Celal Bayar University Journal of Science
Volume 22, Issue 2, 2026, p 269-283
Doi: 10.18466/cbayarfbe.1621669
Research Article

Celal Bayar University Journal of Science

AR and ML Solutions for Facial Expression Recognition on Mobile Phones

Burcu Selçuk[1]*, Tacha Serif[2]

[1,2] Yeditepe University, Department of Computer Engineering, 34755, Istanbul, Türkiye
* bselcuk@cse.yeditepe.edu.tr
* Orcid No: 0000-0002-9200-6017

Received: January 16, 2025
Accepted: March 4, 2026
DOI: 10.18466/cbayarfbe.1621669

Abstract

The widespread use of smartphones has led to the development of a diverse range of personal and business applications. Especially, ever-improving computational power of smartphones has made real-time video processing operations feasible on mobile and portable devices. Consequently, real-time facial expression recognition (FER) on mobile phones has emerged as an efficient approach, despite certain limitations imposed by hardware and power storage limitations. An example of such implementation is the ubiquitous "Face Unlock" feature utilized in numerous smartphones and applications, including online banking. Facial expression recognition that only relies on Machine Learning (ML) faces challenges; such as the requirement for substantial datasets and expertise-level training to achieve accurate models. Accordingly, this study explores an alternative approach which involves Augmented Reality (AR) to achieve expression recognition. To achieve this goal, this study undertakes a comprehensive comparison of these two distinct techniques through the development of separate mobile applications for facial expression recognition. Experimental results indicate that the ML-based architecture achieves the highest classification accuracy of 91.63%, outperforming the AR-based approach (81.63%). However, from an operational efficiency viewpoint, AR-based approach demonstrates superior performance with an average inference latency of 37.8 ms, operating approximately 7.6 times faster than ML application (288.2 ms). Furthermore, environmental stress testing reveals that while RGB-based ML models suffer performance degradation in low light, the AR system maintains consistent tracking in total darkness due to active infrared depth sensing. The findings shed light on the advantages and limitations of each method, contributing to the advancement of FER technology in mobile applications.

Keywords: ARKit, augmented reality, facial expression recognition, machine learning, mobile

1. Introduction

Facial expressions play a critical role in nonverbal communication, conveying emotional states universally recognized across cultures, such as anger, happiness, and surprise. Beyond human interaction, Facial Expression Recognition (FER) has become a vital tool in security, education, and marketing, allowing systems to objectively assess emotional cues for applications ranging from fraud detection to student engagement

Accordingly, the applicability of FER extends across a wide range of domains, including psychology, education, marketing, and business negotiations [1, 2, 3]. In educational settings, for instance, instructors often rely on students' facial expressions to assess comprehension, engagement, or confusion, which is an inherently time-consuming and cognitively demanding process [2]. Similarly, in marketing, consumer reactions to advertisements can be inferred by analyzing facial expressions as viewers engage with promotional content. In both cases, automated FER systems offer the potential to capture such feedback more efficiently and objectively than human observation alone.

While traditional FER systems rely on powerful desktop processors, recent advances in mobile technology have created new opportunities for real-time analysis on smartphones. However, deploying FER on mobile platforms presents unique challenges, including limited computational resources, power constraints, and hardware restrictions. Consequently, mobile recognition

methods must prioritize computational efficiency without sacrificing accuracy.

Accordingly, in this study, three methods for implementing facial expression recognition on iPhone Operating System (iOS) devices are proposed by leveraging the computational capabilities of modern smartphones to perform efficient and real-time emotion analysis. This approach aims to demonstrate the practicality of deploying FER systems on mobile platforms, enabling broader accessibility and integration into diverse application domains.

Just as individuals may express the six basic emotions in distinct ways, FER systems can be implemented using a variety of methodological approaches, each presenting unique strengths and limitations. When deployed on mobile platforms, FER algorithms must operate under strict constraints, including limited computational resources, reduced power availability, and hardware restrictions. Consequently, mobile-based recognition methods must prioritize computational efficiency while maintaining acceptable accuracy. To address these challenges, this study explores three distinct approaches and evaluates their suitability for mobile FER applications.

The first approach leverages ARKit, a framework specifically designed for iOS devices, to perform facial expression recognition using Augmented Reality (AR) and the TrueDepth camera. This method analyzes facial landmark movements and compares them against predefined expression descriptions to infer emotional states. AR has gained considerable popularity due to its ability to process complex visual tasks efficiently and in real time. When combined with high-level facial information, AR-based solutions offer a lightweight and privacy-conscious alternative, as facial geometry can be processed without storing raw image data [4, 5]. Although AR is frequently associated with overlaying virtual elements onto real-world scenes [6], in this study AR is employed strictly for recognition purposes, without visual augmentation.

The second approach follows a more traditional machine learning-based pipeline, utilizing the You Only Look Once (YOLO) architecture for facial expression recognition. This method consists of two primary phases: training and deployment. During training, a dataset containing images of individuals demonstrating the six basic facial expressions is collected and labeled. Bounding boxes are annotated around the facial regions corresponding to each expression, and the model is trained to classify regions accordingly. The trained model is then integrated into an iOS application, facilitating facial expression recognition on both still images and real-time video streams.

2. Related Work

The implementation of real-time facial expression recognition encompasses a wide array of approaches, as it must address numerous environmental and physiological challenges, including variations in illumination, camera resolution, head pose, and occlusion. Bayesian Networks, which model probabilistic relationships between facial action units, have been widely explored and have demonstrated promising performance in FER tasks [7-10]. In parallel, Convolutional Neural Networks (CNNs) have become increasingly dominant due to their strong performance in visual feature extraction and pattern recognition [3, 11-14]. Last but not least, CNN-based approaches have also shown robustness in scenarios involving partial occlusion, such as facial expression recognition under mask-wearing conditions observed during the COVID-19 pandemic [15, 16].

Another robust method for facial expression recognition involves analyzing movements of facial landmarks. Some studies focus on mouth movements [17], while others rely on eyebrow movements [7] or eye-related features [18] for recognition. Furthermore, certain approaches emphasize pattern recognition rather than solely relying on facial landmark movements [19, 20, 21]. Two studies [20, 21] employ rendered facial masks to extract expression-related patterns for classification. Similarly, Hsu and Jain [22] propose a 3D face modeling technique that represents facial structures as mask-like geometries, achieving encouraging results on static images. Lastly, Park and Oh [23] further explore mask-based representations using AR in their recognition system.

Despite these advances, facial expression recognition on mobile devices remains relatively underexplored, primarily due to computational and energy constraints. Alshamsi et al. [24] address these limitations by combining traditional machine learning with image processing techniques in a mobile computing environment. Their approach extracts facial features and applies the K-Nearest Neighbor (KNN) classifier, achieving an accuracy of 89.5% on the Cohn–Kanade (CK) [25] dataset.

Similarly, Yoon and Kim [26] propose a CNN-based FER model optimized for mobile deployment. By carefully adjusting the width and depth of the network's bottleneck layers, the authors achieve a balance between computational efficiency and recognition accuracy. Their model reports an average accuracy of 86.6% across seven expression classes on the RAF [27] dataset.

3. Materials and Methods

This section describes the comprehensive technical framework and experimental design employed to evaluate the proposed FER systems. Furthermore, it outlines the data preparation strategies, training protocols, and the specific metrics established to ensure a rigorous comparison of detection performance and real-time latency on iOS hardware.

3.1. Methodology

Existing research has made significant progress in improving the robustness of facial expression recognition systems; however, most studies focus on static image-based recognition, as real-time analysis introduces substantial computational and environmental challenges. Recent advances in 3D-sensing camera technology, particularly their improved quality and accessibility, offer promising solutions by providing rich geometric information suitable for computer vision tasks. Thus, this study combines the iPhone TrueDepth camera with AR and compares it with ML techniques to identify the most efficient approach for real-time FER on mobile devices.

As previously mentioned, Augmented Reality allows the integration of 2D or 3D elements into the user's environment, creating an appearance as if these elements are part of the user's real world. ARKit [28] is a framework for facilitating AR experiences, combining camera capture with device motion tracking and scene processing. ARKit integrates Visual-Inertial Odometry (VIO) to track the device's position and orientation [28]. Unlike purely RGB-based solutions, this approach leverages the TrueDepth camera system found in modern iOS devices. The TrueDepth camera projects over 30,000 infrared dots onto the user's face to generate a depth map, which is processed alongside the infrared image by the device's Neural Engine. This hardware-accelerated tracking allows for precise detection of facial muscle movements (Action Units) even in low-light conditions, eliminating the need for computationally expensive depth estimation algorithms. Notably, this approach requires only one RGB sensor and an Inertial Measuring Unit sensor, eliminating the requirement for depth sensors or multiple cameras on the device.

The process described above consists of three main stages. The first stage, Motion Tracking, integrates visual data obtained from the RGB camera sensor operating at 60 frames per second with inertial data from the gyroscope and accelerometer sampled at 1000 frames per second. By applying the parallax principle, ARKit identifies and tracks high-contrast feature points across successive frames. This information enables the system to estimate the position and orientation of a virtual camera with six degrees of freedom (DoF), representing translational and rotational movement within three-dimensional space. The second stage, Scene Understanding, focuses on grouping feature points that lie on the same geometric plane, thereby enabling the detection and characterization of planar surfaces within the environment. This spatial understanding provides structural context necessary for accurate tracking and interaction. Finally, the third stage, Rendering, is automatically managed by ARKit in conjunction with the SceneKit framework, which is responsible for visualizing virtual content within the reconstructed scene.

In contrast, the ML-based approaches aim to enable computers to perform specific tasks with minimal explicit programming, relying on data evaluation and decision-making based on experience and knowledge. ML requires a dataset for training, allowing the machine to evaluate data and act with reduced human intervention. Historically, Supervised Learning has been the standard, utilizing algorithms such as Support Vector Machines (SVM) to construct hyperplanes for classification, and K-Nearest Neighbor for instance-based learning. While methods such as Decision Trees offer interpretability by capturing non-linear relationships, they often struggle with overfitting and stability when applied to complex image data.

To overcome the manual feature extraction required by the traditional methods, DL, specifically Convolutional Neural Networks (CNNs), has become the dominant approach. Unlike traditional ML, CNNs automate feature extraction through convolutional layers that detect low-level features and pooling layers that reduce dimensionality as depicted in Figure 1.

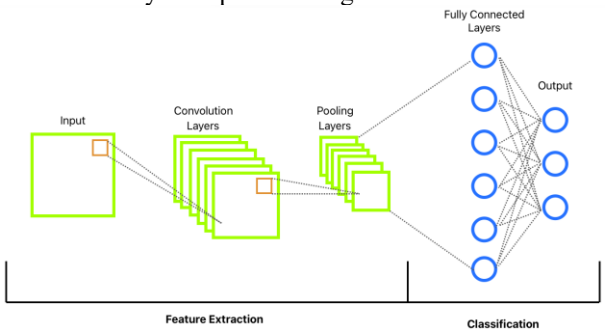


Figure 1. CNN sequence diagram.

For real-time applications, object detection architectures have further evolved. While region-based models like Regional CNN (R-CNN) offer high accuracy, it is computationally expensive. This study therefore utilizes the different versions of the YOLO architecture, which frames object detection as a single regression problem, allowing for significantly faster inference speeds suitable for real-time mobile use by extending the capabilities of R-CNNs. As the name suggests, YOLO examines the given image only once, performing a single forward propagation through the neural network to make predictions. YOLO employs a convolutional

neural network to analyze the entire image and then divides it into bounding boxes with corresponding probabilities for each region. These bounding boxes are weighted based on the predicted probabilities, and the recognized objects are outputted with their respective bounding boxes and confidences. YOLO's appeal lies not only in its real-time performance but also in its ability to learn and generalize objects effectively. After training a YOLO model with a dataset, it can outperform other algorithms in both timing and accuracy when tested with previously unseen data. YOLOv7, the latest addition to the YOLO algorithms, stands out for its simultaneous performance of instance detection and segmentation in a relatively fast manner compared to other YOLO architectures. It surpasses all other object detection algorithms in both speed and accuracy [29].

3.1.1. Datasets

In the existing literature, several well-established datasets have been utilized in various studies to achieve different objectives. Datasets play a crucial role in ML as the performance of a model heavily depends on the quality and diversity of the data it is trained on. Consistency across data is essential, and datasets ensure a standardized scheme throughout.

One of these widely used datasets is the Japanese Female Facial Expressions (JAFFE) [30] dataset, which serves as a prominent resource for FER tasks. This dataset includes images of 10 different women displaying 7 distinct facial expressions several times, resulting in a total of 213 images. As illustrated in Figure 2, the images in the JAFFE dataset are exclusively in black and white, and all subjects are women of similar age and ethnicity.



Figure 2. A sample from the JAFFE dataset [30].

In addition to the JAFFE dataset, the Extended Cohn-Kanade (CK+) [25] dataset is another widely used dataset. This dataset comprises images collected from diverse video sequences and includes 123 different subjects displaying 7 distinct facial expressions. The subjects in the CK+ dataset belong to different genders, ethnicities, and age groups. The dataset contains both black and white and RGB images.

FER-2013 [31] is another prominent dataset in the Facial Expression Recognition domain. It stands out as one of the largest datasets available for training, with an extensive collection of 28,709 images. However, while the dataset offers a vast quantity of data, the image quality is limited, with all images being just 48x48 pixels and presented in grayscale. As shown in Figure 3, the faces in FER-2013 are consistently centered and occupy a similar space in each image with different angles of rotation.



Figure 3. A sample from the FER-2013 dataset [31].

Another dataset appropriate for Facial Expression Recognition is the Static Facial Expressions in the Wild (SFEW) dataset [32]. Each image in the SFEW dataset is labeled with one of the 7 facial expression categories, and the images are colored, featuring people of various genders, ethnicities, and ages. While the training and validation datasets are publicly available, the testing datasets are restricted from general use.

3.1.2. Model Evaluation Metrics

Evaluating an ML model during training or after completing the training process is essential for estimating its performance and making necessary adjustments. Furthermore, selecting the appropriate evaluation metric is crucial, as certain metrics may provide a sense of higher accuracy than reality.

The confusion matrix serves as a fundamental evaluation metric, representing the model's outputs in a matrix format and providing an overview of its performance. The x-axis represents the actual values, while the y-axis represents the predicted values. The Confusion Matrix calculates True Positives (TPs), True Negatives (TNs), False Positives (FPs), and False Negatives (FNs), which are vital for determining metrics like recall, precision, and accuracy of the ML model. TP indicates the number of correct predictions for the positive class, TN represents the number of correct predictions for the negative class, FP corresponds to the number of incorrect predictions for the positive class, and FN denotes the number of incorrect predictions for the negative class.

Precision and Recall are essential metrics derived from the Confusion Matrix using TP and FP. Precision reflects the ratio of TP to the total number of predicted positives (TP + FP), demonstrating the model's

precision in positive predictions. The precision formula is shown in equation 3.1. Recall, on the other hand, indicates how well the model identifies positive classes from the ground truth and is the ratio of TP to the total number of actual positives. The recall formula is represented in equation 3.2.

Intersection over Union (IoU) is another well-known metric used to quantify the overlap between two bounding boxes: one representing the ground truth and the other the prediction. The IoU value ranges between 0 and 1, where a value of 1 indicates a perfect prediction with complete overlap, while 0 indicates no overlap at all. IoU is calculated by taking the ratio of the intersection area to the union area of the two bounding boxes as shown in equation 3.3.

The F1 Score, introduced in equation 3.4, finds the optimal confidence score threshold where precision and recall yield the highest F1 score, thus striking a balance between the two metrics. A high F1 score indicates high precision and recall, while a low F1 score suggests one of them is low and should be the prioritized metric while dealing with imbalanced datasets.

$$Precision\% = \frac{TP}{TP + FP} * 100\% \tag{3.1}$$

$$Recall\% = \frac{TP}{TP + FN} * 100\% \tag{3.2}$$

$$IoU = \frac{IntersectionArea}{UnionArea} \tag{3.3}$$

$$F1 = 2 * \frac{P * R}{P + R} \tag{3.4}$$

Average Precision (AP) is another noteworthy metric in ML models, representing the area under the precision-recall curve and summarizing the P-R Curve to a single scalar value. AP is high when both precision and recall values are high and low when either of them is low across various confidence threshold values. AP ranges from 0 to 1. Additionally, the Mean Average Precision (mAP) is computed by averaging the APs across all classes involved, providing an overall performance measure for object detection models or algorithms. The AP formula is presented in equation 3.5, and the mAP formula is shown in equation 3.6, where AP_k represents the AP of class k, and n denotes the number of classes.

$$AP = \int_0^1 p(r)dr \tag{3.5}$$

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k \tag{3.6}$$

To strictly evaluate the performance of the proposed FER systems on resource-constrained mobile hardware, this study employs a set of metrics chosen to balance spatial accuracy, classification correctness, and real-time performance.

Mean Average Precision (mAP) and Intersection over Union (IoU) are selected as the primary metrics for the ML-based approaches. Since YOLO architectures perform simultaneous object localization and classification, it is vital to verify that the model is detecting the facial region accurately before classifying the expression. IoU quantifies the overlap between the predicted bounding box and the ground truth, ensuring that the expression analysis is performed on the correct region of interest.

For evaluating the classification facial expressions, on the other hand, Precision, Recall, and the F1-Score are utilized as they are necessary for FER tasks to penalize false positives, which is crucial given the morphological similarities between certain expressions (e.g., distinguishing 'Fear' from 'Surprise'). A Confusion Matrix is further employed to visualize these specific inter-class misclassifications and analyze the model's discriminative power across different emotional states.

3.2. Experimental Setup

In order to evaluate the comparative effectiveness of the AR-based and ML-based approaches, a series of experiments are designed to measure classification accuracy, inference latency, and robustness to environmental variations. Firstly, the systems are evaluated under two distinct lighting conditions to test the robustness of the sensors:

- **Low Light (50 lux):** Simulating dim indoor environments to evaluate the dependency on RGB data versus infrared depth sensing.
- **Normal Light (300 lux):** Simulating the standard environments to establish a baseline performance.

Furthermore, to determine if performance is hardware-dependent, evaluations are conducted on two distinct iOS devices: an **iPhone 11** and an **iPhone 13**. This comparison aims to isolate the impact of processor generation (A13 Bionic vs. A15 Bionic) on the inference time of the YOLO models compared to the ARKit implementation.

Last but not least, the models are also assessed using the metrics defined in Section 3.1.2 on both a static test set derived from the CK+ and JAFFE datasets, and a real-time live evaluation involving multiple subjects.

Celal Bayar University Journal of Science
Volume 22, Issue 2, 2026, p 269-283
Doi: 10.18466/cbayarfbe.1621669

4. Implementation

In this study, three facial expression recognition techniques are implemented, focusing on two ML-based approaches using the YOLOv5 and YOLOv7 architectures and an AR approach using ARKit.

To ensure a baseline comparison between the geometry-based ARKit approach and the image-based YOLO models, this study utilizes the JAFFE [30] and CK+ [25] datasets combined. These datasets are specifically selected over larger datasets (e.g., FER-2013) because they feature high-resolution, posed expressions with clear prototypical facial deformations. Since the ARKit approach relies on detecting specific facial muscle movements (Action Units), using controlled datasets minimizes environmental noise such as extreme occlusion or motion blur as a confounding variable which allows for a fair assessment of the architectural differences between the AR and ML methods under ideal conditions before introducing real-world variables.

The JAFFE dataset contributes 213 grayscale images of Japanese female subjects, providing a consistent morphological baseline. To introduce demographic variance, the CK+ dataset is integrated, adding 593 images across diverse age groups, genders, and ethnicities. By combining these sources, the training set balances the controlled inputs with demographic diversity, ensuring the ML models are not overfitted to a single facial topology.

Annotation is the initial step in data processing, where bounding boxes are drawn around the subject's face displaying one of the seven facial expressions. These bounding boxes are labeled to represent their respective expressions, allowing the model to distinguish between different emotions. The images are then preprocessed to ensure uniformity, resized to 416x416, and auto-oriented. After preprocessing, the images are divided into three sets to facilitate training, validation and testing.

Given the limited size of the combined dataset, data augmentation is employed not merely to expand the training set, but to synthetically model the environmental constraints inherent to mobile deployment. Random rotation (±15∘) is applied to simulate the natural instability of handheld device usage, where users rarely maintain a perfectly vertical device orientation. Exposure adjustment (±25%), on the other hand, introduces photometric variability, compelling the model to learn features robust to the uncontrolled lighting conditions typical of mobile environments. Finally, random noise injection (5% of pixels) mimics the sensor grain and high-ISO artifacts frequently present in front-facing mobile cameras, particularly in the low-light environments. Such transformations prevent the model from overfitting to the pristine studio conditions of the source datasets and enforce generalization to the unconstrained domain of real-time mobile video. Figure 4 illustrates the progression of an image from the dataset through annotation, preprocessing, and noise augmentation, demonstrating the step-by-step data preparation process undertaken for the ML-based approach.

After preparing the dataset and exporting it in the appropriate formats for both YOLOv5 and YOLOv7, the models are trained in the Google Colab environment utilizing GPU acceleration. Both architectures are trained on the identical augmented dataset to ensure a fair comparison. The training configuration was set to 60 epochs with a batch size of 8.

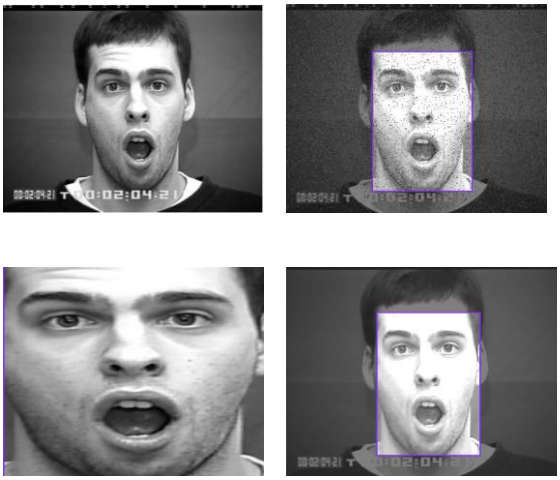


Figure 4. Data preprocessing and augmentation steps on a sample image.

4.1. ARKit Based Solution

The AR-based approach is designed to be a straightforward application that executes in real-time and its purpose is to distinguish between 7 facial expressions, by benefitting ARKit to yield the corresponding prediction at minimum execution time.

Once the application is functioning, the front camera is employed and examines for faces to distinguish emotions from. In this process, the application should first notice the face in its scope. Once the face is detected, a mesh is rendered on top of the subject's facade as a computer-generated element that would check for facial landmark movements. The combination of different facial landmark actions corresponds to distinct facial expressions performed; According to the mesh form, it is expected that the application categorizes the facial expression and associates it with the correct sentiment. Once this process is done, the expression's emoji and its name are revealed to the user.

For the development of the application of this approach, the Swift programming language is utilized. The AR-based application leverages the ARFaceTrackingConfiguration class to detect faces via the TrueDepth camera. Upon detection, an ARFaceAnchor instance is generated, providing real-time tracking of the facial geometry at 60 frames per second. The application utilizes a BlendShape dictionary, which is a set of floating-point coefficients (ranging from 0.0 to 1.0) that represent specific facial muscle movements (Action Units), such as brows' inner part being up or the nose sneering right.

To translate these raw geometric coefficients into emotional classifications, a custom logic layer based on Paul Ekman's Facial Action Coding System (FACS) [33] is implemented. A custom algorithm aggregates the coefficients of relevant muscle groups and triggers a prediction only when their cumulative sum exceeds a confidence threshold of 0.9. For example, while "Happiness" is detected simply by summing *mouthSmileLeft* (the case of mouth's left corner being up) and *mouthSmileRight*, complex negative emotions require monitoring multiple regions of the face simultaneously:

- **Anger:** Is identified by a combination of the brows lowering and eyes squinting. The system triggers "Anger" if the sum of *browDownLeft*, *browDownRight*, *eyeSquintLeft*, and *eyeSquintRight* exceeds the threshold.
- **Disgust:** Is characterized by a nose wrinkle and upper lip movement. The algorithm sums *noseSneerRight*, *noseSneerLeft*, and *mouthShrugUpper*.
- **Surprise:** Is detected when the *jawOpen* coefficient increases in conjunction with *browInnerUp*.

If none of the aggregate sums exceed the threshold, the system defaults to a "Neutral" state. This additive thresholding approach ensures that subtle, involuntary facial twitches do not trigger false positives, requiring a deliberate and distinct expression from the user to register a change in emotional state. Figure 5 presents screenshots captured during a live testing session, demonstrating the system's ability to classify distinct emotional states. The three panels show the successful detection of Happiness, Anger, and Surprise, respectively. In each instance, the specific combination of facial muscle movements (e.g., mouth curvature for happiness, brow lowering for anger) yielded a cumulative coefficient sum exceeding the 0.9 confidence threshold, triggering the immediate display of the corresponding prediction label on the user interface.

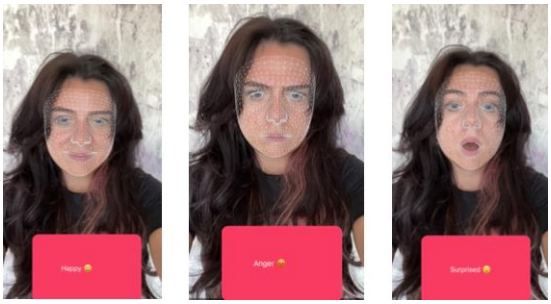


Figure 5. Real-time recognition of facial expressions using the AR-based application.

4.2. YOLOv5 & YOLOv7 Based Solution

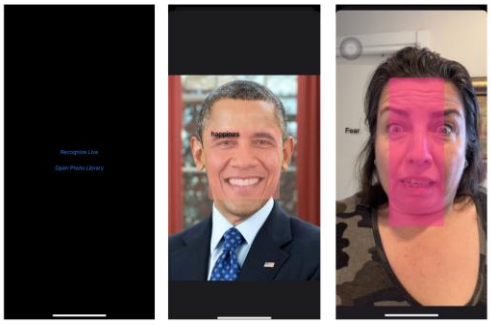
Unlike the AR-based approach which relies on pre-configured hardware logic, the machine learning models require a complete training pipeline. The YOLOv5 and YOLOv7 architectures are chosen due to their balance of accuracy and inference speed, which is critical for mobile deployment.

Training is conducted in a Google Colab environment, utilizing a Tesla T4 GPU to accelerate the training process. In order to overcome the limitations of the smaller dataset, transfer learning from COCO weights is conducted. Both models are initialized with pre-trained weights from the COCO dataset, allowing the networks to leverage previously learned low-level features and converge significantly faster than training from scratch. To ensure a fair comparison, both architectures are trained using identical hyperparameters. The input images were resized to 416×416 pixels to balance the trade-off between feature granularity and inference latency. Furthermore, for optimization, Stochastic Gradient Descent (SGD) optimizer with a momentum of 0.9 and weight decay of 0.0005 is utilized. Both models are trained for 60 epochs with a batch size of 8.

The training process aims to minimize a multi-part loss function consisting of three loss metrics, namely Box Loss, Object Loss, and Classification Loss. Box Loss utilizes a Complete Intersection over Union (CIoU) to penalize bounding box error while accounting for overlap area, center point distance, and aspect ratio. Object Loss, on the other hand, uses Binary Cross Entropy (BCE) to handle the confidence values of object presence. Lastly, Classification Loss penalizes incorrect emotion class predictions using the BCE.
Because iOS devices cannot natively execute raw PyTorch models, trained models are required to be converted into the Apple's CoreML (.mlmodel) format from the PyTorch (.pt) format. Accordingly, 16-bit floating point (FP16) quantization is applied to both trained PyTorch models, which reduces the model size significantly without a significant drop in the mAP, ensuring that the application is lightweight and compatible with the iOS. Furthermore, Figure 6 presents the operational interface of the ML-based application, illustrating its dual-mode functionality. Figure 6 (a)

Celal Bayar University Journal of Science
Volume 22, Issue 2, 2026, p 269-283
Doi: 10.18466/cbayarfbe.1621669

displays the main entry point, where the user selects between two distinct inference pipelines: initiating a real-time video session ("Recognize Live") or processing existing media ("Open Photo Library").



(a) (b) (c)

Figure 6. ML application walkthrough (a) application main menu (b) still image prediction (c) real-time video prediction.

Figure 6 (b), on the other hand, demonstrates the static inference mode, where an image is loaded from the device's local storage. The YOLO model processes the single frame, localizes the face, and overlays a semi-transparent bounding box annotated with the predicted class label (e.g., "Happiness"). Lastly, Figure 6 (c) depicts the real-time inference mode, showing the application processing a live video stream. In this scenario, the system continuously updates the bounding box coordinates and emotion label (e.g., "Fear") as the subject's expression evolves.

5. Testing and Evaluation

In this section, the proposed methods are evaluated under controlled conditions to determine their efficacy. The evaluation is divided into three phases: first, the training metrics of the machine learning models are analyzed to validate their convergence and theoretical performance; second, the deployed ML models are tested in real-world scenarios; and third, the ARKit-based solution is evaluated. Finally, a comprehensive comparison is presented to highlight the trade-offs between the approaches.

5.1. YOLOv5 and YOLOv7 Training Analysis

To validate the machine learning models before mobile deployment, the standard performance metrics generated during the 60-epoch training phase is analyzed. The confusion matrices (Figure 7 and Figure 8) reveal the classification performance of the models on the validation set derived from the dataset.

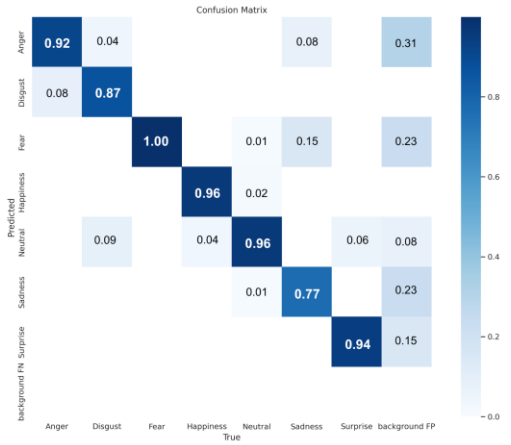


Figure 7. YOLOv5 confusion matrix on the validation set.

The YOLOv5 model successfully classifies 100% of the "Fear" instances in the set. Furthermore, 96% of all "Neutral" instances are classified correctly. The least successfully classified category is "Sadness" (77%), which is confused mostly with the background. This is likely due to the subtle morphological features of sadness; unlike "Happy," which is associated with a distinct smile, or "Surprise," which involves an open mouth and raised eyebrows, "Sadness" is a low-intensity emotion often characterized only by a slight mouth frown or drooping eyelids, making it difficult to distinguish from a neutral face or background noise expression associated with a big smile or "Surprise", often involving an open mouth and certain eyebrow movements, "Sadness" can only consist of a slight mouth frown or drooping eyelids, making its instances harder to classify.

Figure 8 shows the classification performance of the YOLOv7 model on the same test set as YOLOv5. While the model achieves perfect accuracy for all surprise instances in the test set and near perfect results for disgust, happiness, and neutral, sadness class remains poor and dropping slightly compared to the YOLOv5 model. The massive regression in YOLOv7 is the Fear class, dropping from 100% to 56%.



Figure 8. YOLOv7 confusion matrix on the validation set.

Celal Bayar University Journal of Science
Volume 22, Issue 2, 2026, p 269-283
Doi: 10.18466/cbayarfbe.1621669

While Fear and Surprise share very similar features with widened eyes and raised eyebrows, the YOLOv5 model managed to separate them effectively. However, both models struggle with the Sadness classification, which justifies the conclusion that the issue might lie in the morphology of the emotion, not the model architecture.

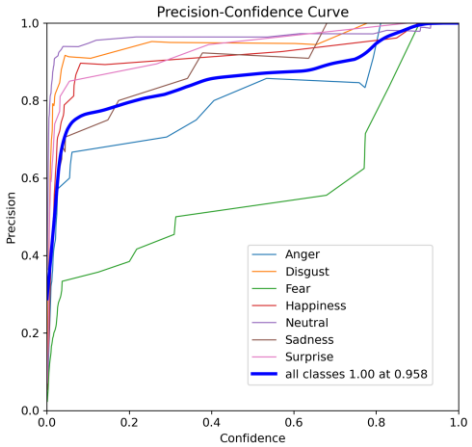


Figure 9. YOLOv5 Precision-Confidence curve on the validation set.

Figure 9 illustrates the precision of the YOLOv5 model across varying confidence thresholds. The bold blue line in the graph represents the mean precision for "all classes" at a given confidence. The model demonstrates a stable increase in precision as confidence rises. Notably, except for Anger and Fear classes, all classes consistently stay above the baseline for all confidence intervals.

Figure 10 depicts the same metric for the YOLOv7 architecture. The curves for some classes such as Disgust and Neutral rise sharply, indicating that the model can reach high precision for these expressions. However, in contrast to YOLOv5, the Sadness curve in YOLOv7 also falls below the average line, signifying that even at moderate confidence levels, YOLOv7 produces a higher rate of FPs for Sadness compared to the other classes.

While YOLOv7 optimizes more distinguished classes that are easier to detect, pushing their curves to the top left, this appears to sacrifice performance on harder-to-detect classes such as Sadness.

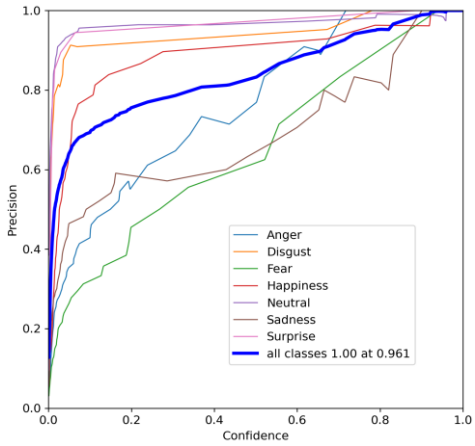


Figure 10. YOLOv7 Precision-Confidence curve on the validation set.

Figure 11 presents the precision-recall curve of the YOLOv5 model. In this context, a model that is capable of robust classification is characterized by a curve that maintains proximity to the top-right corner, thereby maximizing the AUC. Accordingly, the YOLOv5 model demonstrates a stable precision-recall curve during the validation phase, achieving a mAP@0.5 of 0.98 across all classes.

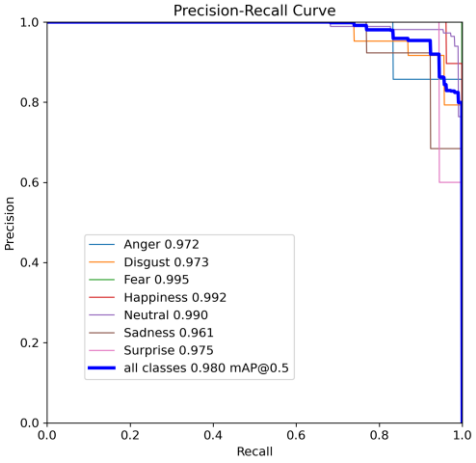


Figure 11. Precision-recall curve of YOLOv5.

Consistent with the confusion matrices analyzed previously, the Happiness class yields the highest individual performance with an AP of 0.99 while Sadness class represents the lower bound at 0.96. The tight clustering of all class curves near the 1.0 precision mark indicates that YOLOv5 maintained a balanced performance profile with minimal inter-class variance during validation.

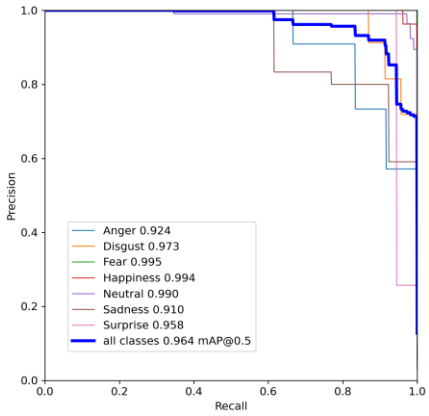
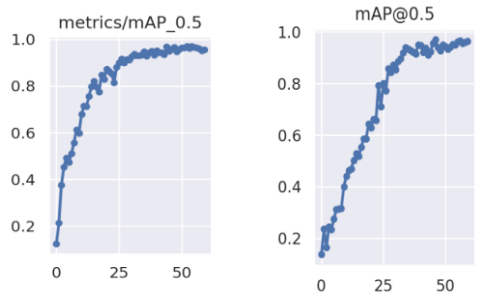


Figure 12. Precision-recall curve of YOLOv7.

In contrast, the precision-recall curves for the YOLOv7 architecture, shown in Figure 12, present a slightly different behavioral profile with an overall mAP@0.5 of 0.964. While YOLOv7 matches or slightly exceeds YOLOv5's performance for distinct expressions such as 'Happiness' (0.994) and 'Fear' (0.995), it exhibits a more significant degradation in the 'Sadness' and 'Anger' classes. Specifically, the 'Sadness' curve drops notably to 0.910, and 'Anger' to 0.924, suggesting that as the model attempts to recall a higher percentage of these subtle negative expressions, its precision deteriorates more rapidly than that of YOLOv5.

Figure 13 shows the mAP@0.5 throughout the 60-epoch training cycle for both models and serves as an indicator of models' learning stability and convergence rates. Figure 13(a) illustrates the training trajectory of the YOLOv5 model. While the model successfully trends towards a high precision value, the curve is characterized by some fluctuations, particularly in the early to mid-stages of training (epochs 0–30).



(a) (b)

Figure 13. mAP@0.5 for (a) YOLOv5 (b) YOLOv7

5.2. YOLOv5 and YOLOv7 Testing Evaluation

As previously mentioned, the machine learning applications are capable of executing inference on both static images and real-time video streams. To benchmark the baseline classification performance, the models are evaluated using a standardized test set comprising 180 static images, distributed evenly with 30 samples per expression class. Accordingly, Figure 14 shows the YOLOv5 model's performance on the separate test set.

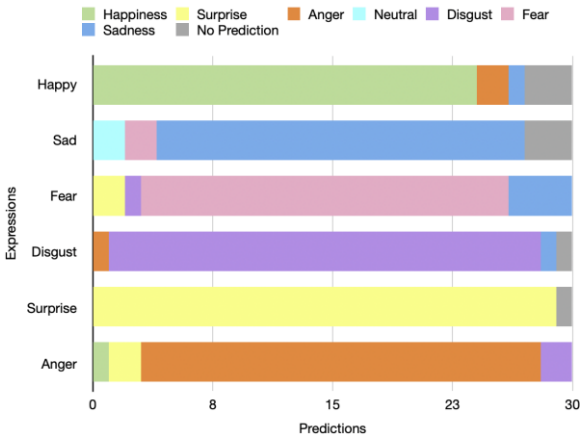


Figure 14. YOLOv5 static image test results.

YOLOv5 achieved an average accuracy of 83.85%, demonstrating strong performance on distinct expressions such as 'Surprise' (96.6%). However, it struggled with the 'Happiness' class (70%), sometimes failing to detect the expression when the facial cues are subtle.

Figure 15, on the other hand, presents the YOLOv7 model's performance on the identical test set. This model demonstrates better generalization while achieving an overall average accuracy of 91.63%. Most notably, it corrects the deficiencies observed in YOLOv5 regarding the Happiness class, increasing its accuracy from 70% to 93.3%.

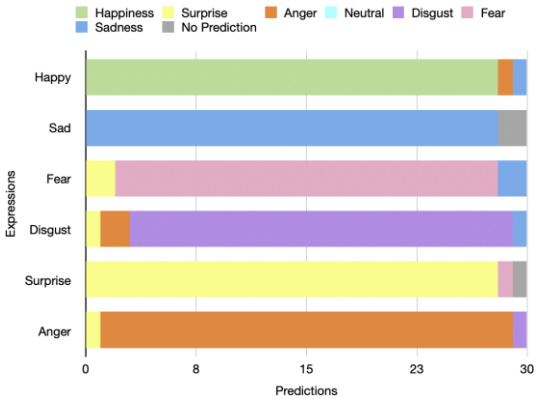


Figure 15. YOLOv7 static image test results.

Furthermore, YOLOv7 maintains high consistency across 'Sadness', 'Disgust', and 'Anger', achieving 93.3% accuracy in each class. This improvement indicates that YOLOv7's deeper architectural complexity allows it to capture and classify morphological facial features more effectively than YOLOv5, particularly in static shots where temporal context is absent.

Table I provides the breakdown of the classification accuracy for each expression, derived from the graphs in Figure 14 and 15. While YOLOv7 achieved a superior overall accuracy of 91.63% compared to YOLOv5's 83.85%, the performance distribution across individual classes offers deeper insight into their respective generalization capabilities.

Table 1. YOLOv5 and YOLOv7 static test results.

Classes	**Models**	
	YOLOv5 (%)	YOLOv7 (%)
Happy	70	93.3
Sad	76.6	93.3
Fear	76.6	86.6
Disgust	90	93.3
Surprise	96.6	90
Anger	83.3	93.3
Overall	83.85	91.63

The most significant performance disparity is observed in the 'Happy' class. YOLOv7 achieved a robust 93.3% accuracy, whereas YOLOv5 dropped drastically to 70%. This 23.3% differential indicates that the lighter YOLOv5 architecture struggles to capture the variations of a smile when the feature intensity is low or the angle is non-ideal. Similarly, for 'Sadness' and 'Anger', YOLOv7 maintained a high consistency (93.3%), significantly outperforming YOLOv5, which scored 76.6% and 83.3% respectively. This confirms that the deeper architectural layers of YOLOv7 are necessary to resolve the subtle morphological features of these complex emotional states.

However, the 'Surprise' class presents an inversion of this trend. YOLOv5 outperforms the YOLOv7 model, achieving 96.6% accuracy compared to 90%. Given that YOLOv7 achieved perfect accuracy for 'Surprise' during training, this drop during testing suggests a slight tendency toward overfitting on this specific class, whereas YOLOv5 maintained better generalization on unseen data. Nevertheless, for the majority of expressions, specifically Disgust, Fear, Sadness, Anger, and Happiness, YOLOv7 proved to be the more reliable detector.

To evaluate the models' performance in a dynamic environment, real-time testing is conducted under a controlled illumination of approximately 150 lux. The test protocol involves five distinct subjects, each instructed to perform the six facial expressions in a randomized sequence to prevent predictive bias. To ensure consistency, each expression is repeated six times per subject, yielding a total of 30 dynamic inference samples per class.

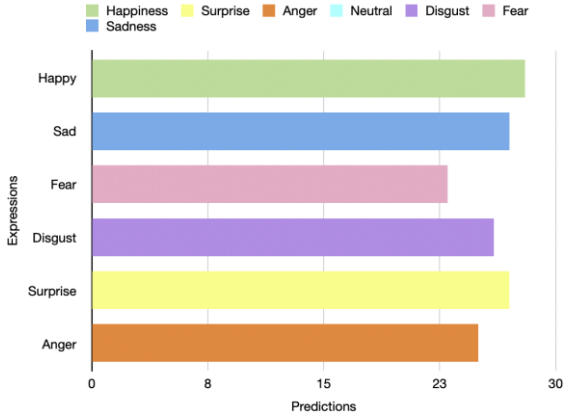


Figure 16. YOLOv5 real-time inference test results.

Figure 16 illustrates the performance of the YOLOv5 architecture under these conditions. Notably, the model demonstrates a significant performance recovery compared to the static image tests, achieving an improved average accuracy of 86.63%. The most substantial improvement is observed in the 'Happiness' class, which surged from 70% in the static test to 93.3% in the live stream. This suggests that YOLOv5 benefits heavily from the temporal consistency of video data, where continuous frame updates allow the model to latch onto distinct features, such as the formation of a smile, that might be ambiguous in a single, isolated snapshot. However, the model continued to exhibit limitations with the 'Fear' class, achieving only 76.6% accuracy, confirming that subtle upper-face deformations remain a challenge for the architecture.

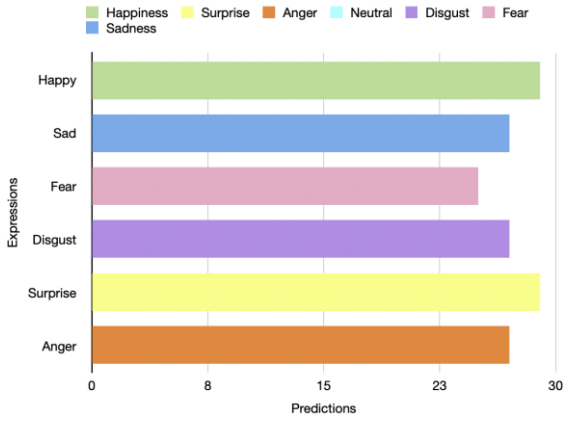


Figure 17. YOLOv7 real-time inference test results.

In comparison, Figure 17 presents the results for the YOLOv7 architecture under the same circumstances, which maintain its superior standing with an overall accuracy of 91.0%. The model exhibits robustness across the board, achieving 96.6% accuracy for both 'Happiness' and 'Surprise'. Unlike YOLOv5, which showed a wide performance variance between its best ('Happiness': 93.3%) and worst ('Fear': 76.6%) classes, YOLOv7 maintains a tighter performance spread, with even its lowest-performing class ('Fear') achieving a respectable 83.3%. This consistency indicates that YOLOv7's deeper feature extraction capabilities

provide a more stable inference platform for real-time applications, making it less reliant on perfect temporal cues than its predecessor.

Accordingly, Table 2 provides the numerical breakdown of the classification accuracy for each expression, derived from the graphs in Figure 16 and 17.

Table 2. YOLOv5 and YOLOv7 accuracy results of real-time video inference test.

Classes	Models	
	YOLOv5 (%)	YOLOv7 (%)
Happy	93.3	96.6
Sad	90	90
Fear	76.6	83.3
Disgust	86.6	90
Surprise	90	96.6
Anger	83.3	90
Overall	86.83	91

Last but not least, in order to evaluate the system' reliability in uncontrolled environments, a light-test is conducted by progressively reducing the ambient illumination from a well-lit standard (300 lux) to near total darkness (0 lux). Accordingly, Figure 18 illustrates the degradation of detection accuracy for the ML models as lighting conditions deteriorate.

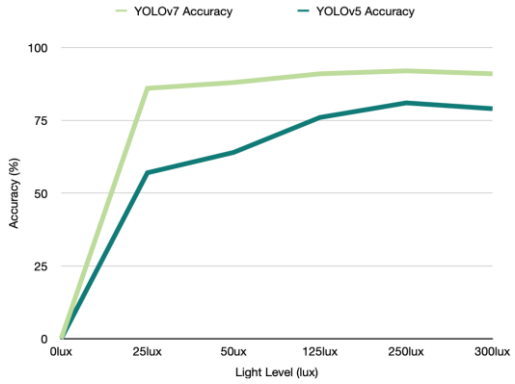


Figure 18. Model tests under varying illumination conditions.

The YOLOv7 architecture demonstrates resilience, maintaining a "satisfactory" accuracy rate even in dimmer environments. This suggests that its advanced E-ELAN architecture is capable of extracting viable feature descriptors even when contrast and texture details are diminished by poor lighting. In contrast, the YOLOv5 model exhibits a sharper decline in performance as the lux level drop. This confirms that the lighter architecture relies heavily on high-frequency details, such as clear edges and shadows, which are lost in low-light scenarios, making it less suitable for evening or outdoor use.

5.3. ARKit Testing Evaluation

The testing protocol for the ARKit-based application is the same as the live-stream conditions utilized for the ML models. The system is evaluated on the same five subjects, with each expression held for approximately 3 seconds to test the stability of the detection. Accordingly, the true positives and false positives of the test procedure is presented in Figure 19.

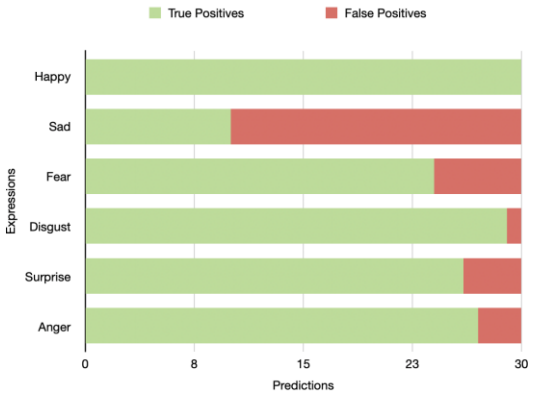


Figure 19. ARKit based solution test results.

The ARKit implementation achieves an overall classification accuracy of **81.63%**. While this falls slightly below the peak accuracy of YOLOv7 (91.0%), the error distribution reveals specific hardware-dependent behaviors. As illustrated in the True Positive/False Positive analysis in Figure 19, the system performs well on expansive expressions such as happiness and surprise. These emotions involve significant mesh deformations (mouth widening, jaw dropping) that are easily registered by the TrueDepth camera's depth sensor.

However, the system exhibits a higher rate of misclassification for contractive expressions, particularly sadness. The confusion matrix indicates that Sadness is frequently misclassified as Fear or "Neutral". This limitation is inherent to the BlendShape logic. The facial muscle movements for "Sadness" share overlapping coefficients with the "Neutral" state. Unlike the ML models, which learn texture, ARKit relies purely on geometry. If the user's "sad" expression does not cross the 0.9 coefficient threshold, the system defaults to "Neutral," leading to a false negative. Nonetheless, a distinct advantage of the ARKit approach is its independence from the visible light spectrums. Figure 19 shows the application's performance under different lighting conditions.

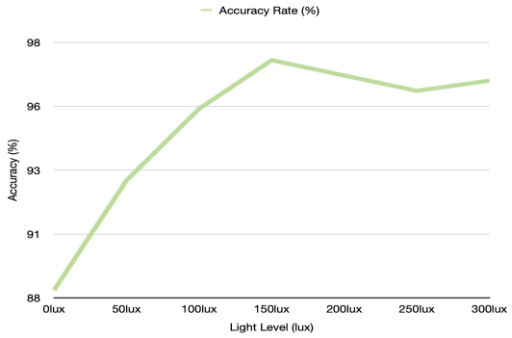


Figure 19. ARKit test under varying illumination conditions.

The system maintains consistent tracking performance as ambient illumination was reduced from 300 lux to 0 lux. Unlike the YOLO models, which suffered degradation below 50 lux, ARKit utilizes the infrared projector of the TrueDepth camera. This allows it to generate a precise 3D face mesh even in total darkness, validating it as the only viable solution for night-time or low-light scenarios where RGB-based solutions fail. While both ML models ultimately failed in total darkness due to their reliance on RGB pixel data, ARKit maintained functionality. Lastly, to determine the optimal solution for mobile deployment, three approaches are compared across inference speeds in Table 3.

Table 3. ARKit, YOLOv7 and YOLOv5 methods' inference times in milliseconds.

Classes	**Models**		
	YOLOv5	YOLOv7	ARKit
Happy	379	292	30
Sad	375	280	42
Fear	380	291	40
Disgust	376	289	39
Surprise	382	293	34
Anger	390	285	38
Overall	380.33	288.23	37.8

The ARKit approach demonstrates a significant performance advantage, operating nearly 7.6x faster than YOLOv7. This can be attributed to ARKit leveraging the dedicated Neural Engine and depth-sensing hardware optimized specifically for this task, whereas the YOLO models must perform computationally expensive convolution operations on the entire RGB frame via the hardware. While YOLOv7 is significantly more accurate than YOLOv5, it also proved to be faster, likely due to its more optimized architecture.

6. Conclusion and Discussion

This study presents a comparative evaluation of two distinct approaches for Real-Time FER on mobile devices: a geometry-based method utilizing Apple's ARKit and a machine learning-based method utilizing YOLO architectures (YOLOv5 and YOLOv7). Through various testing across static datasets, live video streams, and varying environmental conditions, the study identifies the trade-offs between computational efficiency, classification accuracy, and hardware dependency.

The experimental results demonstrate that the YOLOv7 architecture achieves the highest overall classification accuracy of 91.63%, outperforming both YOLOv5 (86.63%) and the ARKit approach (81.63%). YOLOv7 proved particularly robust in classifying emotional states such as "Happiness" and "Surprise," where it achieves near-perfect precision. However, this accuracy comes at the cost of computational latency; with an average inference time of 288ms on an iPhone 13.

In contrast, the ARKit implementation demonstrates superior operational efficiency. With an inference latency of just 37.8ms, it provides a more fluid user experience. Furthermore, the light test reveals a critical advantage: while the RGB-based ML models suffered significant performance degradation below 50 lux, the ARKit system maintains consistent tracking even in total darkness, enabled by its active infrared depth sensing.

The YOLOv7 model yields the best metrics where decision precision is crucial and real-time feedback is secondary, such as post-processing video analysis. ARKit approach, on the other hand, with its lower accuracy, its sub-40ms latency island ability to function in total darkness make it the superior choice for interactive applications where responsiveness is more critical than perfect classification.

However, it is critical to acknowledge that this comparison is not between two pure software algorithms. ARKit represents a hardware-software hybrid approach; its superior performance in low light is not due to a better algorithm, but due to the physical advantage of the TrueDepth Infrared projector. The YOLO models are inherently handicapped by being limited to 2D RGB data, forcing them to infer depth and texture from pixel intensity, whereas ARKit receives 3D geometry directly.

Therefore, this study does not conclude that one approach is universally better. Rather, it establishes that the ARKit approach is the more efficient mobile solution provided the target device supports the necessary hardware. For cross-platform applications or legacy iOS devices lacking depth sensors, YOLOv7 remains the most robust software-only alternative.

Accordingly, future research should focus on sensor fusion techniques that combine the speed of ARKit's geometry tacking with the textural understanding of

lightweight CNNs to potentially bridge the gap between high-speed tracking and high-accuracy classification.

Author's Contributions

Burcu Selçuk: Drafted and wrote the manuscript, performed the experiment and result analysis.

Tacha Serif: Supervised and reviewed the experimental progress, contributed in result interpretation and helped in manuscript preparation.

Ethics

There are no ethical issues after the publication of this manuscript.

References

[1]. Zhan, C., Li, W., Ogunbona, P. O., & Safaei, F. (2006). Facial expression recognition for multiplayer online games.

[2]. Tang, X. Y., Peng, W. Y., Liu, S. R., & Xiong, J. W. (2020, February). Classroom teaching evaluation based on facial expression recognition. In Proceedings of the 2020 9th International Conference on Educational and Information Technology (pp. 62-67).

[3]. Yolcu, G., Oztel, I., Kazan, S., Oz, C., Palaniappan, K., Lever, T. E., & Bunyak, F. (2017, November). Deep learning-based facial expression recognition for monitoring neurological disorders. In 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) (pp. 1652-1657). IEEE.

[4]. Acquisti, A., Gross, R., & Stutzman, F. D. (2014). Face recognition and privacy in the age of augmented reality. Journal of Privacy and Confidentiality, 6(2), 1.

[5]. Alkawaz, M. H., Waili, T., & Adnan, S. M. (2020). Augmented reality for patient information using face recognition and cloud computing. International Journal on Perceptive and Cognitive Computing, 6(1), 24-27.

[6]. Golnari, A., Khosravi, H., & Sanei, S. (2020, February). Deepfacear: deep face recognition and displaying personal information via augmented reality. In 2020 International Conference on Machine Vision and Image Processing (MVIP) (pp. 1-7). IEEE.

[7]. Colmenarez, A., Frey, B., & Huang, T. S. (1999, June). A probabilistic framework for embedded face and facial expression recognition. In Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149) (Vol. 1, pp. 592-597). IEEE.

[8]. Li, Y., Wang, S., Zhao, Y., & Ji, Q. (2013). Simultaneous facial feature tracking and facial expression recognition. IEEE Transactions on image processing, 22(7), 2559-2573.

[9]. Cohen, I., Sebe, N., Garg, A., Chen, L. S., & Huang, T. S. (2003). Facial expression recognition from video sequences: temporal and static modeling. Computer Vision and image understanding, 91(1-2), 160-187.

[10]. Tong, Y., Chen, J., & Ji, Q. (2008). A unified probabilistic framework for spontaneous facial action modeling and understanding. IEEE transactions on pattern analysis and machine intelligence, 32(2), 258-273.

[11]. Liu, K. C., Hsu, C. C., Wang, W. Y., & Chiang, H. H. (2019, July). Real-Time facial expression recognition based on cnn. In 2019 International Conference on System Science and Engineering (ICSSE) (pp. 120-123). IEEE.

[12]. Devries, T., Biswaranjan, K., & Taylor, G. W. (2014, May). Multi-task learning of facial landmarks and expression. In 2014 Canadian conference on computer and robot vision (pp. 98-103). IEEE.

[13]. Puthanidam, R. V., & Moh, T. S. (2018, January). A hybrid approach for facial expression recognition. In Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication (pp. 1-8).

[14]. Wang, X., Huang, J., Zhu, J., Yang, M., & Yang, F. (2018, August). Facial expression recognition with deep learning. In Proceedings of the 10th International Conference on Internet Multimedia Computing and Service (pp. 1-4).

[15]. Yang, B., Jianming, W., & Hattori, G. (2021, September). Face mask aware robust facial expression recognition during the COVID-19 pandemic. In 2021 IEEE International Conference on Image Processing (ICIP) (pp. 240-244). IEEE.

[16]. Yang, B., Wu, J., & Hattori, G. (2020, November). Facial expression recognition with the advent of face masks. In 19th International Conference on Mobile and Ubiquitous Multimedia (pp. 335-337).

[17]. Sarode, N., & Bhatia, S. (2010). Facial expression recognition. International Journal on computer science and Engineering, 2(5), 1552-1557.

[18]. Benedict, S. R., & Kumar, J. S. (2016, October). Geometric shaped facial feature extraction for face recognition. In 2016 IEEE International Conference on Advances in Computer Applications (ICACA) (pp. 275-278). IEEE.

[19]. Lemaire, P., Ben Amor, B., Ardabilian, M., Chen, L., & Daoudi, M. (2011, December). Fully automatic 3D facial expression recognition using a region-based approach. In Proceedings of the 2011 joint ACM workshop on Human gesture and behavior understanding (pp. 53-58).

[20]. Ramanathan, S., Kassim, A., Venkatesh, Y. V., & Wah, W. S. (2006, October). Human facial expression recognition using a 3D morphable model. In 2006 International conference on image processing (pp. 661-664). IEEE.

[21]. Sarvadevabhatla, R. K., Benovoy, M., Musallam, S., & Ng-Thow-Hing, V. (2011, November). Adaptive facial expression recognition using inter-modal top-down context. In Proceedings of the 13th international conference on multimodal interfaces (pp. 27-34).

[22]. Hsu, R. L., & Jain, A. K. (2001, October). Face modeling for recognition. In Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205) (Vol. 2, pp. 693-696). IEEE.

[23]. Park, B. H., & Oh, S. Y. (2015, October). Facial Expression Training System using Bilinear Shape Model. In Proceedings of the 3rd International Conference on Human-Agent Interaction (pp. 239-241).

[24]. Alshamsi, H., Meng, H., & Li, M. (2016, August). Real time facial expression recognition app development on mobile phones. In 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD) (pp. 1750-1755). IEEE.

[25]. Lucey, Patrick, et al. "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression." 2010 ieee computer society conference on computer vision and pattern recognition-workshops. IEEE, 2010.

[26]. Yoon, C., & Kim, D. (2020, October). Mobile Convolutional Neural Networks for Facial Expression Recognition. In 2020 International Conference on Information and Communication Technology Convergence (ICTC) (pp. 1315-1317). IEEE.

[27]. LI, Shan; DENG, Weihong; DU, JunPing. Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017. p. 2852-2861

[28]. OUFQIR, Zainab; EL ABDERRAHMANI, Abdellatif; SATORI, Khalid. ARKit and ARCore in serve to augmented reality. In: 2020 international conference on intelligent systems and computer vision (ISCV). IEEE, 2020. p. 1-7.

[29]. WANG, Chien-Yao; BOCHKOVSKIY, Alexey; LIAO, Hong-Yuan Mark. 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. p. 7464-7475.

[30]. Lyons, Michael, Miyuki Kamachi, and Jiro Gyoba. "The Japanese female facial expression (JAFFE) dataset." (No Title) (1998).

[31]. Khaireddin, Yousif, and Zhuofa Chen. "Facial emotion recognition: State of the art performance on FER2013." arXiv preprint arXiv:2105.03588 (2021).

[32]. Dhall, A., Goecke, R., Lucey, S., & Gedeon, T. (2011). Static facial expression analysis in the wild: An evaluation of the SFEW database. *In 2011 IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011)* (pp. 819-824). IEEE.

[33]. Ekman, Paul, and Wallace V. Friesen. "Facial action coding system." Environmental Psychology & Nonverbal Behavior(1978).