Article

Deep-Learning-Incorporated Augmented Reality Application for Engineering Lab Training

John Estrada [1], Sidike Paheding [2,*], Xiaoli Yang [1] and Quamar Niyaz [1]

[1] Department of Electrical and Computer Engineering, Purdue University Northwest, Hammond, IN 46323, USA; estradag@pnw.edu (J.E.); yangx@pnw.edu (X.Y.); qniyaz@pnw.edu (Q.N.)
[2] Department of Applied Computing, Michigan Technological University, Houghton, MI 49931, USA
* Correspondence: spahedin@mtu.edu

Abstract: Deep learning (DL) algorithms have achieved significantly high performance in object detection tasks. At the same time, augmented reality (AR) techniques are transforming the ways that we work and connect with people. With the increasing popularity of online and hybrid learning, we propose a new framework for improving students' learning experiences with electrical engineering lab equipment by incorporating the abovementioned technologies. The DL powered automatic object detection component integrated into the AR application is designed to recognize equipment such as multimeter, oscilloscope, wave generator, and power supply. A deep neural network model, namely MobileNet-SSD v2, is implemented for equipment detection using TensorFlow's object detection API. When a piece of equipment is detected, the corresponding AR-based tutorial will be displayed on the screen. The mean average precision (mAP) of the developed equipment detection model is 81.4%, while the average recall of the model is 85.3%. Furthermore, to demonstrate practical application of the proposed framework, we develop a multimeter tutorial where virtual models are superimposed on real multimeters. The tutorial includes images and web links as well to help users learn more effectively. The Unity3D game engine is used as the primary development tool for this tutorial to integrate DL and AR frameworks and create immersive scenarios. The proposed framework can be a useful foundation for AR and machine-learning-based frameworks for industrial and educational training.

Keywords: artificial intelligence; augmented reality; machine learning; object detection; computer in education; lab equipment tutorial

check for updates

Citation: Estrada, J.; Paheding, S.; Yang, X.; Niyaz, Q. Deep-Learning-Incorporated Augmented Reality Application for Engineering Lab Training. *Appl. Sci.* **2022**, *12*, 5159. https://doi.org/10.3390/app12105159

Academic Editor: Luis Javier Garcia Villalba

Received: 14 April 2022
Accepted: 17 May 2022
Published: 20 May 2022

1. Introduction

It is important for electrical engineers to understand how to use electrical equipment correctly. However, learning how to use equipment in a few cases has been a challenge for freshman electrical engineering students as many lab equipment are complex with several functionalities that are difficult to understand at the freshman level [1]. Following lab or user manuals and watching video tutorials are traditional approaches to learn how to use equipment. However, they do not guarantee that students will retain all the information. With recent technological advancements, new teaching strategies that create immersive and hands-on experiences are being researched to increase students' interest and knowledge [2].

In this paper, we describe the design and development of a smartphone app that uses deep learning (DL) and augmented reality (AR) to create a learning platform for teaching students how to use electrical lab equipment. These new technologies with their integration into tools and applications used for day-to-day tasks have made life easier not only for students, but also for people in different roles. They also benefit manufacturing industries, gaming, education, health, farming, and a variety of other fields that require process automation [3]. Artificial intelligence (AI) is used to complete complex tasks in the same way that humans do [4]. Extended reality (XR), a concept referring to virtual worlds and human–machine interactions, was developed to supplement the features that computers and mobile devices normally provide [5]. Both AI and XR have the potential to be powerful workplace tools. For example, teams at various locations could work together in a virtual environment using AI and XR technologies to create new products and prototypes seamlessly. The applications of AI and XR are crossing many fields, ranging from workflow optimization in various healthcare processes and industrial training procedures to interactive educational systems [6]. Augmented reality (AR) is one of the XR realities that is commonly used in mobile/tablet devices.

Deep learning (DL), a sub-field of machine learning (ML), embraces artificial neural networks (ANN), which are algorithms inspired by the structure and function of the human brain. ML has made significant advances in recent years because of the need for increased automation and intelligence [7]. XR refers to immersive technology that encompasses three distinct realities: AR, mixed reality (MR), and virtual reality (VR). AR superimposes three-dimensional objects on the physical world, requiring the use of mobile devices to create interactions. MR is a technology that combines the physical and digital worlds to create immersive physical experiences. Users interact with both the physical and digital worlds by using their five senses. VR is a fully digitized world in which users can completely immerse themselves in the computer-generated world by using virtual reality devices [8].

Many AR apps have recently been developed. AUREL [9] is an interactive application that aids in the understanding of specific STEM topics. It enhances the learning experience by projecting 3D models onto physical 2D textures that are part of the AR system, drawing virtual objects using the mobile display, and placing them onto a specific image tracked for the camera. The image detection for the ML system uses the camera as input data to detect specific images based on a trained dataset. Nonetheless, its application is limited to flat image recognition, allowing them to research and extend their idea for object recognition. An AR application [10] was implemented to detect a breadboard and instruct students on how to build a circuit. Their system scans a circuit diagram for circuit symbols and their connections. These components are then arranged by a neural network. The AR system provides a 3D visualization of the scanned circuit diagram which students can use as a guided tutorial to build real circuitry. Another study [11] was to create and test an augmented reality application to teach power electronics to beginners. Two AR applications for RLC circuits and Buck–Boost converters were created, and the experimental results showed that they had a positive effect on students when compared to traditional teaching methods. The results of the experiment indicated improved cognitive performance. Despite the fact that augmented reality has made its way into STEM education, there is no general non-linear framework that can guide the development of an AR-based tutorial to our knowledge. Furthermore, the presented study goes in the direction of facilitating a smooth transition from real-time object recognition using deep learning methods to interactive tutorials using AR technologies, a particular step of the process where there is potential for improvement.

In this paper, we discuss the design and implementation of an AR- and DL-based smartphone app to assist students in learning how to use electrical lab equipment such as multimeters. A similar framework can be applied to develop AR- and DL-based apps for other equipment in the future. The paper is structured as follows. Section 2 provides an overview of the DL and AR techniques suitable for this type of application. Section 3 illustrates the design and implementation of the smartphone app using different AR and DL frameworks. The experimental results are discussed in Section 4. Finally, the paper ends with a conclusion and future works in Section 5.

2. Overview of Deep Learning and Augmented Reality

This work explores the idea of using equipment recognition and an AR-based tutorial to enhance student learning experiences with electrical equipment in their engineering laboratories. Our long-term goal is to develop interactive smartphones apps for lab equipment such as multimeters, oscilloscopes, wave generators, and power supplies. Object detection using DL methods fits our goal because the app can detect specific electrical equipment in the lab with high precision in real-time using state-of-art DL algorithms. AR technology enables us to create virtual scenarios and integrate 3D models, animations, images, and videos embedded into teaching methods. In the developed app, the interactive visualization was created using the Unity3D game engine [12]. The object detection process of the app employs a deep neural network architecture trained with TensorFlow API [13].

2.1. Deep Learning

Innovative DL techniques for performing specific tasks have emerged rapidly in recent years due to significant advances in hardware development and data availability. For big data predictive analytics and multi-modal learning, DL algorithms are quite suitable, while traditional ML algorithms face several limitations. Studies in [14,15] point out that these DL methods are constructed with hierarchical layers and use complex neural networks to improve their performances iteratively. Machines equipped with DL models can perform specialized tasks such as driving a car, identifying weeds in a field, diagnosing diseases, evaluating machinery for errors, and even recognizing objects in real-time. They are used in a wide range of computer science domains including computer vision [15], natural language processing [16], and speech recognition [17]. In this work, we are using a deep neural network to detect objects. Object detection is primarily used in computer vision and has gained popularity in a variety of applications over the last decade, including autonomous vehicles, intelligent video, and surveillance systems [18].

For object detection, a type of deep neural network called a convolutional neural network (CNN) [19] has been widely used. CNN is a well-known DL algorithm that employs backpropagation in a feed-forward neural network with a collection of neurons arranged in hierarchical layers. It also exhibits typical neural network characteristics such as multiple interconnected hidden layers [20,21]. CNN for object detection is trained on large labeled datasets and neural network architectures that learn features directly from data without explicit feature engineering. In recent years, object detection methods such as the region-based convolutional neural network (RCNN) [22], you only look once (YOLO) [23], and single shot detector (SSD) [24] have been proposed. The rapid development of neural networks improves the accuracy and real-time performance of object identification tasks significantly [25]. In this paper, we compared RCNN and MobileNet-SSD v2 [26] and observed that MobileNet-SSD v2 has better performance for real-time applications in terms of speed when implemented on mobile devices. It is important to note that MobileNet-SSD v2 is a lightweight deep neural network architecture designed specifically for mobile devices with high recognition accuracy. Therefore, we employed MobileNet-SSD v2 in this work.

2.2. Augmented Reality

The relationship between the real and virtual worlds, as mentioned in Section 1, is what distinguishes the various XR technologies. In AR, users perceive virtual objects as real-world extensions, whereas MR users combine and interact in both the real and digital world, and VR users immerse themselves entirely in a virtual world [27,28]. This paper focuses on the development of AR applications that allow us to create experiences by utilizing additional digital data about our surroundings. We can receive digital information in real-time through devices such as webcams, mobile phones, and tablets. In other words, AR allows us to overlay layers of visual information in the physical world, allowing humans to interact with virtual 3D objects as well as physical objects around us [29]. This feature has revolutionized the ways humans learn and comprehend [30]. AR development necessitates three key components: a physical object that serves as a model for the virtual object's interpretation and production; intelligence devices with access to a camera that project an image of a targeted object; and software that interprets the signal sent by the camera [31].

There are diverse types of AR suitable for different applications despite the fact that they all have similar capabilities [32,33]. Figure 1 depicts the two primary types of AR: marker-based AR and marker-less AR.

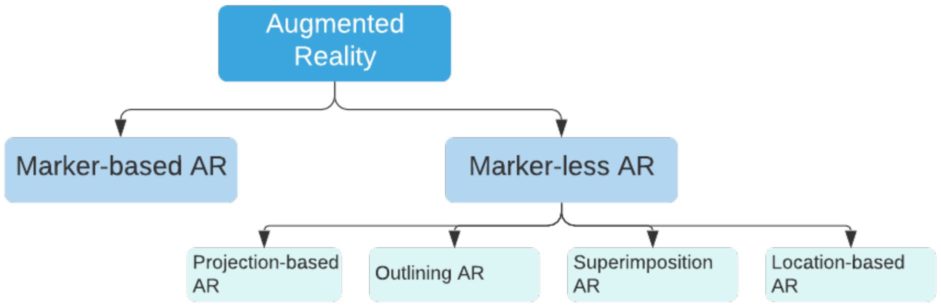


Figure 1. Types of augmented reality.

2.2.1. Marker-Based AR

Marker-based AR works when it is triggered by pre-defined markers. It allows the user to choose where to place the virtual object. Barcodes and QR codes are commonly used as images or photo symbols to be placed on flat surfaces. The program recognizes the marker when the mobile device focuses the target image. The virtual information will be projected by the AR onto the marker that will be displayed on the device. There are many levels of complexity in marker-based AR [34]. For example, a few display virtual information when the device is focused on the marker, while others save that virtual information and allow users to view it again when the device is focused on a different section. The marker-based AR technology leverages images from the actual world or QR codes to extract points, lines, corners, textures, and other properties [35]. These images are used to superimpose and create AR experiences by referencing track points in the physical world.

2.2.2. Marker-Less AR

Marker-less AR is more versatile than marker-based AR. It interacts with the real object without the need for pre-defined markers but leaves the freedom to the user. This allows the user, for example, to position a virtual object anywhere on a real object. Users can experiment with different styles and locations digitally without having to move anything in their immediate surroundings [36]. Marker-less AR collects data from the device hardware such as a camera, a GPS, a digital compass, and an accelerometer for the AR program to function. Marker-less AR applications rely on computer vision algorithms to distinguish objects, and they can function in the real world without specific markers [37,38]. There are four types of marker-less AR discussed as follows:

(a) *Location-based AR*: In this type of AR, simultaneous localization and mapping (SLAM) technology is used to track the user's location as the map is generated and updated on the user's mobile device [39]. To display AR content in the physical environment, the user must detect a surface with a mobile device [40,41]. As an example, the world-famous AR-based game app, Pokemon Go, uses SLAM technology that allows its users to battle, navigate, and search for 3D interactive objects based on their geographical locations [42].

(b) *Superimposition-based AR*: Superimposition-based AR applications can provide an additional view along with the original view of the object. Object recognition is required to determine the type of object to partially or completely replace an object in the user's environment with a digital image [43,44]. Using HoloLens glasses, surgeons can superimpose images previously gathered through scanners or X-rays on the patient's body during the operation. They can anticipate potential problems using this approach.

(c) *Projection-based AR*: Projection-based AR (also known as projection mapping and augmented spatial reality) is a technique that does not require the use of head-mounted or hand-held devices. This method allows augmented information to be viewed immediately from a natural perspective. Using projection mapping, projection-based

AR turns an uneven surface into a projection screen. This method allows for the creation of optical illusions [45].

(d) *Outlining-based AR*: This type of AR employs image recognition to create contours or forms and highlight components of the real world using special cameras. It is used by human eyes to designate specific items with lines to make situations easier. Vuforia's Model Target is an example of outlining-based AR. Vuforia is a platform that enables developers to quickly incorporate AR technology into their applications. Model Targets allow apps to recognize and track real-world objects based on their shape [46].

In our project, we built a superimposition-based AR app. We built user interfaces on top of lab equipment, allowing step-by-step instructions to be incorporated into the application for users to understand and learn how to use specific equipment. Using AR technology, immersive experiences are created in a variety of ways. It does, however, have some limitations such as the inability to recognize multiple objects at once. On the other hand, DL models show high performances in recognizing multiple objects at the same time. Integrating AR apps with DL models will help trigger specific AR scenarios based on objects being aimed at with a camera and allow an AR scenario to perform a single tracking without decreasing mobile device performance.

3. Design and Implementation of the AR App

This section describes the design and development of the AR app that integrates two independent frameworks for object detection and augmented reality as shown in Figure 2. Unity 3D combines the output of these systems by inferring the object detection model with OpenCV and using an AR dataset target with a Vuforia Engine. Furthermore, Unity 3D enables the development of interactive user interfaces. Users can first use their mobile device to infer the object detection model to detect the lab equipment. The inference will classify and localize lab equipment that has been targeted with the mobile camera. When an object is detected, a user interface (UI) button appears, indicating that an AR-guided tutorial is available for the object. Then, an AR scenario will be loaded, allowing students to use their mobile camera to aim at a specific target. Following that, a 3D object will superimpose on top of the physical object, activating UI panels with instructions on how to use the equipment.

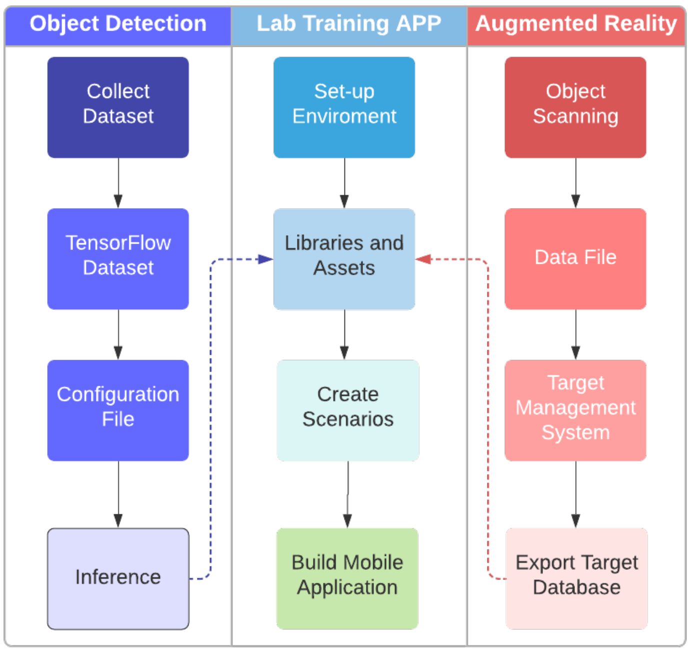


Figure 2. Design framework of AR-based smartphone app for lab equipment training.

The app development process consists of integrating a number of different independent systems with their frameworks. In the interactive tutorial development framework, Unity3D was used as the primary development software for generating specific UI instructions and creating immersive interactions between the mobile app and the user. The development framework was integrated with a MobileNet-SSD DL model and a marker-less superimposition AR that activates immersive modules containing 2D/3D objects. The detailed framework integration is discussed below.

3.1. Object Detection Framework

MobileNet-SSDv2 [26] architecture was used to build a deep neural network model to detect electrical lab equipment. The architecture comprised MobileNet-v2 as the backbone network,an SSD detector, and feature pyramid network (FPN). MobileNet, as the name implies, is intended for embedded applications on mobile devices to improve accuracy while effectively considering constrained resources. A loss function method was calculated using the predicted and labeled values of the classes and offsets to evaluate how the algorithm models the data. The confidence loss ($L_{confidence}$) occurs when attempting to predict a class, which is softmax loss over multiple classes confidences. Localization loss ($L_{localization}$) is defined as a mismatch between the ground truth and the intended boundary boxes [24,47], where α is the weight coefficient, expressed as:

$$L_{loss} = L_{confidence} + \alpha \times L_{localization} \tag{1}$$

MobileNet is a low-latency and low-power model that can be tailored to meet the resource constraints of various use cases. For multi-scale object detection, MobileNetv2 provides a number of feature maps with different dimensions for the backbone detection network to the SSD convolutional layer that uses small convolutional filters to predict scores and class offsets for a fixed set of the standard bounding boxes. MobileNet-SSDv2 extracts features from images, which are then processed through SSD predictor layers that reduce image size to recognize objects at various scales [26,48] as shown in Figure 3. Mobilenet-SSDv2 detector improves the SSD detector by combining MobileNetv2 and FPN while maintaining memory efficiency.

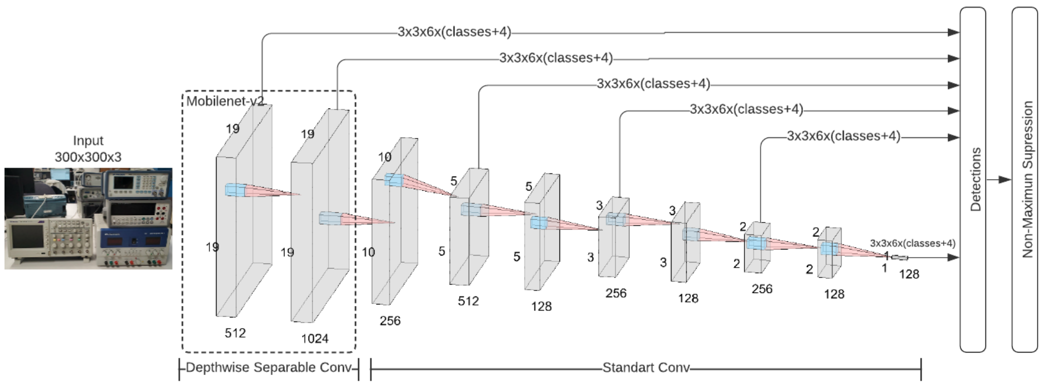


Figure 3. MobileNet SSD deep neural network architecture.

3.2. TensorFlow Object Detection API

TensorFlow (TF) API, developed by Google Brain, is a framework for creating a DL network [13]. It is a powerful tool that can be used to create a robust object detection framework with a set of standard functions, eliminating the need for users to write code from scratch. It also provides a list of pre-trained models, which are useful not only for inference but also for building models with new data. Model Zoo is a collection of models that have been previously trained using the common objects in context (COCO) dataset [49]. A workflow for training a DL model using the TF API is shown in Figure 4 and can be described through the following steps:

(a) *Image Dataset*: The model was given input of 643 images collected from various perspective views and in different lighting settings. Each image is of 4032 × 3024 pixels in size. It is necessary to annotate these images before using them to train the model. A software, `LabelImg` [50], is used in the annotation process [51] that allows users to draw a rectangle in a specific area of the image. During training, the annotation will help the model precisely locate the object in the image. The outlining will generate and save coordinate points in an XML file.

(b) *TensorFlow Dataset*: To make the computation of the DL framework efficient, TF records use a binary file format. Furthermore, TF records enable the dataset to be stored as a sequence of binary strings that improves the model's performance while using less disk space. We converted the XML files generated by LabelImg into TF binary records using a Python script. The last step in configuring the TF dataset is to create a `.pbtxt` file containing all of the categorical label classes that will be stored in a TF record file.

(c) *Configuration File*: Multiple pre-trained models based on the common objects in context (COCO) dataset are available in TF. These models can be used to set up DL models prior to training on a new dataset. Table 1 lists several popular architectures with pre-trained models. For instance, *ssd_MobileNet_v1_coco* is the SSD with a MobileNet v1 configuration, *ssd_inception_v2_coco* represents an SSD with an Inception v2 configuration, and *faster_rcnn_resnet101_coco* stands for Faster R-CNN with a Resnet-101 (v1) configuration. All these configurations have been derived for the COCO dataset. From Table 1, it can be observed that *ssd_MobileNet_v1_coco* reaches the fastest inference speed of 30 ms but with the lowest mean average precision (mAP). In contrast, *faster_rcnn_resnet101_coco* has the slowest inference speed but the highest mAP of 32. We tested both MobileNet SSD v2 and faster RCNN [52] and concluded that MobileNet SSD v2 performs faster inference in mobile devices than the faster-RCNN model in our study. Using a pre-trained model saves time and computing resources. A configuration file, in addition to the pre-trained model, is also required. It must match the same architecture of the pre-trained model. It is recommended to fine-tune the model to maximize the prediction outcome. The process of fine-tuning is divided into two steps: restoring weights and updating weights. After we completed the requirements, we ran the python code provided for TF API to start the training job. Following training, the API will generate a file serving as a training checkpoint in a specific format named `.ckpt`. This file is a binary file containing all of the weights, biases, and other variables' values.

(d) *Inference*: After training the model, the last step is to put it into production and feed the model with live data to calculate the predicted output. Before testing, we can evaluate the model's accuracy using mAP. In Section 4, the evaluation result is described in detail. We also need a lightweight version of the model to perform inference, so we choose an OpenCV library.

Table 1. Comparison of pre-trained models in Model Zoo collection based on COCO dataset [13]. The *ssd_MobileNet_v1_coco* and *ssd_MobileNet_v2_coco* are the SSD models with MobileNet v1 and v2 configurations, respectively; *ssd_inception_v2_coco* represents SSD with Inception v2 configuration, and *faster_rcnn_resnet101_coco* stands for Faster R-CNN with Resnet-101 (v1) configuration. All these configurations are used for the COCO dataset.

Model Name	Speed (ms)	COCO (mAP)	Output
ssd_mobilenet_v1_coco	30	21	Boxes
ssd_mobilenet_v2_coco	31	22	Boxes
ssd_inception_v2_coco	42	24	Boxes
faster_rcnn_resnet101_coco	106	32	Boxes

Note: Speed (ms) relates to the network's inference speed, or how long it takes to produce an output based on the input. The mAP calculates a score by comparing the ground-truth bounding box to the detected box. The higher the score, the better the model's detection accuracy is.

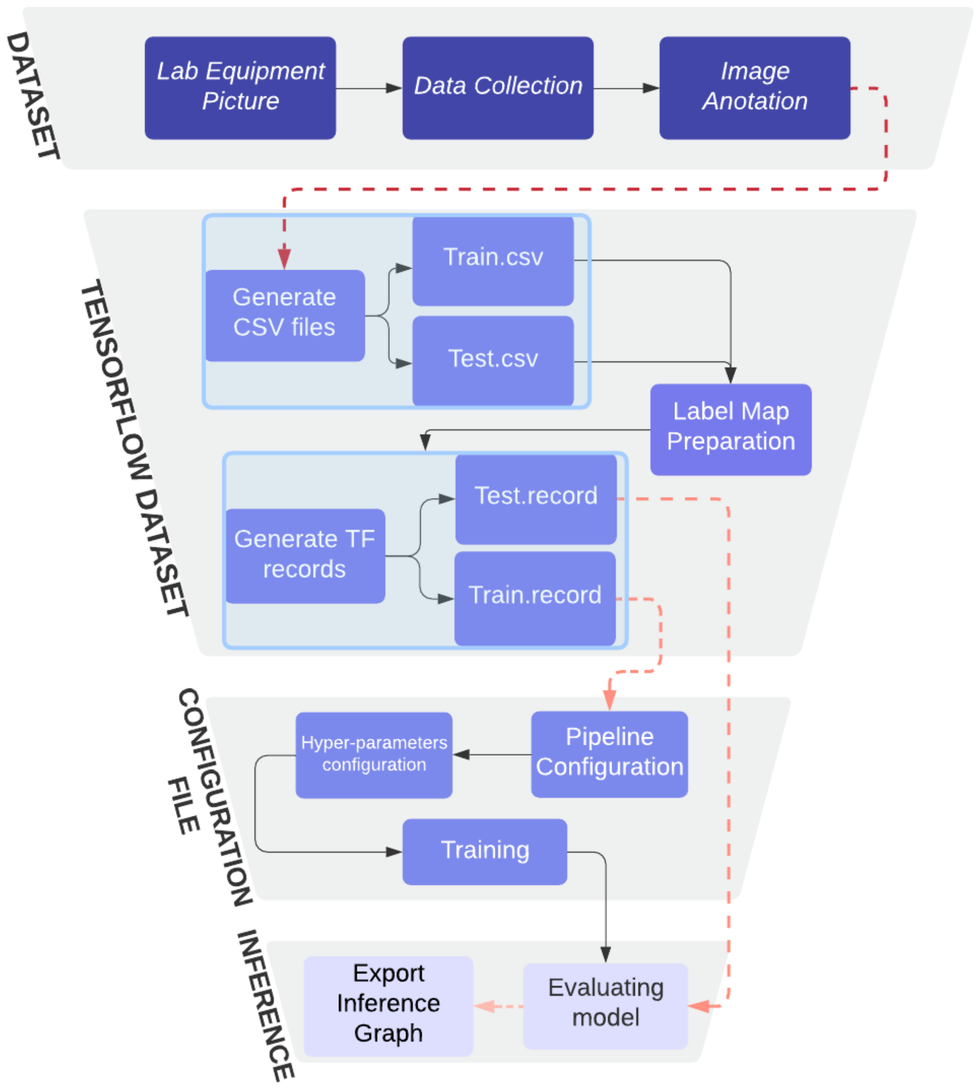


Figure 4. Object detection workflow using TensorFlow API.

In addition, there is a frozen trained model, a ready-to-use inference model that can generate an output based on the live data input, and the frozen process file is stored in Protobuf (`.pb`) file. The Protobuf model contains graph definition and trained parameters in a binary format. The text graph representation of the frozen process file is in a human-readable format required by the OpenCV library and is kept in a `.pbtxt` format. After creating the corresponding file, it is time to examine and test the trained model. We use a function called `VideoCapture` from OpenCV to test the model, which loads the input video using the PC webcam and then predicts the relevant labels and object location with an enclosed rectangle indicating its pixel location within the input image. Finally, with the Protobuf and the configuration file, we can now use the Unity3D game engine and OpenCV to create our application by triggering AR scenarios based on the detection of electrical lab equipment performed by the DL model during its inference.

3.3. Augmented Reality Framework

Vuforia is a framework that enables the creation, recognition, and tracking of virtual and physical objects in the real world using an electronic device. To test the prototype of our tutorial that integrates both object recognition and interactive augmented reality, we developed a tutorial on how to use a multimeter in the lab. A scene (a live video) captured by the camera will be saved to a mobile device. The Vuforia SDK creates a frame (a single image within a series of photos) of the captured scene. It improves the quality of the image captured by the camera so that an AR tracker component can correctly treat it. It uses the latter to analyze the image and search the database for matches, which may include one or more targets. Finally, the program renders virtual material such as photographs, videos, models, and animations on the device screen, creating a hybrid image of what we perceive as holographs. The process of generating AR targets is depicted in Figure 5 and can be described in the following steps:

(a) *Object Scanning*: It is the primary tool for generating an object data file, which is required for generating an object target in the target manager on the Vuforia webpage. This app is available on the Vuforia developer website, which users can access after creating a developer account. The ObjectScanner [53] app is used, and the scanning environment is configured. The Vuforia developer portal provides a printable target image that defines the target position and orientation relative to the local coordinate space to scan the object and collect data points. It also distinguishes and removes undesirable areas. This printable target image is used in conjunction with an Android application, which is available for free download from the Vuforia official website. During scanning, the printable target image must be placed under the object to be scanned. Using the Vuforia scanning mobile application, the user can start collecting data points from the object. To achieve the best scanning quality, it is recommended to work in a noise-free environment with moderately bright and diffuse lighting. It is also recommended to avoid objects with reflective surfaces. In this work, a multimeter met all of the requirements, and a successful scanning was achieved.

(b) *Data File*: Following the scanning, an object data file is created. The mobile app will also show how many scanning points the object has. The completed scanning area is evidenced by a square grid that changes color from gray to green. The object data file contains all the object's information. There is a test scenario to determine whether the scanned object has sufficient data for augmentation. In this scenario, a green rectangular prism will be drawn in one of the object corners relative to the target image coordinate space.

(c) *Target Management System*: Vuforia has a framework that allows developers to choose from various target types, such as picture targets, stimuli targets, cylinder targets, object targets, and VuMarks. The system will process and manage the data for visual examination. A developer license is required to fully utilize the Vuforia manager web-based tool, which includes access to a target manager panel where a database can be uploaded, and targets can be added to the management system. The 3D object option must be selected when selecting the target type, and the object data file must be uploaded.

(d) *Export Target Dataset* Following the web-tool processing the information, the database can be downloaded by choosing the desired platform. The platform can be converted into a package that can be used in the primary development process as well as to create AR experiences in Unity.

3.4. Lab Training Application Framework

With the help of AR and DL, the equipment learning application focuses on teaching and improving the student's learning experience on how to properly use electrical equipment. Unity3D will provide libraries that allow these technologies to be combined on top of assets, animations, and 3D models to create training scenarios that will engage students in learning through experience. The development procedure is shown in Figure 6 and can be described in the following steps:

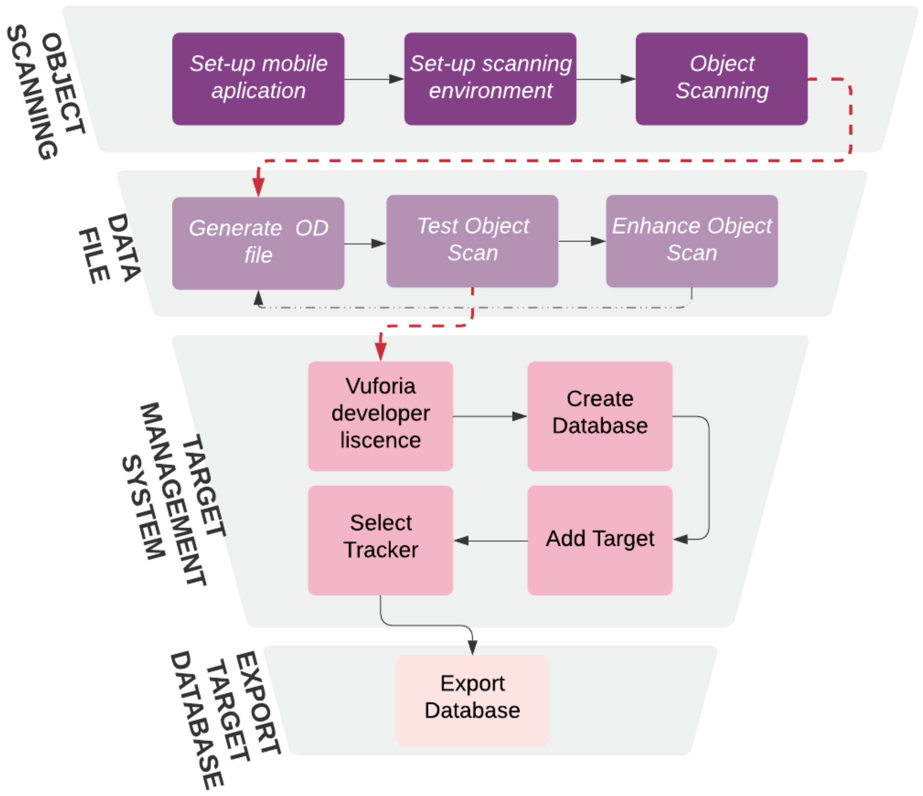


Figure 5. Vuforia object tracking framework.

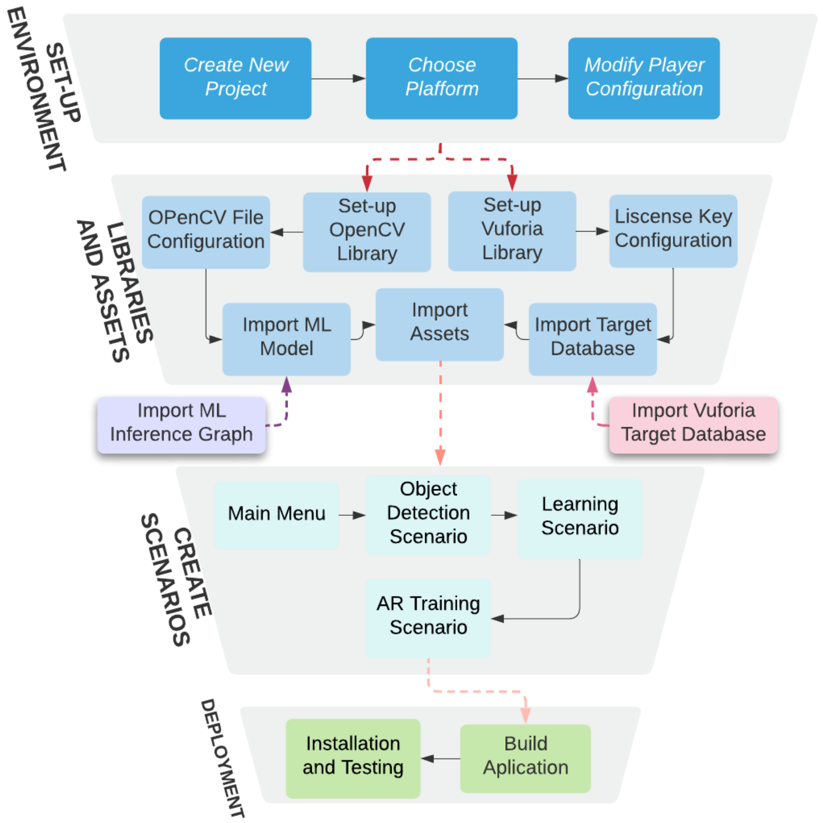


Figure 6. Lab training application framework.

(a) *Setup Environment*: The setup starts with the creation of a new project using a Unity hub. After creating and opening the project, it is essential to switch to a different build platform because Unity allows us to create once and deploy anywhere. In other words, we can select a platform from the list of available platforms in Unity, such as Windows, WebGL, Android, iOS, or any gaming console. We chose Android as the deployed platform for this project. The platform can be changed in the build settings windows, which can be accessed via the file bar. Additionally, the rendering, scripting, and project configuration must be modified.

(b) *Libraries and Assets:*

  (1) OpenCV Library: OpenCV For Unity [54] is a program that uses AI algorithms to analyze and interpret images on computers or mobile devices. This Unity asset store product allows users to test AI pre-trained models that can be used to run algorithms and executable applications on mobile devices. The model employs a script that requires a binary file of a DL model with trained weights (weights of deep neural networks are not modified in this stage), and a file model network configuration. This script is granted access to the device resource, specifically the camera, so that the script can pass input to the model and start object detection inference, which will generate bounding boxes and labels around the object detected.

  (2) Vuforia Engine: This library allows Unity to create AR experiences for mobile devices. It is a collection of scripts and pre-made components for developing Vuforia apps in Unity. It includes API libraries written in the C# language that expose the Vuforia APIs. This library supports all traceable functions as well as high-level access to device hardware such as the device camera.

  (3) Assets: They are graphical representations of any items that could be used in the project. It is made up of user interface sprites, 3D models, images, materials, and sounds, all with their own design and functionality. Photoshop is used to create art sprites, such as images for a virtual manual and blender. A 3D modeler software is used to create 3D models.

(c) *Scenarios creation*

  (1) Main menu: The application includes a menu scenario, as shown in Figure 7, that will allow the user to select various modes based on their preferences. It includes a tutorial that teaches students how to use the application. There is a training mode to help students learn more about lab equipment or electrical components.

  (2) Object detection: In this case, the DL model is used in conjunction with the OpenCV library in Unity. The application has access to the device's camera from which it will infer the object detection model provided by the object detection framework. Furthermore, depending on the object that is being targeted, the application automatically generates bounding boxes around the desired object with its respective label and confidence. When the user points to the desired equipment, a bottom panel will appear with the option to load the AR experience or continue looking for other lab equipment. The OpenCV library allows us to specify the desired confidence value threshold during the model inference. During model inference, we can specify the desired confidence value threshold using the OpenCV library. The model draws a green rectangle around the detected equipment. The detection threshold confidence value is set to 90%, which means that the confidence must be greater than or equal to 90% to indicate a detection with a rectangular bounding box. This percentage was chosen because the lab equipment is quite different. The score of 90% would ensure that the lab equipment detected had a high confidence level.

  (3) Learning scenarios: A 3D visual aid is provided in this scenario to understand the essential functions of the equipment selected during the detection scenario. Figure 8 shows how users will be able to access an interactive 3D model repre-

sentation of the equipment or view the equipment from various perspectives. In other words, it is a virtual manual introductory guide.

(4) AR training scenario: When the application detects an object that has previously been configured in Unity, a 3D model will be superimposed on top of the physical object in the mobile app. It will also include a UI for the user to interact with, allowing them to understand and explore the physical object, while the mobile application provides additional information in the form of holograms, as shown in Figure 9.

(d) *Deployment*: The final step of the framework is to build the project for the desired end platform, which can be Android or iPhone. The scenarios in Unity must be selected and linked together. Unity will launch and generate a platform associated file that can be directly installed on mobile devices.

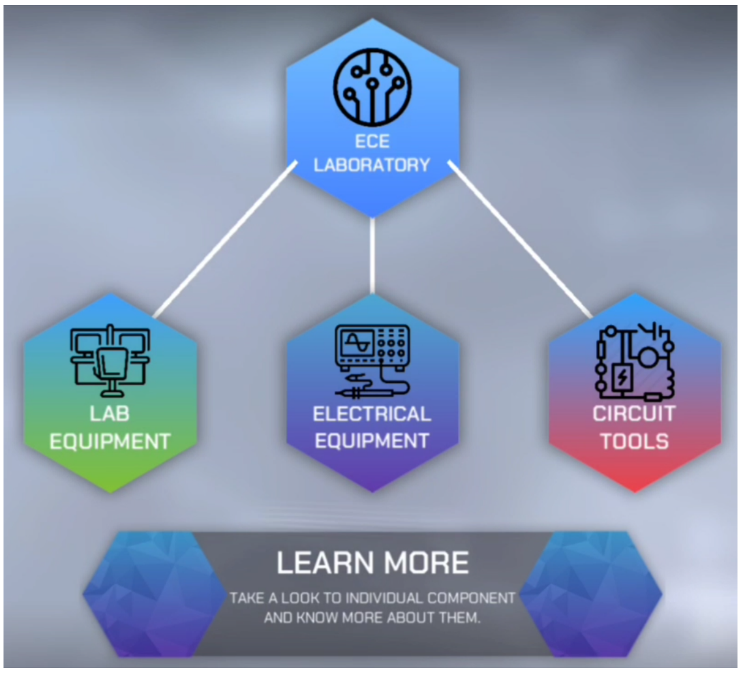
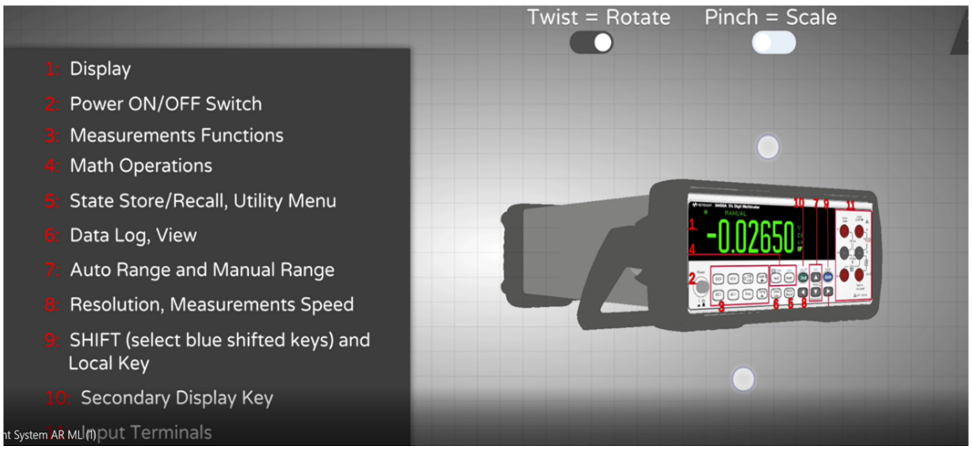


Figure 7. Main menu interface.



1: Display
2: Power ON/OFF Switch
3: Measurements Functions
4: Math Operations
5: State Store/Recall, Utility Menu
6: Data Log, View
7: Auto Range and Manual Range
8: Resolution, Measurements Speed
9: SHIFT (select blue shifted keys) and Local Key
10: Secondary Display Key
11: Input Terminals

Figure 8. Learning interactive scenario.

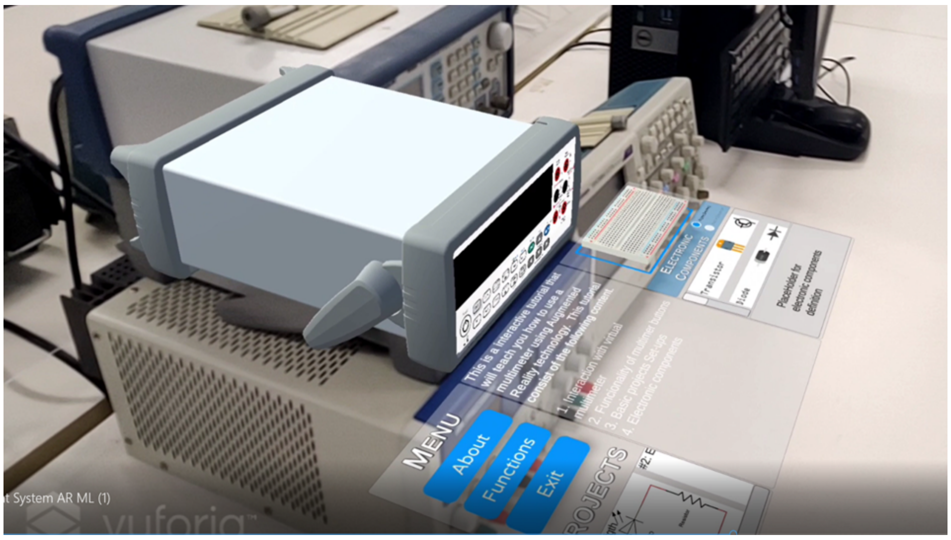


Figure 9. AR training scenario: object tracking and superimposition of a 3D object.

4. Experimental Results

In this section, performance for DL and AR frameworks are discussed.

4.1. Object Detection

The dataset used in this study is a collection of 643 images with annotations. The dataset is divided into four classes: multimeter, oscilloscope, power supply, and wave generator. The collected samples were randomly split into training and test sets with the ratio of 70% and 30%, respectively. We employ commonly used evaluation metrics such as precision, recall, and mAP to evaluate the model performance in the application.

4.2. Evaluation Metrics

- Precision: The percentage of positive detections that were correct is referred to as precision. If a model produces no false positives, it has a precision of 1.0. Equation (2) describes precision as the True Positive divided by the sum of the True Positive (TP) and False Positive (FP). TP is defined as a correct prediction of the positive class, whereas FP is an incorrect prediction of negative class as the positive class.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

- Recall: The percentage of true positives that were correctly identified by the model. A model with a recall of 1.0 produces zero false negatives. Recall can be computed as a ratio of True Positives predictions and the sum of TP and False Negatives (FN), as shown in Equation (3). FN is defined as an incorrect prediction of the positive class as the negative class.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

- Intersection over Union (IoU): It is also known as the Jaccard index used for measuring the similarity and diversity of sample sets. In an object detection task, it describes the similarity between the predicted bounding box and the ground truth bounding box. Equation (4) expresses IoU in terms of area of the prediction and ground truth bounding boxes.

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} \tag{4}$$

It is important to define a threshold to define what is the correctness of the prediction for IoU.

$$\begin{aligned} \text{T} &\leftarrow \text{Threshold} \\ \text{IoU} \geq \text{T} &\rightarrow \text{Correct} \\ \text{IoU} < \text{T} &\rightarrow \text{Incorrect} \end{aligned} \tag{5}$$

- mean Average Precision (mAP): It takes under consideration both precision and recall. It is also the area beneath the precision–recall curve. The mAP can be computed by

$$\text{mAP} = \frac{\sum_{k=1}^{n} AP_k}{n} \tag{6}$$

where AP_k is the average precision of class k, and n is the number of classes.

Figure 10 shows our model during inference when the new dataset was fed to the model. It demonstrates the correct detection of four types of lab equipment in a single shot when the confidence threshold value (i.e., the threshold related to the confidence score to determine whether the detection is an object of interest or not. Confidence scores of the predicted bounding boxes above the threshold value are considered as positive boxes, or vice versa) is greater or equal to 90%. The experimental results shown in Table 2 support that our model has a high mAP. In practice, the DL model can recognize all of the electrical lab equipment that has been pre-selected.

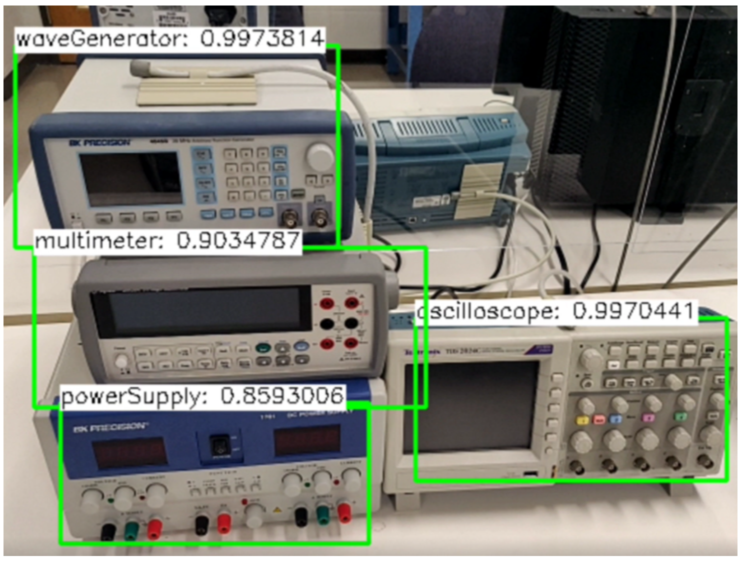


Figure 10. Example of automatic equipment detection by the DL model. The number above each green bounding box indicates confidence score of the model, which is the probability that a bounding box contains an object of interest, for the detection.

Table 2. Mean average precision (mAP) with different IoU scores of the trained MobileNet-SSD V2 model.

Description	IoU	mAP
Average Precision	0.50:0.95	0.814
	0.50	0.976
	0.75	0.954

Table 2 shows the average precision and average recall for a given IoU score and mAP. The IoU is a range between 0.50 and 0.95. Using 193 testing images, the average precision of our proposed model achieved a mAP of 81.4% and an average recall of 85.3%. Some failure cases were due to the low ambient lighting and a lack of training datasets with varying lighting conditions.

The DL model deployed on a mobile device uses CPU resources of the device to infer and predict objects. We ran the DL model on two different mobile devices to evaluate performances of devices for real-time prediction. We used the frame per second (FPS) unit, which measures the number of images that the mobile device screen displays every second.

According to Table 3, Samsung has a performance of 8.5 FPS, and One plus has a performance of 5.5 FPS, indicating that the device hardware resources are required to accelerate the inference performance.

Table 3. Mobile device hardware specification and FPS during inference.

Mobile Device	CPU	RAM	FPS
Samsung S21 Ultra	Samsung Exynos 2100	12 GB	8.5
One Plus 6T	Octa-core Kryo 385	6 GB	5.5

4.3. Augmented Reality

Detecting a multimeter in real-time is a good way to test the accuracy and precision of AR-based object detection. Figure 11 shows two mobile devices with three different luminous intensities of 25 lux, 150 lux, and 350 lux. In addition, we included various distances between the mobile camera and the multimeter in our evaluation to understand how good our scanning process was when collecting data points from the multimeter. This evaluation enables us to determine the optimal room lighting configuration for good AR-based object detection.

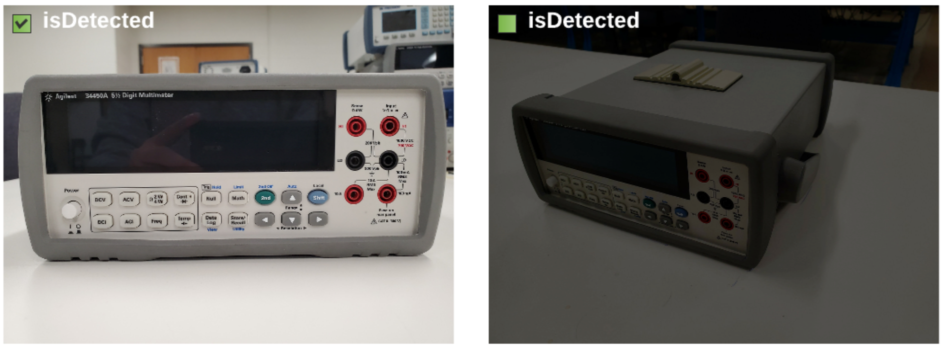


Figure 11. Image reference of Samsung s21 camera with a light intensity of 350 lux (**Left**) and 25 lux (**Right**).

We included a toggle button in the test AR scenario during the evaluation to indicate whether the application keeps detecting the multimeter. Table 4 shows the results of the AR-based detection experiments. Due to the camera's lack of focus, we discovered that our camera was not tracking the multimeter during our preliminary results. We included a script in our Unity engine project that allowed us to focus on the mobile camera. The evaluation table includes focus parameters that will help us decide whether to include this feature in the AR experience. We chose 50 cm and 100 cm for our evaluation because these are the typical distances between the lab equipment and the students. The final column contains the result in True/False format, indicating whether the multimeter was detected. We concluded that Vuforia can detect objects even in low light conditions. However, the distance will have an impact on the detection results. According to our table evaluation, the focus parameter increases the likelihood of detecting a multimeter in different light intensities, but it also depends on the camera resolution.

Table 4. AR-based object detection experiments using different devices, luminous intensities, distances between mobile camera and multimeter, and camera's focus.

Device	Luminous Intensities	Distance (cm)	Focus	Detection Status
1	25	50	No	Yes
1	25	100	No	No
1	25	50	Yes	Yes
1	25	100	Yes	Yes
1	150	50	No	Yes
1	150	100	No	No
1	150	50	Yes	Yes
1	150	100	Yes	Yes
1	350	50	No	Yes
1	350	100	No	Yes
1	350	50	Yes	Yes
1	350	100	Yes	Yes
2	25	50	No	Yes
2	25	100	No	No
2	25	50	Yes	Yes
2	25	100	Yes	No
2	150	50	No	Yes
2	150	100	No	No
2	150	50	Yes	Yes
2	150	100	Yes	No
2	350	50	No	Yes
2	350	100	No	No
2	350	50	Yes	Yes
2	350	100	Yes	Yes

5. Conclusions and Future Work

In this study, we developed an interactive multimeter tutorial using deep learning and augmented reality. We integrated a deep learning model, namely MobileNet-SSD v2, and an AR target database into a game engine to detect objects automatically. Unity3D was used to create the augmented tutorial, which includes a mobile game infrastructure. The tutorial functions as a virtual manual for the equipment, which provides an immersive experience by projecting holograms on objects recognized by the app via a mobile camera. In the future, we will create tutorials for additional lab equipment. One application will be the addition of a 3D interactive breadboard in the app to help students understand electrical circuits. Another potential enhancement of the proposed AR- and AI-based education tool would be to support remote learning, in which students can learn lab equipment through the AR streaming on their mobile devices or personal computers.

Author Contributions: J.E.: writing—original draft, methodology, software, visualization, data curation; S.P.: conceptualization, methodology, writing—reviewing and editing, supervision, funding acquisition; X.Y.: conceptualization, methodology, writing—reviewing and editing, supervision, funding acquisition; Q.N.: writing—reviewing and editing, conceptualization, funding acquisition. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Science Foundation #2129092 and #2129093, 2021.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data may be available upon request due to privacy.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mejías Borrero, A.; Andújar Márquez, J. A pilot study of the effectiveness of augmented reality to enhance the use of remote labs in electrical engineering education. *J. Sci. Educ. Technol.* **2012**, *21*, 540–557. [CrossRef]
2. Singh, G.; Mantri, A.; Sharma, O.; Dutta, R.; Kaur, R. Evaluating the impact of the augmented reality learning environment on electronics laboratory skills of engineering students. *Comput. Appl. Eng. Educ.* **2019**, *27*, 1361–1375. [CrossRef]

3. Ray, S. A quick review of machine learning algorithms. In Proceedings of the 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 14–16 February 2019; pp. 35–39.
4. Xue, M.; Zhu, C. A study and application on machine learning of artificial intelligence. In Proceedings of the 2009 International Joint Conference on Artificial Intelligence, Hainan, China, 25–26 April 2009; pp. 272–274.
5. Gong, L.; Fast-Berglund, Å.; Johansson, B. A framework for extended reality system development in manufacturing. *IEEE Access* **2021**, *9*, 24796–24813. [CrossRef]
6. Nisiotis, L.; Alboul, L. Work-In-Progress-An Intelligent Immersive Learning System Using AI, XR and Robots. In Proceedings of the 2021 7th International Conference of the Immersive Learning Research Network (iLRN), 17 May–10 June 2021; pp. 1–3.
7. Khomh, F.; Adams, B.; Cheng, J.; Fokaefs, M.; Antoniol, G. Software engineering for machine-learning applications: The road ahead. *IEEE Softw.* **2018**, *35*, 81–84. [CrossRef]
8. Hu, M.; Weng, D.; Chen, F.; Wang, Y. Object Detecting Augmented Reality System. In Proceedings of the 2020 IEEE 20th International Conference on Communication Technology (ICCT), Nanning, China, 28–31 October 2020; pp. 1432–1438.
9. Ang, I.J.X.; Lim, K.H. Enhancing STEM Education Using Augmented Reality and Machine Learning. In Proceedings of the 2019 7th International Conference on Smart Computing & Communications (ICSCC), Miri, Malaysia, 28–30 June 2019; pp. 1–5.
10. Thiwanka, N.; Chamodika, U.; Priyankara, L.; Sumathipala, S.; Weerasuriya, G. Augmented Reality Based Breadboard Circuit Building Guide Application. In Proceedings of the 2018 3rd International Conference on Information Technology Research (ICITR), Moratuwa, Sri Lanka, 5–7 December 2018; pp. 1–6.
11. Sandoval Pérez, S.; Gonzalez Lopez, J.M.; Villa Barba, M.A.; Jimenez Betancourt, R.O.; Molinar Solís, J.E.; Rosas Ornelas, J.L.; Riberth García, G.I.; Rodriguez Haro, F. On the Use of Augmented Reality to Reinforce the Learning of Power Electronics for Beginners. *Electronics* **2022**, *11*, 302. [CrossRef]
12. Technologies, U. Unity Real-Time Development Platform. 2005. Available online: https://unity.com/ (accessed on 8 April 2022).
13. Yu, H.; Chen, C.; Du, X.; Li, Y.; Rashwan, A.; Hou, L.; Jin, P.; Yang, F.; Liu, F.; Kim, J.; et al. TensorFlow 1 Detection Model Zoo. 2020. Available online: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf1_detection_zoo.md/ (accessed on 28 April 2022).
14. Chen, X.W.; Lin, X. Big data deep learning: Challenges and perspectives. *IEEE Access* **2014**, *2*, 514–525. [CrossRef]
15. Alom, M.Z.; Taha, T.M.; Yakopcic, C.; Westberg, S.; Sidike, P.; Nasrin, M.S.; Hasan, M.; Van Essen, B.C.; Awwal, A.A.; Asari, V.K. A state-of-the-art survey on deep learning theory and architectures. *Electronics* **2019**, *8*, 292. [CrossRef]
16. Deng, L.; Liu, Y. *Deep Learning in Natural Language Processing*; Springer: Berlin/Heidelberg, Germany, 2018.
17. Deng, L.; Hinton, G.; Kingsbury, B. New types of deep neural network learning for speech recognition and related applications: An overview. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouve, BC, Canada, 26–31 May 2013; pp. 8599–8603.
18. Nishani, E.; Çiço, B. Computer vision approaches based on deep learning and neural networks: Deep neural networks for video analysis of human pose estimation. In Proceedings of the 2017 6th Mediterranean Conference on Embedded Computing (MECO), Bar, Montenegro, 11–15 June 2017; pp. 1–4.
19. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
20. Pandiya, M.; Dassani, S.; Mangalraj, P. Analysis of Deep Learning Architectures for Object Detection-A Critical Review. In Proceedings of the 2020 IEEE-HYDCON, Hyderabad, India, 11–12 September 2020; pp. 1–6.
21. Arora, D.; Garg, M.; Gupta, M. Diving deep in deep convolutional neural network. In Proceedings of the 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), Greater Noida, India, 18–19 December 2020; pp. 749–751.
22. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Region-based convolutional networks for accurate object detection and segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *38*, 142–158. [CrossRef] [PubMed]
23. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
24. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
25. Ryu, J.; Kim, S. Chinese character detection using modified single shot multibox detector. In Proceedings of the 2018 18th International Conference on Control, Automation and Systems (ICCAS), PyeongChang, Korea, 17–20 October 2018; pp. 1313–1315.
26. Chiu, Y.C.; Tsai, C.Y.; Ruan, M.D.; Shen, G.Y.; Lee, T.T. Mobilenet-SSDv2: An improved object detection model for embedded systems. In Proceedings of the 2020 International Conference on System Science and Engineering (ICSSE), Kagawa, Japan, 31 August–3 September 2020; pp. 1–5.
27. Heirman, J.; Selleri, S.; De Vleeschauwer, T.; Hamesse, C.; Bellemans, M.; Schoofs, E.; Haelterman, R. Exploring the possibilities of Extended Reality in the world of firefighting. In Proceedings of the 2020 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR), Utrecht, The Netherlands, 14–18 December 2020; pp. 266–273.
28. Andrade, T.M.; Smith-Creasey, M.; Roscoe, J.F. Discerning User Activity in Extended Reality Through Side-Channel Accelerometer Observations. In Proceedings of the 2020 IEEE International Conference on Intelligence and Security Informatics (ISI), Arlington, VA, USA, 9–10 November 2020; pp. 1–3.

29. Dandachi, G.; Assoum, A.; Elhassan, B.; Dornaika, F. Machine learning schemes in augmented reality for features detection. In Proceedings of the 2015 Fifth International Conference on Digital Information and Communication Technology and Its Applications (DICTAP), Beirut, Lebanon, 29 April–1 May 2015; pp. 101–105.
30. Sendari, S.; Anggreani, D.; Jiono, M.; Nurhandayani, A.; Suardi, C. Augmented reality performance in detecting hardware components using marker based tracking method. In Proceedings of the 2020 4th International Conference on Vocational Education and Training (ICOVET), Malang, Indonesia, 19 September 2020; pp. 1–5.
31. Mahurkar, S. Integrating YOLO Object Detection with Augmented Reality for iOS Apps. In Proceedings of the 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, 8–10 November 2018; pp. 585–589.
32. El Filali, Y.; Krit, S.D. Augmented reality types and popular use cases. *Int. J. Eng. Sci. Math.* **2019**, *8*, 91–97.
33. Poetker, B. What Is Augmented Reality? (+Most Common Types of AR Used Today). 2018. Available online: https://www.g2.com/articles/augmented-reality (accessed on 8 April 2022).
34. Gao, Y.F.; Wang, H.Y.; Bian, X.N. Marker tracking for video-based augmented reality. In Proceedings of the 2016 International Conference on Machine Learning and Cybernetics (ICMLC), Jeju Island, Korea, 10–13 July 2016; Volume 2, pp. 928–932.
35. Sendari, S.; Firmansah, A. Performance analysis of augmented reality based on vuforia using 3d marker detection. In Proceedings of the 2020 4th International Conference on Vocational Education and Training (ICOVET), Malang, Indonesia, 19 September 2020; pp. 294–298.
36. Vidya, K.; Deryl, R.; Dinesh, K.; Rajabommannan, S.; Sujitha, G. Enhancing hand interaction patterns for virtual objects in mobile augmented reality using marker-less tracking. In Proceedings of the 2014 International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 5–7 March 2014; pp. 705–709.
37. Beier, D.; Billert, R.; Bruderlin, B.; Stichling, D.; Kleinjohann, B. Marker-less vision based tracking for mobile augmented reality. In Proceedings of the The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, Tokyo, Japan, 7–10 October 2003; pp. 258–259.
38. Pooja, J.; Vinay, M.; Pai, V.G.; Anuradha, M. Comparative analysis of marker and marker-less augmented reality in education. In Proceedings of the 2020 IEEE International Conference for Innovation in Technology (INOCON), Bangalore, India, 6–8 November 2020; pp. 1–4.
39. Batuwanthudawa, B.; Jayasena, K. Real-Time Location based Augmented Reality Advertising Platform. In Proceedings of the 2020 2nd International Conference on Advancements in Computing (ICAC), Malabe, Sri Lanka, 10–11 December 2020; Volume 1, pp. 174–179.
40. Unal, M.; Bostanci, E.; Sertalp, E.; Guzel, M.S.; Kanwal, N. Geo-location based augmented reality application for cultural heritage using drones. In Proceedings of the 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 19–21 October 2018; pp. 1–4.
41. Argotti, Y.; Davis, L.; Outters, V.; Rolland, J.P. Dynamic superimposition of synthetic objects on rigid and simple-deformable real objects. *Comput. Graph.* **2002**, *26*, 919–930. [CrossRef]
42. Ketchell, S.; Chinthammit, W.; Engelke, U. Situated storytelling with SLAM enabled augmented reality. In Proceedings of the The 17th International Conference on Virtual-Reality Continuum and Its Applications in Industry, Brisbane, QLD, Australia, 14–16 November 2019; pp. 1–9.
43. Knopp, S.; Klimant, P.; Schaffrath, R.; Voigt, E.; Fritzsche, R.; Allmacher, C. Hololens ar-using vuforia-based marker tracking together with text recognition in an assembly scenario. In Proceedings of the 2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), Beijing, China, 10–18 October 2019; pp. 63–64.
44. Soulami, K.B.; Ghribi, E.; Labyed, Y.; Saidi, M.N.; Tamtaoui, A.; Kaabouch, N. Mixed-reality aided system for glioblastoma resection surgery using microsoft HoloLens. In Proceedings of the 2019 IEEE International Conference on Electro Information Technology (EIT), Brookings, SD, USA, 20–22 May 2019; pp. 79–84.
45. Lee, J.D.; Wu, H.K.; Wu, C.T. A projection-based AR system to display brain angiography via stereo vision. In Proceedings of the 2018 IEEE 7th Global Conference on Consumer Electronics (GCCE), Nara, Japan, 9–12 October 2018; pp. 130–131.
46. Vuforia Developer Library. Introduction to Model Targets. 2021. Available online: https://library.vuforia.com/articles/Solution/introduction-model-targets-unity.html (accessed on 8 April 2022).
47. Zhang, S.; Tian, J.; Zhai, X.; Ji, Z. Detection of Porcine Huddling Behaviour Based on Improved Multi-view SSD. In Proceedings of the 2020 Chinese Automation Congress (CAC), Shanghai, China, 6–8 November 2020; pp. 5494–5499.
48. Rios, A.C.; dos Reis, D.H.; da Silva, R.M.; Cuadros, M.A.D.S.L.; Gamarra, D.F.T. Comparison of the YOLOv3 and SSD MobileNet v2 Algorithms for Identifying Objects in Images from an Indoor Robotics Dataset. In Proceedings of the 2021 14th IEEE International Conference on Industry Applications (INDUSCON), Sao Paulo, Brazil, 15–18 August 2021; pp. 96–101.
49. Phadnis, R.; Mishra, J.; Bendale, S. Objects talk-object detection and pattern tracking using tensorflow. In Proceedings of the 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, India, 20–21 April 2018; pp. 1216–1219.
50. Lin, T. LabelImg. 2015. Available online: https://github.com/tzutalin/labelImg (accessed on 8 April 2022).
51. Kilic, I.; Aydin, G. Traffic sign detection and recognition using tensorflow's object detection API with a new benchmark dataset. In Proceedings of the 2020 International Conference on Electrical Engineering (ICEE), Istanbul, Turkey, 25–27 September 2020; pp. 1–5.
52. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the 28th Annual Conference on Neural Information Processing Systems, Montreal, Canada, 7–12 December 2015; pp. 91–99.

53. Park, Y.; Chin, S. An Efficient Method of Scanning and Tracking for AR. *Int. J. Adv. Cult. Technol.* **2019**, *7*, 302–307.
54. Enoxsoftware. About OpenCV for Unity. 2016. Available online: https://enoxsoftware.com/opencvforunity/documentation/about-opencv-for-unity (accessed on 8 April 2022).