouds results in a high-dimension input space that introduces noise and redundancy. In contrast, HAWC's height-aware 2D projections effectively filter noise and preserves human-related features, achieving superior detection accuracy. Notably, OC-SVM performs the worst, with a test accuracy of 48.6% and an F1 score of 0.49, as it misclassifies every test LiDAR sample as "human". This result indicates that 3D point cloud-based classification tasks are exceedingly challenging for SVM-based methods. Figure 8a illustrates the test accuracy progression of all the models. HAWC shows more fluctuation across iterations than the baselines due to learning from multiple views of each LiDAR capture, which introduces view-specific variability; however, its consistently strong performance on unseen test data demonstrates its robustness.

Robustness to limited training data. Figure 8b shows the test accuracy of HAWC, PointNet, and AutoEncoder trained on varying percentages of the training data, from 100% to 0.1%. HAWC demonstrates the highest robustness to limited training data, achieving an accuracy of 99.97% with the full training dataset and a notable 90.29% with only 0.1% of the training

TABLE I: Accuracy comparison for single-person human detection between HAWC and the baselines in full-precision and quantized 8-bit formats. We do not include OC-SVM for quantized accuracy comparison (denoted with "-") since its reliance on support vectors and kernel methods makes it incompatible with reduced bit widths.

Model	FP32				Int8	
	Test Acc. (%)	F1	Precision	Recall	Test Acc. (%)	Test Acc. Diff. (%)
OC-SVM	48.60	0.49	0.47	0.50	-	-
AutoEncoder	77.94	0.49	0.88	0.87	73.35	- 4.59
PointNet	94.91	0.89	0.85	0.77	89.59	- 5.32
HAWC (Ours)	**99.97**	**1.00**	**1.00**	**1.00**	**99.53**	**- 0.44**



(a) 100% training data. (b) Varying percentage of training data.

Fig. 8: Test accuracy of HAWC with baselines on (a) 100% training data and (b) varying percentages of the training data.

data. PointNet's accuracy decreases from 94.91% to 75.82%, while AutoEncoder shows significant sensitivity, dropping from 77.94% to 12.44%. These results highlight HAWC's superior robustness to limited training data.

Inference speed. Table II shows the inference speed of HAWC, PointNet, AutoEncoder, and OC-SVM in both full precision (32-bitwidth) and 8-bit quantization on the Coral Dev Board and Nvidia Jetson Nano. On the Jetson Nano, HAWC achieves faster inference speed than PointNet by 23× in floating points and by 37× in 8-bit integers. Moreover, although AutoEncoder performs faster than HAWC. e.g., 0.03 ms vs. 0.29 ms in 8-bit, this comes at the expense of significantly lower accuracy. Specifically, the test accuracy of 8-bit AutoEncoder is 73.35%, lower than the 8-bit HAWC by 26.18%, as shown in Table I. Furthermore, quantization enables HAWC to achieve the highest speedup of 1.87×, outperforming AutoEncoder at 1.62× and PointNet at 1.13×.

On the Coral Board, with 8-bit quantization, HAWC achieves the lowest inference speed of 0.62 ms, outperforming AutoEncoder and PointNet by 1.76× and 1.69×, respectively. Meanwhile, 8-bit HAWC achieves the highest test accuracy of 99.53%, as shown in Table I. These results validate that HAWC is both the fastest and the most accurate model.

On the Coral Board, the 8-bit AutoEncoder surprisingly has a longer inference time than its floating-point counterpart, whereas 8-bit HAWC achieves a significant 3.05× speedup. This is because the edge TPU on the Coral Board is optimized for operations such as convolutions and pooling but it cannot handle fully connected layers, heavily used in AutoEncoder, efficiently. In contrast, the Jetson Nano avoids this issue with its general-purpose GPU and libraries like CUDA and cuDNN

TABLE II: Inference time of full-precision and 8-bit HAWC and baselines for processing a single LiDAR sample on the Coral Dev Board and Nvidia Jetson Nano ("±" indicates standard deviation).

Edge Device	Model	Inference Time (ms)		Speedup
		FP32	Int8	
Jetson Nano	OC-SVM	0.30 ± 0.01	-	-
	Autoencoder	0.04 ± 0.02	0.03 ± 0.01	1.62 ×
	PointNet	12.15 ± 1.51	10.75 ± 0.74	1.13 ×
	HAWC (Ours)	0.54 ± 0.20	0.29 ± 0.05	1.87 ×
Coral Dev Board	OC-SVM	0.32 ± 0.03	-	-
	Autoencoder	0.07 ± 0.04	1.05 ± 0.19	0.07 ×
	PointNet	57.14 ± 14.63	1.09 ± 0.78	52.33 ×
	HAWC (Ours)	1.88 ± 1.33	0.62 ± 0.48	3.05 ×

TABLE III: Test accuracy comparison between object data sampling and Gaussian distribution sampling with varied standard deviation σ.

Sampling Method	Object data sampling	Gaussian distribution sampling		
		σ=3	σ=5	σ=7
Test Accuracy (%)	**99.97**	99.70	94.30	97.15
Accuracy Difference (%)	0	-0.27	-5.67	-2.82

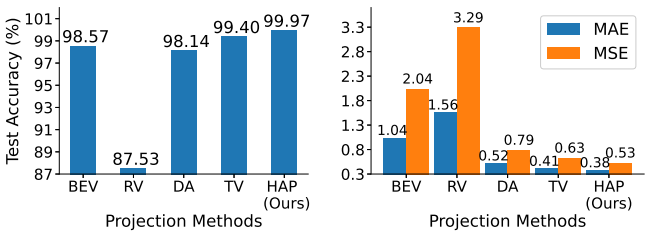
which support a broad range of operations. This observation shows that CNN-based frameworks for processing 3D cloud points are well-suited for deployment on typical edge hardware.

Object data sampling. Point clouds exhibit diverse sizes, whereas CNNs require fixed-size inputs. To address this, HAWC up-samples the point clouds to a fixed size through object data sampling, which selects points from the "Object" dataset consisting of scenes without humans. Alternatively, Gaussian distribution sampling generates synthetic points with a fixed mean $\mu = 0$ and varying standard deviations σ, e.g., $\sigma \in \{3, 5, 7\}$. As shown in Table III, object data sampling outperforms Gaussian distribution sampling by up to 5.67%.

Height-aware projection (HAP). HAWC uses HAP to convert 3D point clouds into multiple 2D views. To evaluate the effectiveness of HAP, we compare HAWC's test accuracy for human detection and HAWC-CC's accuracy for crowd counting against four alternative projection methods: bird-eye-view (BEV) [21], range-view (RV) [22], density-aware (DA) [23], and three-view (TV) projections, a simplified version of HAP without height integration.

Figure 9 demonstrates that HAP enables HAWC and HAWC-CC to achieve significantly higher accuracy than other projection methods. Specifically, for human classification, HAP outperforms all baselines by up to 12.44%, and for crowd counting, it achieves 7.32% to 75.61% lower MAE and 15.87% to 83.88% lower MSE.

HAP's superior performance stems from its enhanced utilization of height information, which improves differentiation between overlapping objects at different elevations and mitigates occlusion issues. It achieves a balanced representation of height, width, and depth, leading to higher spatial resolution and more accurate crowd localization. Unlike bird-eye-view, range-view, and three-view, which lose crucial height information, and density-aware projection, which loses spatial details, HAP integrates height directly into the projection process, reducing



(a) HAWC (b) HAWC-CC

Fig. 9: (a) Test accuracy of HAWC and (b) MAE and MSE of HAWC-CC using our height-aware projection (HAP) and alternative methods, including bird-eye-view (BEV), range-view (RV), density-aware (DA), and three-view (TV) projections.

TABLE IV: Accuracy of HAWC-CC with different clustering methods. The number inside the bracket represents the relative improvement of the proposed adaptive clustering over those methods.

Method		MAE	MSE
Fixed-ϵ Clustering	0.1	1.56 (75.57%)	3.28 (83.85%)
	0.3	0.67 (43.03%)	1.04 (49.22%)
	0.5	0.40 (4.63%)	0.55 (3.49%)
	0.7	0.42 (9.82%)	0.63 (15.46%)
	0.9	0.45 (15.1%)	0.71 (25.4%)
Hierarchical Clustering		134.7 (99.72%)	28236.72 (100%)
Adaptive Clustering		**0.38**	**0.53**

ambiguity and providing a more comprehensive understanding of crowd distributions.

C. Significance of Adaptive Clustering

Table IV shows that HAWC-CC achieves the highest accuracy (lowest MAE and MSE) with the proposed adaptive clustering compared to fixed-ϵ clustering and hierarchical clustering. Specifically, adaptive clustering outperforms fixed-ϵ clustering with $\epsilon \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ by 4.63% to 75.57% in MAE and 3.49% to 83.85% in MSE. Hierarchical clustering is less favorable as it often attributes bounding boxes of the same person to separate clusters, significantly overestimating crowd size. This observation aligns with Xu et al.'s findings [37], where hierarchical clustering performed poorly for Object Re-identification (ReID) in city-scale edge camera networks. These results confirm the effectiveness of the proposed adaptive clustering, which adaptively selects the optimal ϵ for each 3D point cloud.

D. Crowd Counting Performance

Accuracy. Table V shows the accuracy of full-precision (32-bit floating points) and 8-bit quantized HAWC-CC compared to the baseline crowd counting frameworks. HAWC-CC consistently achieves the highest accuracy, i.e., the lowest MAE and MSE, outperforming the baselines by substantial margins, up to 86.61% in MAE and 90.44% in MSE. Specifically, in full precision, HAWC-CC outperforms 1) PointNet-CC by 39.68% and 45.92%, 2) AutoEncoder-CC by 11.63% and 32.05%, and 3) OC-SVM-CC by 86.61% and 90.44%, in MAE and MSE, respectively. After quantizing all the classifiers to 8-bit, HAWC-CC still outperforms 1) PointNet-CC by 73.72% and 83.03%, and 2) AutoEncoder-CC by 43.91% and 64.22%, in MAE and MSE, respectively. OC-SVM-CC is excluded from quantized accuracy comparison since its reliance on support vectors and kernel methods makes it incompatible with reduced bitwidths.

As shown in Table V, HAWC-CC also demonstrates remarkable robustness to quantization, exhibiting the smallest increments in MAE and MSE among all the evaluated frameworks. For example, quantization increases HAWC-CC's MSE by only 0.03, significantly lower than PointNet-CC's 2.32 and AutoEncoder-CC's 0.79. These results confirm that HAWC-CC is the most accurate and robust framework in both full and reduced precisions.

Speed. Table V shows the crowd counting speed of HAWC-CC and baseline frameworks. Speed is measured as the inference time of the entire system for processing a single LiDAR sample on Nvidia Jetson Nano. Consider a 60 frames-per-second sensor, each frame must be processed within 16 ms to meet real-time requirements. HAWC-CC is the only framework that closely meets this requirement. Specifically, HAWC-CC processes one LiDAR sample in 17.42 ms, faster than PointNet-CC by 1.5× (26.25 ms) and AutoEncoder-CC by 2.7× (46.98 ms).

Although AutoEncoder-CC uses a lightweight classifier AutoEncoder, it is slower than HAWC-CC due to its time-consuming feature extraction. PointNet-CC is also slower than HAWC-CC because it uses PointNet, a computationally expensive model that directly processes raw 3D point clouds. Moreover, PointNet's feature transformations and max pooling operations, although effective for capturing 3D structures, further increase the computational burden. In comparison, HAWC-CC converts the complex 3D point clouds into 2D views and processes them with a lightweight classifier HAWC, which has fewer parameters and lower computational requirements.

In summary, HAWC-CC achieves the highest accuracy and fastest speed among all the baselines, enabling accurate and real-time crowd counting. *To the best of our knowledge, HAWC-CC is the first-of-its-kind real-time and accurate framework for LiDAR-based crowd counting on a smart campus.*

Scalability. Publicly available LiDAR datasets for high-density outdoor crowds are scarce. For example, CrowdBot [38] supports densities of only 0.1 to 1 person/m^2. To overcome this, we generated a synthetic dataset doubling the density to 2 persons/m^2, providing a more realistic and challenging crowd counting evaluation. We applied random offsets to the x and y coordinates of single-person point clouds to simulate varying pedestrians distances. To improve realism, we added object data proportional to the number of simulated pedestrians (e.g., 10 object data samples for 20 pedestrians). Figure 11 (a)-(c) visualizes the point clouds of these density levels and Figure 11 (d)-(f) visualizes the offset distributions.

We simulated three density levels based on Fruin et al.'s criteria [39]: 1) *Low Density* (up to 1 person/m^2), 2) *Moderate Density* (less than 2 people/m^2), and 3) *High Density* (more than 2 people/m^2). To model these densities, we generated synthetic data following our setup, where the LiDAR deployment covers a 5-meter walkway located 12 to 35 meters from the sensor. We simulated a 100-square-meter area to match

TABLE V: Crowd counting performance including accuracy and speed for HAWC-CC and baselines models in full precision (FP32) and 8-bit (Int8). Bracketed values indicate the relative MAE/MSE improvements of HAWC-CC over the baseline frameworks. OC-SVM-CC is excluded from quantized accuracy comparison (denoted with "-") since it does not support quantization. Speed is measured as the inference time of the entire system for processing a single LiDAR sample on Nvidia Jetson Nano, with both the average value and the standard deviation reported.

Framework	FP32		Int8				
	MAE	MSE	MAE	MSE	MAE Difference	MSE Difference	Speed (ms)
OC-SVM-CC	2.84 (86.61%)	5.55 (90.44%)	-	-	-	-	-
AutoEncoder-CC	0.43 (11.63%)	0.78 (32.05%)	0.73 (43.91%)	1.57 (64.22%)	+ 0.30	+ 0.79	46.98 ± 1.70
PointNet-CC	0.63 (39.68%)	0.98 (45.92%)	1.56 (73.72%)	3.3 (83.03%)	+ 0.93	+ 2.32	26.25 ± 0.65
HAWC-CC (Ours)	**0.38**	**0.53**	**0.41**	**0.56**	**+ 0.03**	**+ 0.03**	**17.42 ± 0.46**

TABLE VI: Scalability results for crowd counting with pedestrian counts from 20 to 250, averaged over three runs on 100 LiDAR samples ("±" indicates standard deviation).

# Pedestrians	Density Level	MAE	MSE	Total Count (K)	Actual Count (K)
20	Low	0.473 ± 0.057	0.920 ± 0.062	2	1.952 ± 0.005
30	Low	0.733 ± 0.235	1.647 ± 0.755	3	2.926 ± 0.023
40	Low	0.913 ± 0.095	2.427 ± 0.324	4	3.909 ± 0.008
50	Low	1.107 ± 0.168	3.140 ± 0.738	5	4.889 ± 0.016
60	Low	1.443 ± 0.166	4.543 ± 0.806	6	5.855 ± 0.016
70	Low	1.643 ± 0.075	5.603 ± 0.356	7	6.835 ± 0.007
80	Low	1.877 ± 0.095	6.910 ± 0.238	8	7.812 ± 0.009
90	Low	2.007 ± 0.093	7.600 ± 0.571	9	8.799 ± 0.009
100	Moderate	2.430 ± 0.161	10.983 ± 2.336	10	9.757 ± 0.016
150	Moderate	3.173 ± 0.405	17.747 ± 4.509	15	14.682 ± 0.04
200	High	4.483 ± 0.280	30.977 ± 5.237	20	19.551 ± 0.028
250	High	5.903 ± 0.156	52.090 ± 3.978	25	24.409 ± 0.015

these conditions, with pedestrians randomly distributed within -5 to 5-meter offset along both the x and y axes. This setup generates synthetic crowd data ranging from 7 meters $(12-5)$ to 40 meters $(35+5)$ from the sensor. This approach enables varying pedestrians numbers to simulate various density levels.

Table VI lists the scalability evaluation results across various densities. HAWC-CC is highly accurate and effective in both moderate and high-density settings. Specifically, in moderated density setting (150-person scene), it achieves an average MAE of 3.1 on average, i.e., miscalculation of 3.1 people, achieving 98% accuracy or 2% error. Even in the challenging, high-density setting (250-person scene), our method still performs highly accurately, achieving 97.64% accuracy or only 2.36% error. Additionally, HAWC-CC significantly outperforms SOTA RGB image-based methods in high-density settings by up to 22.54%, compared to 90.9% by Su et al. [13], 77.1% by Liu et al. [14], and 86.27% by Hao et al. [15].

Deployment Issues. We deployed smart blue light poles on ASU campus with elevated summer temperatures. To ensure the reliability of edge devices housed within the pole's compartment, we monitored and analyzed their temperature. Given the vast temperature data collected over a year, we focus on a specific summer period from June 24, 2023 to July 11, 2023. A temperature sensor within the pole records temperature every 1.7 minutes, generating 2500 data points daily. We also cross-referenced this data with weather temperature data from Visual Crossing [40] during the same period.

Figure 10 shows the weather and pole temperatures during this period. The pole temperatures varied each day closely aligned with weather changes, with a maximum temperature of 57.81°C, a minimum of 21.00°C, and an average of 41.95°C.

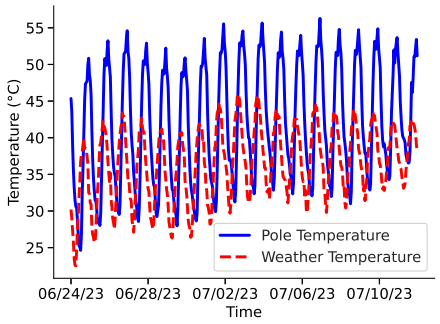
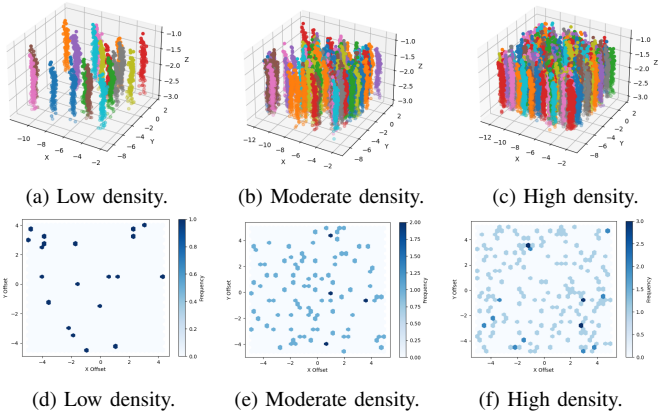


Fig. 10: Pole temperature analysis from summer 2023. We cross-referenced this data with weather temperatures from the same period.



(a) Low density. (b) Moderate density. (c) High density.

(d) Low density. (e) Moderate density. (f) High density.

Fig. 11: Visualization of point clouds (a-c) and offset distribution (e-f) for varied density levels.

The temperature difference between the pole and the weather remains consistent: approximately 10°C higher during peak heat and less than 5°C during cooler period. Notably, despite slightly exceeding the recommended operational range of 0°C to +50°C [41], the Coral Dev Board operated with no issue.

VIII. Conclusions

This paper presents Height-AWare Human Classifier for Crowd Counting (HAWC-CC), a smart blue pole-based crowd management solution for smart campuses. HAWC-CC leverages LiDAR-based data collection, ML-based human detection and counting, and edge computing-based real-time processing. It proposed a novel adaptive clustering method and a novel Height-AWare Human Classifier (HAWC), trained using LiDAR data from the blue light poles and deployed on the Nvidia Jetson and Coral TPU embedded on the poles. Evaluation results show

that HAWC-CC achieves a much faster and more accurate end-to-end performance than the representative related works. HAWC-CC offers a pioneering smart campus infrastructure that synergistically integrates new IoT, ML, and edge computing solutions, and its real on-campus deployment also demonstrates that its design and implementation work well in practice.

While HAWC-CC provides accurate and privacy-preserving crowd counting for smart campus, the proposed classifier, HAWC, relies on an assumption of the average college students height, which may limit its generalizability to populations with significantly different height distributions. A direction to address this is by integrating non-intrusive complementary sensors like thermal sensors to improve the classification result without compromising privacy.

ACKNOWLEDGMENT

This work is supported by National Science Foundation awards CNS-2311026, SES-2231874, OAC-2126291 and CNS-1955593. We thank the anonymous reviewers for their valuable feedback. We also thank Divy Kamlesh Patel and Ahraz Rizvi for their help with data labeling.

REFERENCES

[1] D. Liang, X. Chen, W. Xu, Y. Zhou, and X. Bai, "Transcrowd: weakly-supervised crowd counting with transformers," *Science China Information Sciences*, vol. 65, no. 6, p. 160104, 2022.

[2] B. S. et al., "Switching convolutional neural network for crowd counting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5744–5752, 2017.

[3] X. Jiang, L. Zhang, M. Xu, T. Zhang, P. Lv, B. Zhou, X. Yang, and Y. Pang, "Attention scaling for crowd counting," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4706–4715, 2020.

[4] Z. et al., "Cross-scene crowd counting via deep convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 833–841, 2015.

[5] P. Zhang, Z. Tao, W. Yang, M. Chen, S. Ding, X. Liu, R. Yang, and H. Zhang, "Unveiling personnel movement in a larger indoor area with a non-overlapping multi-camera system," *arXiv preprint arXiv:2104.04662*, 2021.

[6] Z. Wang, R. Arablouei, J. Liu, P. Borges, G. Bishop-Hurley, and N. Heaney, "Point-syn2real: Semi-supervised synthetic-to-real cross-domain learning for object classification in 3d point clouds," in *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1481–1486, IEEE, 2023.

[7] Ouster, "OS0: Ultra-Wide View High-Resolution Imaging Lidar.," 2020.

[8] M. De Deuge, A. Quadros, C. Hung, and B. Douillard, "Unsupervised feature learning for classification of outdoor 3d scans," in *Australasian conference on robitics and automation*, vol. 2, University of New South Wales Kensington, Australia, 2013.

[9] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*, pp. 3354–3361, IEEE, 2012.

[10] Q. et al., "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

[11] C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou, "Autoencoder for words," *Neurocomputing*, vol. 139, pp. 84–96, 2014.

[12] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[13] J. Su, H. Zhang, X. Luan, and Q. Tian, "Fusion attention mechanic crowd counting network based on transformer," in *Proceedings of the 2024 8th International Conference on Control Engineering and Artificial Intelligence*, pp. 37–43, 2024.

[14] L. et al., "Cross-modal collaborative representation learning and a large-scale rgbt benchmark for crowd counting," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4823–4833, 2021.

[15] L. Hao, B. Huang, B. Jia, and G. Mao, "Heterogeneous dual-attentional network for wifi and video-fused multi-modal crowd counting," *IEEE Transactions on Mobile Computing*, 2024.

[16] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 589–597, 2016.

[17] C. et al., "Mac address randomization tolerant crowd monitoring system using wi-fi packets," in *Proceedings of the 16th Asian Internet Engineering Conference*, pp. 27–33, 2021.

[18] W. et al., "People counting with carry-on rfid tags," in *2023 IEEE/ACM 31st International Symposium on Quality of Service*, pp. 1–10, 2023.

[19] J. e. a. Kay, "The caltech fish counting dataset: A benchmark for multiple-object tracking and counting," in *Computer Vision – ECCV 2022*, pp. 290–311, Springer Nature Switzerland, 2022.

[20] H. et al., "Crowdotic: Transformer-based occupancy estimation for hospital waiting rooms with non-speech audio and differential privacy," *arXiv preprint*, 2023.

[21] J. Zarzar, S. Giancola, and B. Ghanem, "Efficient bird eye view proposals for 3d siamese tracking," *arXiv preprint arXiv:1903.10168*, 2019.

[22] K. et al., "Rethinking range view representation for lidar segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 228–240, 2023.

[23] L. et al., "Density-net: A density-aware network for 3d object detection," in *2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 1105–1112, IEEE, 2021.

[24] B. et al., "Birdnet+: End-to-end 3d object detection in lidar bird's eye view," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, IEEE, 2020.

[25] O. Vinyals, S. Bengio, and M. Kudlur, "Order matters: Sequence to sequence for sets," *arXiv preprint arXiv:1511.06391*, 2015.

[26] H.-P. Kriegel, P. Kröger, J. Sander, and A. Zimek, "Density-based clustering," *Wiley interdisciplinary reviews: data mining and knowledge discovery*, vol. 1, no. 3, pp. 231–240, 2011.

[27] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 86–97, 2012.

[28] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[29] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J. M. Pérez, and I. Perona, "An extensive comparative study of cluster validity indices," *Pattern recognition*, vol. 46, no. 1, pp. 243–256, 2013.

[30] M. Hahsler, Piekenbrock, *et al.*, "dbscan: Fast density-based clustering with r," *Journal of Statistical Software*, vol. 91, pp. 1–30, 2019.

[31] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[32] Google, "TFLiteConverter.," 2019.

[33] B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," *Advances in neural information processing systems*, vol. 12, 1999.

[34] A. Leigh, J. Pineau, N. Olmedo, and H. Zhang, "Person tracking and following with 2d laser scanners," in *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 726–733, IEEE, 2015.

[35] T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi, *et al.*, "Keras Tuner." https://github.com/keras-team/keras-tuner, 2019.

[36] S. Zhou, L. Li, X. Zhang, B. Zhang, S. Bai, M. Sun, Z. Zhao, X. Lu, and X. Chu, "Lidar-ptq: Post-training quantization for point cloud 3d object detection," *arXiv preprint arXiv:2401.15865*, 2024.

[37] T. Xu, K. Shen, Y. Fu, H. Shi, and F. X. Lin, "Rev: A video engine for object re-identification at the city scale," in *2022 IEEE/ACM 7th Symposium on Edge Computing (SEC)*, pp. 189–202, IEEE, 2022.

[38] D. Paez-Granados, Y. He, D. Gonon, L. Huber, and A. Billard, "3d point cloud and rgbd of pedestrians in robot crowd navigation: Detection and tracking," *IEEE DataPort*, vol. 12, 2021.

[39] J. Fruin, *Pedestrian Planning and Design*. Metropolitan Association of Urban Designers and Environmental Planners, 1971.

[40] Visual Crossing Corporation, "Visual Crossing Weather. 2023/06/24-2023/07/11.," 2023.

[41] Google, "Coral Dev Board Datasheet, Version 1.7," 2022.