Cutting Edge Surveillance: *Final*

Paul Mitchell
Undergraduate Student of William and Mary
CSCI 420 Edge Computing
Williamsburg, VA
pjmitchell@wm.edu

https://github.com/pjmitchell7/cs420*underscore*final

I. Summary

The primary research question is: "Can edge computing be effectively leveraged to achieve person profile detection for home security applications?" This question focuses on the need for more efficient and effective methods of identifying individuals. Edge computing technologies can enable real-time, effective and accurate detection. The significance of this study goes beyond mere academic concern: it aims to address the urgent public concern of how to improve home security in light of the increasingly digital worlds in which we live. More specifically the focus is to assess the viability of edge computing for applying Support Vector Machine (SVM) based facial recognition, for optimizing home security systems. Another focus of the project aims to test whether edge data processing can effectively reduce latency compared to cloud while minimizing network bandwidth consumption to ensure a timely response to security incidents. The approach holds out the promise of making home security systems both more responsive and efficient. The system would theoretically offer more immediate input than traditional cloud based home security systems for recognizing threats earlier on. Among other things, it should promote the safety and health of people living in smart environments. By using edge computing technology in the home security system, this will be addressing both the current explosive growth of the home security market and also technological progress in edge computing.

II. Background and Motivation

In the United States alone, the number of households equipped with intrusion alarms or video surveillance has now exceeded 39 million. The global home security market is anticipated to increase from 40.7 billion dollars in 2020 to 84.4 billion dollars in 2027 that 8.2 percent average annual growth rate. The driving forces behind this growth are knowledge, crime, and technology. At least 39 million American homes have installed alarm systems, and video surveillance is the most popular technology. This leads them to monitor both children and homes during family absences for safety reasons. It was also found that non-secured property was the source of three times more of the burglaries. [1] As the household security market continues to grow, the edge computing market is seeing a huge boom. The edge computing market in the United States was assessed at USD 97.09 billion in 2023 and is expected to reach USD 1,043.89 billion by 2032, with a growth rate of 30.20 percent. This rapid rise highlights the importance of edge computing in increasing operational efficiencies, reducing latency, and ensuring better data security coupled with lower costs. The inclusion of edge computing technology into home security systems can greatly enhance real-time analytics capabilities, decisions made about these systems, and their overall effectiveness. [2] The confluence of the growth in home security and the advent of edge computing presents a compelling case for exploring and adopting edge computing systems to bolster home security solutions. This approach not only provides a way forward in terms of consumer expectations—whether for improved responsiveness or increased efficiency in home alarm systems—but it also coincides nicely with the broader market trends. [3]

III. Existing Methods Related To Edge Computing Home Security

A. Edge AI Cameras

AI cameras for real-time surveillance that process their data locally. This includes detection of moving objects or people's faces to tell whether they are normal or suspicious-carried out without network support, so as to protect privacy while at the same time reducing latency. A real world example of edge computing applied in home security, is the Orbbec Persee N1 camera.[4] Using local processing technology to analyze the environment with people counting and feature recognition software, in such an effective way that demonstrates how edge computing can indeed be useful in enhancing security - thus this provides a guiding reference to aid in the the feasibility of this project's surveillance objectives.

B. Decentralized Data Processing

The decentralized processing method improves the security system against possible collapse. The system processes data on devices in the area, and can work well on such equipment as conventional computers, network appliances, PDAs (personal digital assistants), printers, etc. Even if one piece of equipment is infiltrated or damaged, the whole network can still function normally–cutting the need for reliance on centralized servers.

IV. Issues Current Research Has Yet to Address

A. Data Privacy and Security

There is a growing need for reliable data security directly from edge devices as cyber-crimes become more and more common each year, including safeguards against destruction due to physical tampering and unauthorized access. This situation demands continued attention. One example is the 'Kia Boys' who walk up to people's electric vehicles in front of their house and within a matter of seconds or minutes they can unlock the doors and rewire the engine to drive away before anyone would notice.

B. Realtime Processing Efficiency

Edge devices must be optimized to have low processing latencies and still do their job quickly if we want effective home security systems. This is still an area in which progress can be made.

V. Closing Research Gaps

A. Enhancing Data Protection Mechanisms

I intend to incorporate cutting-edge encryption techniques and secure data handling practices specifically tailored for edge computing environments. My goal would be that users' data is protected both during transmission and on the devices themselves, as I believe this would address crucial privacy concerns that can be significant in home security contexts.

B. Efficiency for Real-Time Analysis

Focusing on the refinement of algorithms capable of running on edge devices, my project will seek to reduce latency and improve the accuracy of real-time data analysis. By optimizing these algorithms for efficiency, the project aims to ensure that security systems are capable of immediate response to threats, thereby elevating the standard of protection offered to homeowners.

VI. Experimental Design

A. Methodology

The methodology for this experiment involves a comparative study on the performance of SVM-based facial recognition in two different environments: edge and cloud computing. Either case will involve the training of an SVM model using a component-oriented approach to facial recognition, as the method allows for the analysis of smaller portions of an image rather than the whole face. This is chosen for its resilience to pose variation, a common challenge in real-world scenarios such as surveillance systems. It is also well-suited for real-time processing requirements

B. Variables

The key variables in this study are accuracy and processing speed of the facial recognition system. Accuracy will be quantified as the ratio of correct identifications by the system, with a confusion matrix providing additional information on true and false positives and true and false negatives. Processing speed will be calculated through latency, the time taken from image acquisition until the decision from the recognition algorithm to provide a clear measure of system lag. The cloud latency will be compared with the edge latency for a ttest to determine statistical difference.

C. Rationale Behind the Design

This design aligns well with determining if and how edge computing can be used to improve facial recognition systems of home security. Additionally, using a component-oriented method of facial recognition allows for definitively determining the reliability of SVM-based systems when facial orientation is varied. By comparing edge and cloud computing, this experiment can provide a more nuanced view of the trade-offs between local and centralized computing in terms of both speed and accuracy

D. Alignment with Research Question

The design fits well with the main research question as it provides a framework for systematically evaluating the potential of edge computing for person profile recognition, an essential task of home surveillance contexts. Through this comparison, it can allow for a more informed decision on edge computing and ultimately development decisions for surveillance applications.

VII. Sample Selection and Data Preprocessing

A. Sample Selection

The selected sample is gathering footage myself, which would be beneficial for this experiment because it allows control over multiple key factors: First, recruiting various participants to be in the footage. I would ensure that the cast of volunteers vary in skin tone, age, facial features, and accessories the people wear. This diversity is critical for assessing the SVM's AI potential to generalize. Second, recording original footage grants the experimenter the ability to authorize some volunteers' faces and simulate an unauthorized intruder for others. This distinguishment will allow testing the rate of false positives and false negatives subject to the facial recognition system testing. Since these impact the reliability of the identification process, the approach is instrumental for the system's reliability evaluation. Third, the footage would record the volunteers' from different poses and angles. This later will provide a dataset to test the model's robustness to unpredictable angles and its capacity to work under various capture scenarios. These key factors are critical for the SVM AI model's application in real life. I had 8 classmates participate in my recording and I chose 17 video clips of 8 celebrities videos from youtube. 4 of the 17 videos were interview or award acceptance speech clips where the celebrity's face is clear and they look left, right, and center to train for angle variety. Another 4 of those videos are for testing, where those same 4 celebrities are now moving back and forth with their face somewhat in frame but mimics realistic activity that security cameras would expect. 4 of the videos are of classmate participants frontal face profiles and 5 videos are of them walking in and out of a doorway for testing.

B. Data Preprocessing Methods using OpenCV

Feature Extraction: OpenCV [1] is used due to its robust algorithm capabilities for detecting and extracting facial features. Its acclaim feature detection algorithms will capture images and identify essential elements of the face necessary for the component-based SVM facial recognition process, including the eyes, nose, nose bridge, mouth, and eye brows. Normalization and Standardization: OpenCV will take all the facial "features" resulting from the feature extraction process and normalize/textualize them. Each face will be standardized to ensure they are scaled and angled the same way. This aspect will improve the SVM model's ability to recognize different remotely aligned faces and facial expressions. Data Quality Control: Preprocessing will also entail several quality checks including contrast adjustments, noise reduction, to ensure the facial images meet high-resolution standards appropriate for recognition.

VIII. Method

Software OpenCV and Dlib: OpenCV and Dlib [2] libraries are used for face detection and feature extraction. OpenCV supports basic image processing, and Dlib helps to detect facial landmarks accurately. These are important for analyzing facial images accurately in facial recognition.
Python with Face Recognition Library: Python programming is used as a base language because of its relative ease to code and popularity in data science. The Face Recognition library uses deep learning on unfamiliar data to recognize faces. Python with the Face Recognition library, Dlib face detector and ResNet50 feature extractor for initial svm training, helps in processing facial data.
SVM in Scikit-Learn[3]: Scikit-Learn is used for SVM classifier implementation, and Scikit-Learn also provides a library of machine-learning types through programming with Python. Scikit-Learn offers robust SVM functionality that's critical for the classification and analysis of facial features.
SQLite Database: Used For local data storage and management. This database offers simplicity and effectiveness in storing and retrieving facial embeddings, providing a lightweight solution for the edge computing aspect of the testbed.

TestBed Configurations
Edge Computing Testbed Configuration: The edge device, in this case, a laptop, will be configured with a high quality camera to capture live footage. Then process the pre recorded video through the OpenCV library which opens the stored video and processes it frame by frame to simulate real-time processes.This allows for testing and evaluation of the functionality of the system in processing, accuracy and speed of facial recognition in the hard edge computing environment.

Cloud Computing Testbed Configuration: In the cloud environment, a virtual machine running on Amazon's AWS Platform with a Cloud EC2 instance ID [4] will have python file with similar facial processing functionality to Edge processing. This cloud instance is accessed through SSH tunneling of local Jupyter Code block that extracts embeddings and sends it to cloud python file to return a prediction.

IX. Experimental Procedure

A. Hardware: Preparation and setup

* Install or setup the OpenCV, Dlib library, and the SVM classifier on the edge device which is a laptop on the cloud-based virtual machine on Google Cloud Platform. * Install the stack browser versions and configurations of software used to run the experiment in both setups. * Set up the SQLite database on the edge device for storing faces embeddings Data collection.

B. Data collection

* Using a high quality camera, record 5 videos from 10 seconds to 1 minute 30 seconds, which consists of volunteers who walk in and out of door way in various head poses, angles, lights is taken when the video is designed to be a home security video. * Transfer the input video to the two platforms to enable each to access an analysis dataset and process data pre-process the dataset.

C. Data preprocessing

* Make 3 directories to hold videos. One directory has authorized faces, and the second one is unauthorized faces. These two are for training. The third directory is for testing videos. * Use OpenCV to pre-process the video frame and extract the face faces in the frame. with the Dlib software library and provide APIs that draw faces.

D. Data processing

* Feature extraction will generate a numerical vector (which we shall refer to as embeddings) of the detected face using the facial recognition library. * Store these embeddings into the sqlite database. Each training videos has one person per video so the embeddings would be stored under the profile id of the mp4 file name. Also if the video from the authorized directory, then it recieves a label '1' meaning authorized, and if the video is from the unauthorized directory then it recieves a label '0' for unauthorized. This allows my SVMs to perform binary classification.

E. Model training

* Calibrate hyperparameters for training svms. Gridsearch was used to find the C value that optimized accuracy for each component to train with. * Train the SVM classifier on your sqlite database of extracted embeddings and save the trained models as pkl files. *Perform cross validation to determine decision weights for each component based on accuracy metrics (reliability).

F. Testing

* Use recognition logic to process through test video directory. * Write the predictions into a alert summary text file that shows predictions and latency *Set up cloud instance and upload svm pkl model files to cloud along with python processing file for testing. Remote access the server and send extracted embeddings to cloud and have it predict recognition and return its prediction. *Write predictions to new cloud alert summary text file.

G. Performance evaluation

* Measure processing time for each prediction from initial face detection event to the result. Can use a simple python timer. The alert summary file should have the recognition status with svm's confidence, and the latency. *Parse text summaries to get values and then perform statistical analysis. * Confusion matrix to assess accuracy of facial recognition, recording occurrences of true positives, false positives, true negatives, and false negatives.

H. Comparison

* Conduct all above steps on the edge and the cloud platforms to evaluate the platform performance using the same video and profile to perform comparison.

I. Data analysis and documentation

Data analysis and documentation. * Analyze the data collected to determine efficiency and accuracy of facial recognition on edge vs cloud computing. * Report of findings, highlighting any anomalies or error during the experiment.

J. Literature Comparison

The referenced literature focus on how component-based facial recognition is more accurate compared to global models, especially when it comes to pose changes. A similarity between the literature and my study would be the focus on evaluating robustness of models against pose variance, because of its relevance to real world unpredictability. I am also borrowing from the recognition logic that component based SVMs use, where the components have a voting system in which the most accurate components have the highest weights. This guided the logic for my recognition functions. It is important to note that since this study came out in 2003, I would have access to extraction features that are more modern, which differentiates my procedure from theirs. On the other note, while the literature compares different SVM facial recognition models within the same computational environment, my study focuses on one SVM mode and compares its performance between edge and cloud platforms. Thus, my experiment extends the literature from comparison of model performances to real-life changeability in different computational platforms. [5]

TABLE I
INITIAL SVM ACCURACIES FOR FACIAL COMPONENTS

Component	Accuracy
Left Eye	0.87
Right Eye	0.86
Nose Bridge	0.99
Full Nose	1.00
Mouth	1.00
Left Eyebrow	0.99
Right Eyebrow	0.90

X. TRAINING RESULTS

The initial training accuracies demonstrates that svms are indeed strongly capable of distinguishing facial features through embeddings. Particularly the Nose Bridge, Full Nose, and Mouth—all achieving near or perfect scores. Lower accuracies in eye-related components suggest room for improvement, possibly improving my preprocessing steps or collecting higher quality data.

XI. ROC CURVES OF TRAINED SVMS

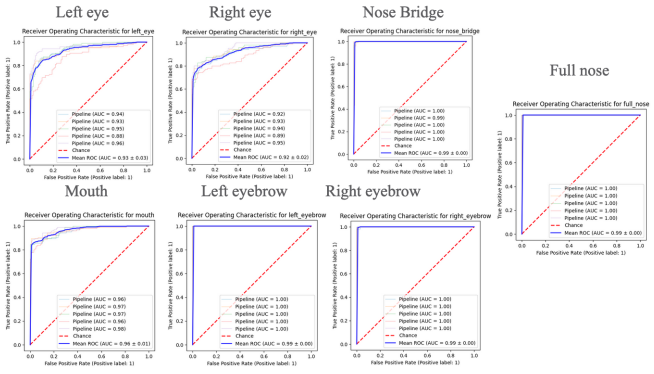

Fig. 1. ROC Curves of the Trained SVMs.

The Receiver Operating Characteristic (ROC) curves for our trained SVMs exhibit high Area Under the Curve (AUC) values, closely mirroring the findings from figure 7 of B. Heisele et al. (2003), where the component based accuracy was consistently reported above 90 percent. ROC curves help us to visually assess the trade-off between sensitivity (true positive rate) and specificity (false positive rate) at various threshold settings. This is particularly important in facial recognition where the cost of a false negative can be as critical as a false positive. The high values suggest that the svms can effectively distinguish between the classes (authorized vs unauthorized users), which is essential for maintaining security integrity in real-time surveillance systems.

XII. 5-FOLD CROSS-VALIDATION

5-fold cross-validation helps predict how the component svms will perform on unseen data, which is crucial for a real-world surveillance application. The results varied across components, which implies that some of my trained components are better suited than others to generalize beyond the training

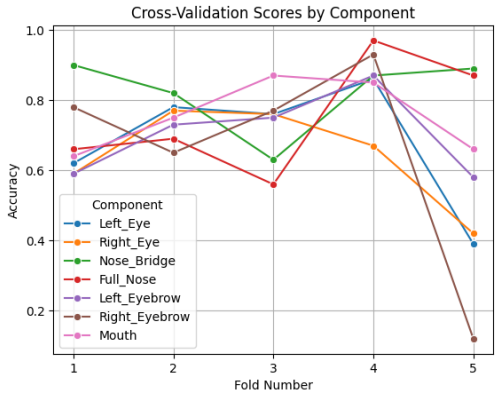

Fig. 2. 5-Fold Cross-Validation Results.

dataset. For example, while the Nose Bridge component had an average CV score of 0.82, suggesting good generalization, other components like the 'Right Eyebrow' performed worse with an average score of 0.65. Components that performed worse resulted in their weights lightened for the decision making process to make the overall model more effective.

XIII. SVM Testing Confidence Scores

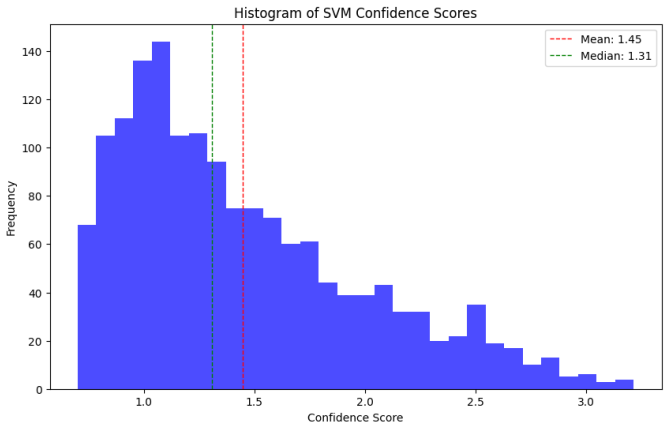

Fig. 3. Histogram of SVM Confidence Scores

These are maximum confidence scores of all components from each processed face. This allows me to gaugue an appropriate threshold to filter out low confidence, unrecognized embeddings. High confidence scores indicates a stronger belief SVM's decision, which is good for security-sensitive real-time surveillance. I set my svm confidence threshold slightly below these maximum confidences to consider accepting an optimal degree of certainty before it risks false positives while maintaining an acceptable recognition rate for true positives.

The right-skewed distribution suggests that generally the models are moderately confident in its predictions, but a few decisions with extremely high confidence suggest overfitting on certain features which could lead to false positives.

```
Confusion Matrix:
[[  3   0   0   0   0   0   0   7   0]
 [  1  32   0   0   2   0   1  36   0]
 [  0   0 214   0   1   2   0 164   0]
 [  0   0   0  25   0   0   0  20   0]
 [  0   0   0   0   0   0   0   0   0]
 [  3   0  18   0  14 121   0  54   0]
 [  2   1   1   2   3   0 139 115   0]
 [ 19   2   4   4  17   1  27 500   1]
 [  0   0   2   0  31   1   2 451 449]]

Classification Report:
              precision    recall  f1-score   support

  auth_face8       0.11      0.30      0.16        10
  auth_face4       0.91      0.44      0.60        72
  auth_face3       0.90      0.56      0.69       381
  auth_face7       0.81      0.56      0.66        45
  auth_face6       0.00      0.00      0.00         0
  auth_face1       0.97      0.58      0.72       210
  auth_face2       0.82      0.53      0.64       263
unauthorized       0.37      0.87      0.52       575
  auth_face5       1.00      0.48      0.65       936

    accuracy                           0.60      2492
   macro avg       0.65      0.48      0.52      2492
weighted avg       0.81      0.60      0.63      2492
```

Fig. 4. Confusion Matrix and Classification Report.

XIV. Confusion Matrix and Classification Report

The confusion matrix and classification report for our facial recognition system reveal several critical insights into the performance of our SVM classifiers across different facial components. Here, I'll delve into what the confusion matrix and classification metrics suggest about our system's precision, recall, and overall accuracy.

Confusion Matrix Analysis: Notable observations: - High misclassification rates for face 5 being falsely identified as 'unauthorized' more time than recognized, suggesting a sensitivity issue, possibly indicating either a high variability within this class or similarity in features with unauthorized faces. - The model performed relatively well in isolating face 1 and face2, with correct recognition rates of 121 and 139, though both still faced substantial misclassification like the rest of faces.

Classification Metrics Interpretation: The classification report provides a deeper insight into the effectiveness of our model: - **Precision and Recall**: faces 1 and face 4 yield high precisions, 0.97 and 0.91, indicating a high reliability in the model's predictions when it classifies a face as belonging to these categories. Conversely, both their recall rates 0.58 and 0.44, respectively suggest that the model misses a significant number of true cases. This could mean my decision threshold could be slightly too high. - **Overall Accuracy**: The results return an overall accuracy of 60 percent, which means my

svms recognition demonstrates capibility but not yet reliability to depend on it for security due to inconsistent accuracy depending on person and video quality. I would attribute this to overfitting a limited training dataset.

XV. Recognition Intervals in Video Analysis

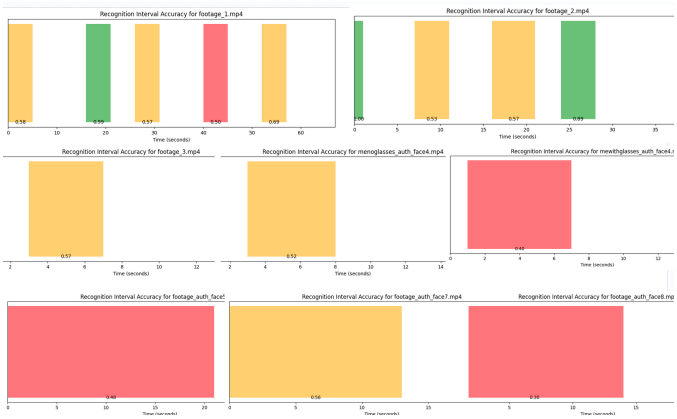

Fig. 5. Recognition Intervals of All Videos in a Condensed Grid Display.

Notable observations: - The accuracy comparison for face 4 going from 0.52 without glasses to 0.4 with glasses suggests that the svm models, although it tries to be robust to slight alterations of face, accessories have a significant impact on recognition as expected. - The first (top) two videos had the most participants, whereas the last 6 were individual people. The results from the first two suggest that some faces are more likely to be recognized than others, which suggest that my recognition is limited in applicability to diverse appearances.

XVI. Comparison of Latencies

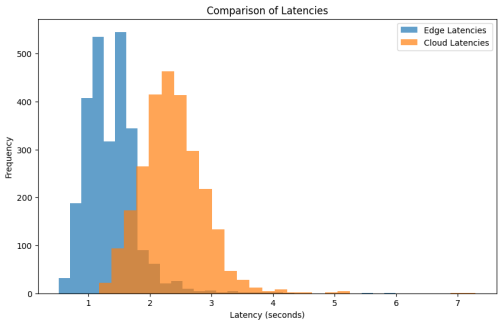

Fig. 6. Latency Comparison Between Edge and Cloud Implementations.

Quantitative Analysis: The latency for edge computing has a calculated mean of 1.36 seconds an a standard deviation of 0.43 seconds. The distribution has a shape shifted towards well below the 2-second threshold. The distribution for the cloud-based system presents a mean of 2.37 seconds and a standard deviation of 0.52 seconds. The population is not only less clustered but also smeared over higher values, extending well beyond the 3-second-latency mark. Statistical Significance:

The calculated t-statistic of -76.81 and a p - value of 0.0000. These two factors indicate that the observed differences in latency are not purely random and are statistically significant. Thus, edge computing is indeed faster than cloud computing in processing terms.

XVII. Discussion on Practical Implications

: add here Discussion on Practical Implications

The results support that there are certain differences in terms of latency between edge and cloud computing. Evidently, the revealed lower latency characterizing edge computing suggests that the latter can be beneficial within the application of real-time surveillance. Therefore, the outcomes of the analysis suggest that deciding whether to use edge or cloud computing can be of paramount importance to operations management, implying that the organization should make a choice based on the initially set goal, i.e. whether local processing should be preferred to urgency or comprehensiveness.

However, the fact that the results reveal discrepancies between the recorded values of latency and recommend edge computing times of near instant supposes the need of strong local computing solutions. The model should also arrive at a decision concerning the cost-to-benefit weighs of this approach. Overall, the findings made suggest the viability of component svm recognition in edge computing use cases in which local response is needed urgently.

XVIII. Conclusion and Future Research

: The outcomes of the research demonstrate the exceptional potential of using SVM-based facial recognition systems to achieve the high recognition of components, especially particular facial features. The significance of the approach to using SVM as the foundation for facial recognition systems can be seen. However, the fact that the approach was limited in terms of generalization and provided a range of outcomes with different components and the significance of accessories also means that certain amendments can be made to the current approach.

The following project may focus on increasing the generalization of SVM classifiers so that they can be successfully used with other features. Another project idea is connected to the applicability of the recognized approach to edge computing since the latter will be in demand, and the approach to recognizing components must be improved.

As a means of enhancing the current component recognition system in the long-term, one can reflect on creating a hybrid model combining SVM with a deep learning framework. In terms of future research, studies of the application of the studied approach to other components and more complex scenarios can be provided. Overall, as the technology advances, the implementation of edge computing solutions must be studied within a broader framework.

References

[1] [1] Bradski, Gary. "The OpenCV Library." Dr. Dobb's Journal of Software Tools, 2000, www.drdobbs.com/open-source/the-opencv-library/184404319.

[2] [2] King, Davis E. ”Dlib-ml: A Machine Learning Toolkit.” Journal of Machine Learning Research, vol. 10, July 2009, pp. 1755-1758. jmlr.org/papers/v10/king09a.html.
[3] [3] Pedregosa, F., et al. ”Scikit-learn: Machine Learning in Python.” Journal of Machine Learning Research, vol. 12, Oct. 2011, pp. 2825-2830.
[4] [4] ”Google Compute Engine: Virtual Machines (VMs).” Google Cloud, cloud.google.com/compute.
[5] [5] Heisele, Bernd, et al. ”Face Recognition: Component-Based Versus Global Approaches.” *Computer Vision and Image Understanding*, vol. 91, no. 1-2, 2003, pp. 6–21, doi:10.1016/S1077-3142(03)00073-0.