# Area4U: Predicting Interaction Hotspots for Large Public Displays

Xinyuan Zhang[1,2], Xiaoyang Mao[2], Wan-Young Chung[3], ZhiGang Gao[4], Jianwen Feng[1], Kentaro Go[2*]
[1] School of Computer Science, Hangzhou Dianzi University, Hangzhou, China
[2] Department of Computer Science and Engineering, University of Yamanashi, Kofu, Japan
[3] Department of Artificial Intelligence Convergence, Pukyong National University, Busan, South Korea
[4] College of Information Engineering, China Jiliang University, Hangzhou, China
E-mail: [1]{zhxy, fengjianwen}@hdu.edu.cn, [2]{mao, go}@yamanashi.ac.jp, [3]wychung@pknu.ac.kr, [4]gaozhigang@cjlu.edu.cn

***Abstract*—Effective interaction with large public interactive displays (LPIDs) remains a significant challenge in human-computer interaction. This paper introduces "Area4U," a user interaction area prediction framework designed for enhanced interaction with LPIDs. Area4U employs lightweight deep-learning techniques to analyze webcam imagery and deduce potential users' location and motion information, utilizing this information to predict and allocate sub-regions users prefer for their usage, the framework streamlines the transition of potential users to active participants. Our model was trained on video data containing 212 sub-region selections from 12 participants. Results indicate that our method effectively identifies areas of interest to potential users in an experimental setting, proving the feasibility and effectiveness of our framework.**

***Index Terms*—Large Public Interactive Displays (LPID), User Behavior Prediction, Lightweight Deep Learning, Interaction Hotspots, Multimedia Computing**

## I. Introduction

Large public interactive displays (LPIDs) are gaining popularity in general settings like shopping malls and subway stations due to their real-time information delivery, interactive capabilities, personalized content, and improved affordability. In recent years, these public interactive screens have begun to replace traditional static billboards in some public areas [1] and have been widely applied in places such as semi-public whiteboards [2] and informational boards [3].

LPIDs, unlike traditional devices that require touch, initiate interaction when a user enters the screen's influence range. As depicted in Figure 1 Individuals entering the influence range are classified as potential users who can either become high-probability users or passersby. Although it is easy for high-probability users to become real users, it will also lead to user loss if there is no good human-computer interaction design. In their study [4], Madrian et al. pointed out that users often prefer to choose the default settings when using a system, which helps them complete tasks quickly. For LPID, providing default choices can reduce decision stress and attract more user.

Spatiotemporal resource utilization is another problem for LPID. Users often need to move back and forth to grasp complete information and engage in touch interactions, leading to fatigue [5]. Additionally, current interaction systems primarily cater to individual users, leading to decreased screen utilization when a group shares the different purpose. Implementing a multi-user interaction system would be more efficient in improving LPID utilization rates.

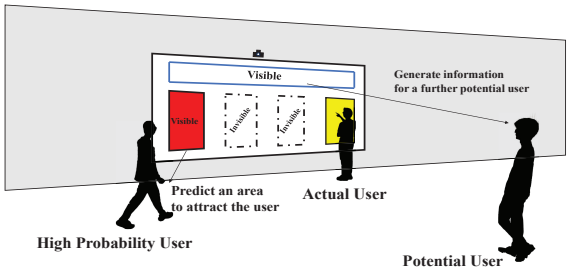



Fig. 1: Overview of proposed architecture Area4U.

In this paper, we propose "Area4U," an interactive sub-area prediction framework for multi-users while adhering to the constraints of using off-the-shelf equipment. Our study scenario concept diagram is shown in Figure 1, we divide the LPID into two distinct areas: the "easy operate area" and the "difficult operate area." The "difficult operate area" refers to the top part of the LPID, which often presents challenges for most users to access and interact with. In contrast, the "easy operate area" is subdivided into several "sub-interaction areas"(the number of sub-areas needs to be defined by the actual environment). Area4U focuses on identifying users with high probability. By analyzing the user's location information and body behavior, Area4U predicts and allocates one of the sub-interaction areas, making this area look like a default choice to attract high-probability users. As our interaction architecture concentrates on generating an interaction area, no explicit interaction events are designed in the interaction phase.

Our main contribution lies in developing an interactive sub-area prediction framework without large amount of computing that can be deployed on edge devices. Considering practical LPID usage scenarios with limited computing resources and deployment costs, we utilize only two webcams as additional equipment. Both the analysis and prediction modules in the framework use lightweight deep-learning models to

*: Corresponding author

reduce computational requirements. By testing on the data set obtained in the data collection experiment, our proposed framework process is effective and has a considerable accuracy rate.

## II. Related Work

### A. Multiple User Interaction

Past research has focused on multiple user interactions on LPIDs. Vogel et al. [6] use sensors to create personalized interaction windows based on the user's position and distance from the screen. In recent research, Courtoux et al. [7] have designed a novel controller that allows for the arbitrary slicing and copying of content on the LPID to achieve multi-user interaction.

Compared to existing studies, our study focuses on using off-the-shelf devices to enable multi-user interaction, that is, by dividing the screen into multiple sub-regions and assigning them to users. Compared to controllers or sensors, touch reduces user learning costs and has better potential for real-world deployment because it is easier to maintain without additional equipment.

### B. User Location Prediction

User location prediction has always been a prominent topic, frequently mentioned in areas like smart cities. Castro-Gonzalez et al. [8] statistically analyzed users' hotspot locations and movement patterns, utilizing Hidden Markov Chain methods to achieve pedestrian location prediction. Kooij et al. [9] constructed Bayesian networks to predict whether pedestrians will cross the road in autonomous driving scenarios.

There is no doubt that their contribution is excellent, but the above studies have ignored the computational performance of the deployed machines in the actual deployment. LPID, as an edge computing device, has weak computational power, running normal deep learning models is challenging, and placing a high-performance GPU device behind the screen is impractical in the actual deployment environment. Therefore, to ensure the mobile deployment capability of the proposed method, the use of lightweight deep learning models is a better choice which was our research goal.

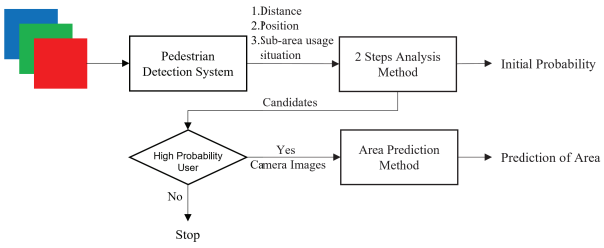



Fig. 2: Architecture of the whole framework.

## III. Framework Architecture

Area4U aims to analyze whether a user will become a high-probability user through their location information. For high-probability users, an interaction area is predicted by analyzing their movements and allocating to them. This framework consists of three parts: Pedestrians Detection System, Two Steps Analysis Method, and Area Prediction Module.

### A. Pedestrians Detection System

The Pedestrians Detection System offers vital data structure for subsequent procedures. Precise pedestrian detection is a critical factor for the framework's effective operation. To fulfill this, we have devised our Pedestrian Detection System based on a lightweight deep learning algorithm YOLOv7-tiny, a streamlined iteration of YOLOv7 [10], which is well-suited for edge devices such as LPIDs.

By modifying its backbone network architecture, we enhanced YOLOv7-tiny's capability to detect details and small objects, specific implementation details will be disclosed in a later paper. In the output layer, we optimize the loss function. The loss function used in YOLOv7-tiny is represented as equation (1), consisting of bounding box position loss $L_{box}$, class loss $L_{cls}$, object confidence loss $L_{obj}$.

$$L = L_{box} + L_{cls} + L_{obj} \tag{1}$$

We choose VarifocalLoss to enhance YOLOv7-tiny's learning ability on low-resolution images. Varifocal Loss puts forward an indicator $IACS$(IoU-Aware Classification Score) that can simultaneously represent $L_{box}$ $L_{cls}$ and the equation is shown below:

$$VFL(p,q) = \begin{cases} -q(qlog(p)) + (1-q)log(1-p) & q > 0 \\ -\alpha p^{\gamma} log(1-p) & q = 0 \end{cases} \tag{2}$$

Where $p$ is the prediction box's $IACS$ score and $q$ is the target $IoU$ score. For positive samples in training, $q$ is set to $IoU$ between the predict bounding box and the ground truth box, and for negative samples in training are set to 0. The final loss function of our method is as follows:

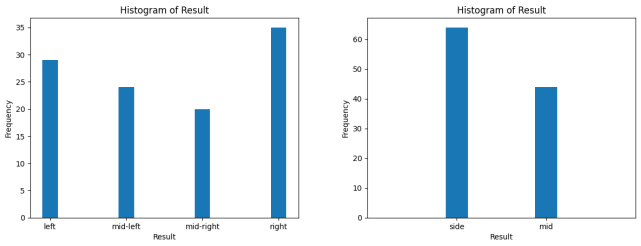
$$L = L_{VFL} + L_{obj} \tag{3}$$

To verify the reasonableness of the structural changes and loss function optimization, and let the model learn the multi-scale features of pedestrians, we wrote a Python script to merge WiderPerson [11] and CrowdHuman [12]. Finally, we trained the improved YOLOv7-tiny on it and estimate the performence.

In the detection layer of YOLOv7-tiny, we integrated the Mono-Depth to estimate pedestrians' distances. The distance of the detected pedestrian can be generated with coordinates and the picture's depth map. While monocular depth estimation may not match the precision of stereo or depth cameras, it performs well when the background is stable. Using monocular depth estimation as the distance generator is a good choice, as it aligns with our principle of minimizing additional devices.

### B. Two Steps Analysis Method

Due to the absence of a public data set on user interactions with LPIDs, using the deep learning method to estimate pedestrians' using probability is infeasible. Therefore, we




(a) Original Data (b) Data in "side" and "mid"

Fig. 3: Data distributions used in initial probability analysis.

choose to collect simulated data which represents the selection when users stand at the edge of the LPID's influence scope. The distribution of the selection is shown in Figure 3, here "side" refers to the regions at the leftmost and rightmost, and "mid" encompasses the areas excluding the "side" regions, concepts of them will be used in the following. Obvious data imbalance observed in Figure 3(a), making it difficult to train a classifier to calculate the usage probability of each choice, and to combat this problem, we proposed Two Steps Analysis Method.

The Method is applied to the scenarios shown in Figure 4. An operating zone was defined to simulate the range of user interactions. Subsequently, distinct zones were designated at varying distances: Zone 1(1m to 2.5 m) zone 2 (2.5m to 4m), and Zone 3 (4m to further). Distance setting depends on the actual environment and the display's size. This setting of different interaction intervals was applied in [13] to design different interaction methods, and here we set it to examine the impact of distance on users' sub-region selection.

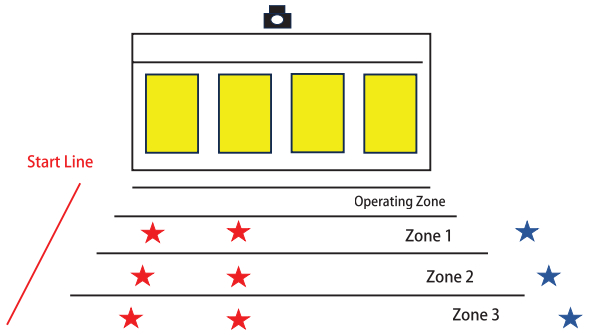



Fig. 4: Conceptual data collection experiment setting.

The Two Steps Analysis Method is based on XGBoost [14] machine learning algorithm for analyzing user interaction probabilities for each sub-area. The input data structure for the Two Steps Analysis Method is represented as follows:

$$\begin{aligned}(Z(dis),\ P, U = [A_1,\ A_2,\ \ldots,\ A_x] \mid A_x \in \{-1,\ 0,\ 1\}) \\ x\ =\ Number\ of\ SubArea\end{aligned} \tag{4}$$

Here, $U$ signifies the collection of usage statuses for all screen sub-regions. These statuses encompass three conditions: $-1$ (in current use), 0 (not currently used), and 1 (selected). Permissible transitions exist from $-1$ to 0 and 0 to 1; no transition occurs from $-1$ to 1. $Z$ is denoted as the zone based on the distance $dis$. $P$ is the left and right position of where the detected pedestrian is.

The Two Steps Analysis Method involves using DSM (Divide Side and Mid) and XGBoostTrainer. DSM helps in dividing the data into two subsets "side" and "mid." The algorithm for this division is shown in Algorithm 1. Here, $ds$ represents the Data Structure, while $side$ and $mid$ represent the two subsets. This algorithm identifies the regions with the highest and lowest labels within the input data and assigns them to the side subset. The remaining regions are grouped into the mid-subset.

**Algorithm 1** $DSM$ (Divide Side and Mid)

**Input:** Data Structure ds = (Z, P, U, Selection Result) as ds
**Output:** Side Training Data Mid Training Data
**if** $ds$ = null **then**
  **return** null
**end if**
$side \leftarrow \{\}$
$mid \leftarrow \{\}$
$leftSideIndex \leftarrow ds$.Selection Result.Min
$rightSideIndex \leftarrow ds$.Selection Result.Max
**for** $item$ **in** $ds$ **do**
  **if** $leftSideIndex = rightSideIndex$ **then**
    $mid$.append($item$)
  **end if**
  **if** $item$.Selection Result $= leftSideIndex$
  **or** $item$.Selection Result $= rightSideIndex$ **then**
    $side$.append($item$)
  **else**
    $mid$.append($item$)
  **end if**
**end for**
**return** $side$, $mid$

For Algorithm 2 XGBoostTrianer uses the copied data set to train "MS_Classifier" to judge "mid" or "side," and uses the original "side" data set to train left side and right side classifier "LR_Classifier." Then put them in the list of global variables "ListOfXGBoost" in the order of MS_Classifier followed by LR_Classifier.

**Algorithm 2** XGBoostTrainer

**Input:** Data Structure = (Z, P, U, Selection Result) as ds
**Output:** $ListOfClassifier$
$ListOfXGBoost(GlobalVariable) \leftarrow \{\}$
$(side, mid) \leftarrow DSM(ds)$
**if** $side \neq$ null **then**
  $side_{copy} \leftarrow side$
  $mid_{copy} \leftarrow mid$
  $side_{copy}$.Selection Result $\leftarrow$ "side"
  $mid_{copy}$.Selection Result $\leftarrow$ "mid"
  $train\_ds \leftarrow side_{copy}$.append($mid_{copy}$)
  $x \leftarrow train\_ds$.drop(Selection Results)
  $y \leftarrow train\_ds$.get(Selection Results)
  $MS\_Classifier \leftarrow$ XGBoost.fit($x, y$)
  $xx \leftarrow side$.drop(Selection Result)
  $yy \leftarrow side$.get(Selection Result)
  $LR\_classifier \leftarrow$ XGBoost.fit($xx, yy$)
  $ListOfXGBoost$.append($MS\_Classifier, LR\_classifier$)
  XGBoostTrainer($mid$)(Start from line 2)
**else**
  **return** null
**end if**
**return** $ListOfXGBoost$

For the usage method of classifiers, inter-group classifiers retain odd identifiers, while intra-group classifiers possess even

identifiers. We assign the index 0 to the first MS_Classifier in the list. The Two Steps Analysis Method proceeds as follows:

1) Input the data structure and apply the MS_Classifier to calculate the probability for the "side" mid "mid."
2) If the probability for "side" is higher then use the following LR_Classifier to determine left or right. If "mid" is higher skip the next LR_Classifier and use the next MS_Classifier for mid-subset, repeat step 1.

At each step, if the probability of both options is less than 50%, then the person will not be considered a high-probability user and the framework process terminates.

### C. Area Prediction Method

As the last part of the framework, the Area Prediction Method aims to predict the interaction area for high-probability users by analyzing their body motivation identified by the first two modules. Based on the idea of easy deployment, the module is built on the lightweight deep learning framework MoviNets [15].

MoviNets uses streaming buffers to substantially decrease the memory requirements for training and inference. Moreover, it facilitates online inference for streaming videos. The underlying principle of this approach is elucidated in Equation (5). Where $B_i$ is denoted as the buffer, $x_i^{clip}$ is the different clips of a video. When calculating the feature of clip $F_i$, the next buffer $B_{i+1}$ is updated with the last $b$ frames' feature, concatenated with the next clip's $x_{i+1}^{clip}$ for subsequent analysis. As the number of clips increases, features expand progressively without incurring additional computational load, facilitating online video inferences.

$$F_i = f(B_i \oplus x_i^{clip}) \tag{5}$$

$$F_i = f(B_i \oplus x_i^{clip}) \tag{6}$$

Due to its ability to motivate analysis and need less computation, we chose this lightweight deep learning method to be basis of the prediction. Apart from MoviNets itself, we incorporated a video data preprocessing module to generate training sets and enhance prediction speed for the subsequent sub-interaction area.

## IV. Data Collection Experiment

### A. Overall

We designed a data collection experiment to deal with the absence of publicly available data on user interactions with LPIDs. The core idea of the design was to simulate the actual situation as much as possible under the condition of controlled variables to obtain the datasets for training the Two Steps Analysis Method and Area Prediction Module.

### B. Experiment Setting

The real-world experiment setting is the same as the conception diagram in Figure 4, as shown in Figure 5(a): short green markers correspond to the red stars, while long markers on the right correspond to the blue stars. The left line signifies the boundary of the screen's influence range. We invited 12 graduate students as participants, ages 23 to 28, with an average age of 24.6.

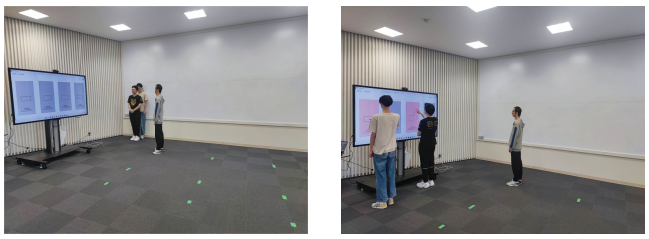
For devices, an 85.6-inch large touchscreen (AHA Co. Ltd) was used as the experimental LPID with the data collection system shown in Figure 5(b), a PC (HP OMEN 7) with two webcams (Logitech C870) with different angles used to record participants' selections and body movements during the experiment.

(a) Environment (b) Data Collection System

Fig. 5: Experimental environment and equipment.

### C. Collection Task

The experiment consists of the right-side static experiment and the left-side dynamic experiment. To calculate the initial probability, right side experiment needs to collect the tendency of the participants to each sub-area when they are located at the edge of the influence range of the screen with different distances. The left-side dynamic experiment collected participants' choices and body motivation information in the screen's influence range. It was set on the left because previous experiments on the right gave participants the impression of the space distribution of the sub-area, which reduced their screen selection time and distorted the movement data collected. Switching the directions of the two experiments won't affect the results, because there is a mirror effect on the area selection task.



(a) First Select (b) Last Selection

Fig. 6: Right side experiment setting.

*1) Right-side Static Experiment:* In the experiment on the right-hand side, one participant is positioned on the rightmost label of a specific zone and asked to choose an area that is currently unused. Once this initial selection is made, an assistant will utilize the chosen area. After that, the participant makes a second selection from the remaining choices, and another assistant will use that selection. Finally, for the third choice, the participant completes the selection process,

concludes the experiment for that particular label, and moves on to the next rightmost label in a different zone. Figure 6 shows some photos of the experiment.

In this experiment, the screen occlusion by the assistant is crucial for two reasons. Firstly, it is essential to simulate a real-world scenario involving multiple users. Secondly, the absence of the assistant would result in all three choices under the same conditions. This absence would distort the selection results, consequently impacting the subsequent calculation of user usage probabilities.

*2) Left-side Dynamic Experiment:* The left-side experiment is illustrated in Fig 7. Participants are instructed to stand outside the "Start Line" on the left side of the experiment scene to prepare. Once prepared, the participant can hold the timer controller and proceed to the specified ground label. During both the preparation and walking phases, participants must keep their gaze parallel to the screen to avoid selection bias caused by observing the screen. From another perspective, this approach aims to control the variable factors in the experiment, ensuring its reproducibility.

(a) Preparation (b) Move to screen

Fig. 7: Left side experiment setting.

After using the controller to start the timer, the participant can move their gaze and body, select one area, walk to its operating area, close the timer, and touch the area to complete the selection. The data collection system records the selected area and time cost. At this stage, there are no restrictions on selecting and reaching the selected area, such as walking directly or wandering. Like the right-side experiment, an assistant would use the selected area after each participant's selection, and the participant would make the following selection. Each experiment point involves three selections, with six experiment points on the left side.

### D. Collected Data

In the right-side experiment, we collected 108 times choices from participants' selection for initial probabilities analysis. The left-side experiment collected 216 area selections and participants' motion video data.

The area selection behavior consists of three stages: preparation, decision, and action. The preparation stage won't be used to train the Area Prediction Module. Participants accompanied their body movements during the decision stage and moved their gaze to find the best choice. In the action stage, participants moved towards the operating zone of the selected area. An illustrative example is shown in Fig 9.

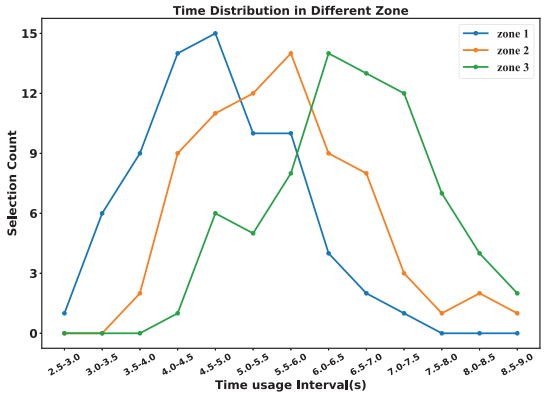



Fig. 8: Data distribution in three different zones.

### E. Data Process

Using the time-recording function of the experimental system, we analyzed the time spent in different zones. As shown in Fig 8, it takes approximately 4-6 seconds to complete area selection actions in zone 1, 5-7 seconds in zone 2, and 4-8 seconds in zone 3. The time cost data revealed four erroneous motion data, which we removed manually. The remaining 212 motion data contained various ways of moving and walking toward the screen from six different positions in front of the screen. We extracted relevant video frames from the recorded footage using this information. For training the area prediction method combined with the decision and action stage, we extracted 180, 210, and 240 frames from the videos of zone 1, zone 2, and zone 3, respectively.

To solve the problem that the selected frames contain redundant information, affecting the model's training performance due to the same frame with different labels, some experiment was set to find the best FPS for model training. Additionally, to synchronize with the Pedestrians Detection System, we wrote an automatic script to preprocess the input motion data, resized the videos to 512×512 resolution, and cropped them according to the specified duration. Finally, we get the data set for training the Area Prediction Method with the motion video as the data selection area as the label.

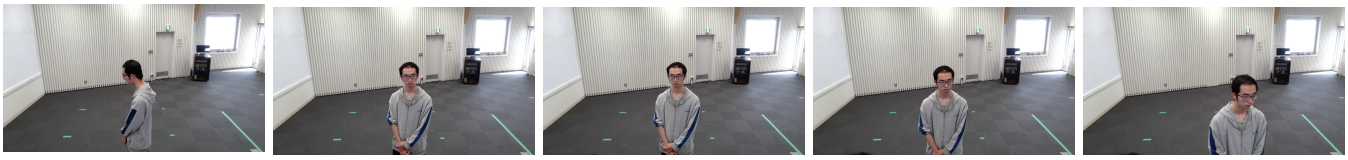
### F. Training Result

TABLE I: Pedestrian Detection System's Accuracy

| Model | mAP@.5 | mAP@.95 | FLOPs | FPS |
|---|---|---|---|---|
| YOLOv7 | 67.6 | 39.2 | 103.2G | 69.0 |
| YOLOv7-tiny | 66.6 | 36.9 | 13.0G | 106.4 |
| Ours | **69.1** | **40.2** | 12.8G | 102.0 |

Results are shown in Table I by training on the joint dataset of WiderPerson and CrowdHuman, with input 512×512 pixels, and testing on the WiderPerson test set. The test results show that our changes to YOLOv7-tiny improve the model's accuracy in low-resolution scenes and significantly reduce the amount of calculation, providing theoretical support for mobile device deployment.

We employed the Two Steps Analysis Method on the Random Forest and XGBoost, respectively, to test the method's



Prepare Consider Consider Consider Move

Fig. 9: Example area selection actions.

TABLE II: Result for Apply Two Steps Ananlys Method

| Method | Accuracy | F1-Score |
|---|---|---|
| Random Forest | 65.74% | 65.74 |
| XGBoost | 70.67% | 70.33 |
| Random Forest(Two Steps Analysis Method) | 71.30% | 73.24 |
| XGBoost(Two Steps Analysis Method) | **80.13%** | **79.21** |

effectiveness, as outlined in Table II. Due to the limited availability of training data, the precision of the results is not as high as observed in other studies. Still, outcomes demonstrate that the decision models derived from the Two Steps Analysis Method surpass the accuracy of directly classified models. The method mitigates the data imbalance problem in the sub-interaction area selection scenario.

TABLE III: Architecture's Final Accuracy and Time Cost

| Image Size | FPS | Accuracy(%) | Inference Time(s) |
|---|---|---|---|
| 1920×1080 | 30 | 78.81 | 100 |
| 512×512 | 30 | 77.56 | 40 |
| 512×512 | 10 | **88.60** | 18 |
| 512×512 | 5 | 88.12 | 4 |
| 512×512 | 2 | 86.74 | **2** |

The accuracy of our Area Prediction Method in predicting areas with motion videos under different conditions is shown in Table III. The accuracy generated with five-fold cross-validation represents the overall performance of our framework, as the first two modules are responsible for screening the prediction objects of the prediction module and providing corresponding motion data. The final region prediction still depends on the prediction module itself. Results show that our preprocessing for motion data is effective. Additionally, we found that reducing FPS does not cause a significant loss of accuracy, making our framework theoretically feasible for practical deployment when faster prediction speed is needed in actual scenes.

## V. Conclusion

In this paper, we proposed "Area4U," a framework with a three functions pipeline that detects pedestrians, judges the probability of being the user, and predicts and allocates an area to the high-probability user. The methods used in this framework all have low computational requirements and with just two webcams, making it easy to deploy in real-world environments. Our framework achieved promising results in predicting interested areas by analyzing 212 selection actions from 12 participants. However, our study has limitations. We only tested our idea using simulated data and under ideal conditions. Additionally, there are unresolved user privacy concerns due to webcams. In the future, we plan to conduct further tests in real-world environments to validate our research and prove its feasibility, where we can deal with more complex situations.

## References


[1] J. Müller, F. Alt, D. Michelis, and A. Schmidt, "Requirements and design space for interactive public displays," in *Proceedings of the 18th ACM international conference on Multimedia*, 2010, pp. 1285–1294.

[2] D. M. Russell, C. Drews, and A. Sue, "Social aspects of using large public interactive displays for collaboration," in *UbiComp 2002: Ubiquitous Computing: 4th International Conference Göteborg, Sweden, September 29–October 1, 2002 Proceedings 4*. Springer, 2002, pp. 229–236.

[3] A. Azad, J. Ruiz, D. Vogel, M. Hancock, and E. Lank, "Territoriality and behaviour on and around large vertical publicly-shared displays," in *Proceedings of the Designing Interactive Systems Conference*, 2012, pp. 468–477.

[4] B. C. Madrian and D. F. Shea, "The power of suggestion: Inertia in 401 (k) participation and savings behavior," *The Quarterly journal of economics*, vol. 116, no. 4, pp. 1149–1187, 2001.

[5] M. R. Jakobsen, Y. Jansen, S. Boring, and K. Hornbæk, "Should i stay or should i go? selecting between touch and mid-air gestures for large-display interaction," in *Human-Computer Interaction–INTERACT 2015: 15th IFIP TC 13 International Conference, Bamberg, Germany, September 14-18, 2015, Proceedings, Part III 15*. Springer, 2015, pp. 455–473.

[6] D. Vogel and R. Balakrishnan, "Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users," in *Proceedings of the 17th annual ACM symposium on User interface software and technology*, 2004, pp. 137–146.

[7] E. Courtoux, C. Appert, and O. Chapuis, "Surfairs: Surface+ mid-air input for large vertical displays," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 2023, pp. 1–15.

[8] A. Castro-Gonzalez, M. Shiomi, T. Kanda, M. A. Salichs, H. Ishiguro, and N. Hagita, "Position prediction in crossing behaviors," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 5430–5437.

[9] J. F. Kooij, F. Flohr, E. A. Pool, and D. M. Gavrila, "Context-based path prediction for targets with switching dynamics," *International Journal of Computer Vision*, vol. 127, no. 3, pp. 239–262, 2019.

[10] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7464–7475.

[11] S. Zhang, Y. Xie, J. Wan, H. Xia, S. Z. Li, and G. Guo, "Widerperson: A diverse dataset for dense pedestrian detection in the wild," *IEEE Transactions on Multimedia*, vol. 22, no. 2, pp. 380–393, 2019.

[12] S. Shao, Z. Zhao, B. Li, T. Xiao, G. Yu, X. Zhang, and J. Sun, "Crowdhuman: A benchmark for detecting human in a crowd," *arXiv preprint arXiv:1805.00123*, 2018.

[13] V. Cheung and S. Scott, "Proxemics-based visual concepts to attract and engage public display users: Adaptive content motion and adaptive user shadow," in *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces*, 2016, pp. 473–476.

[14] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

[15] D. Kondratyuk, L. Yuan, Y. Li, L. Zhang, M. Tan, M. Brown, and B. Gong, "Movinets: Mobile video networks for efficient video recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 020–16 030.