CITIZEN SCIENCE:
THEORY AND PRACTICE

SmartCS: Enabling the Creation of Machine Learning–Powered Computer Vision Mobile Apps for Citizen Science Applications without Coding

RESEARCH PAPER

FAHIM HASAN KHAN
AKILA DE SILVA
GREGORY DUSEK
JAMES DAVIS
ALEX PANG

*Author affiliations can be found in the back matter of this article

ubiquity press

ABSTRACT

It is undeniable that citizen science contributes to the advancement of various fields of study. There are now software tools that facilitate the development of citizen science apps. However, apps developed with these tools rely on individual human skills to correctly collect useful data. Machine learning (ML)–aided apps provide on-field guidance to citizen scientists on data collection tasks. However, these apps rely on server-side ML support, and therefore need a reliable internet connection. Furthermore, the development of citizen science apps with ML support requires a significant investment of time and money. For some projects, this barrier may preclude the use of citizen science effectively. We present a platform that democratizes citizen science by making it accessible to a much broader audience of both researchers and participants. The SmartCS platform allows one to create citizen science apps with ML support quickly and without coding skills. Apps developed using SmartCS have client-side ML support, making them usable in the field, even when there is no internet connection. The client-side ML helps educate users to better recognize the subjects, thereby enabling high-quality data collection. We present several citizen science apps created using SmartCS, some of which were conceived and created by high school students.

CORRESPONDING AUTHOR:
Fahim Hasan Khan
University of California Santa Cruz, US
fkhan4@ucsc.edu

KEYWORDS:
machine learning application; computer vision; citizen science; mobile app building platform; system design

TO CITE THIS ARTICLE:
Khan, FH, de Silva, A, Dusek, G, Davis, J and Pang, A. 2024. SmartCS: Enabling the Creation of Machine Learning–Powered Computer Vision Mobile Apps for Citizen Science Applications without Coding. *Citizen Science: Theory and Practice*, 9(1): 14, pp. 1–16. DOI: https://doi.org/10.5334/cstp.642

INTRODUCTION

Citizen science is a form of scientific research that involves the general public as participants. The participants are typically not trained scientists, but rather individuals who are interested in or concerned about a particular subject and want to contribute to scientific knowledge (Aristeidou and Herodotou 2020; Haklay et al. 2021). Citizen science projects often involve monitoring and collecting visual data, such as images or videos, of various subjects. Participants may also be involved in designing experiments, analyzing results, and solving problems related to the research project. In the rest of this article, the term "researchers" refers to those who conduct the research projects, and "participants" refers to those who contribute to research projects via a citizen science platform (Eitzel et al. 2017). Citizen science benefits both researchers and participants in terms of data collection and learning experience. For example, the iNaturalist app allows participants to collect data while learning about plant and animal species (Horn et al. 2018).

With emerging technologies and shifting paradigms, citizen science platforms are also becoming mobile, making it easier for participants to collect visual data (Newman et al. 2012). Mobile devices, such as smartphones and tablets, equipped with multiple cameras and advanced processing capabilities, can perform complex computer vision (CV) tasks using machine learning (ML) (Qin et al. 2019). ML-enhanced customized mobile application software or apps have the potential to significantly improve the effectiveness of citizen science projects.

While anyone can engage in citizen science projects, some basic skills are necessary for effective data collection. Often, participants need to accurately identify and label objects, a task that can be challenging for nonexperts. Mobile apps with integrated machine learning (ML), capable of detecting objects of interest, can assist participants in efficiently collecting visual data and improving the data quality. This also opens the possibility of recruiting more volunteers by educating people about new topics and growing new interests among them (Roche et al. 2020).

Citizen science platforms such as Zooniverse (Simpson, Page, and De Roure 2014), SPOTTERON (Hummer and Niedermeyer 2018), Anecdata (Disney et al. 2017), etc., provide the service to build citizen science apps for crowdsourced research projects (Liu et al. 2021). While these platforms offer standardized purpose-specific tools and features, ML guidance is largely unavailable. A few apps, such as iNaturalist, have ML guidance through cloud servers. However, the required connectivity may be unavailable in remote locations. Moreover, existing open source systems like iNaturalist and Zooniverse are not designed to integrate with client-side ML (Simpson, Page, and De Roure 2014). Creating a new app with ML guidance de novo for every similar citizen science project is not ideal. For instance, "Seek by iNaturalist" was built from scratch to add client-side ML guidance for classifying plant and animal species, despite iNaturalist having an existing ML server (Horn et al. 2018).

This paper introduces an ML-integrated citizen science mobile app creation platform for faster app building and deployment without programming knowledge. Developing apps for citizen science involves considering many factors, such as design and technical build (Lemmens et al. 2021). Furthermore, the investment required for app design and development often spans from tens to hundreds of thousands of US dollars (Odenwald 2019). The cost of crafting apps, especially those with an extensive range of features, can hinder innovation (Howard et al. 2022). With SmartCS, users can bypass the complexities inherent in app development. Our platform offers pre-built features and templates within a single framework. This facilitates rapid prototyping and faster deployments. In addition to helping participants capture better data, the apps created by this platform can serve as educational tools to increase engagement in a wide variety of citizen science projects. We validate our platform's usefulness to researchers by asking a group of non-programmers to create apps and measuring their success and comfort in doing so. We further validate our platform's usefulness to the participants by comparing success at correctly identifying the subjects of data to be captured and through a survey of participant engagement.

MOTIVATION

Here, we discuss the challenges participants face in research data collection for rip current detection (Philip and Pang 2016). Rip currents are strong, seaward flowing currents that can occur on any beach with breaking waves, leading to an estimated 100 drownings a year in the US (Castelle et al. 2016; Gensini and Ashley 2010). To answer questions like "Which beaches have rip currents?" the researcher needs to gather and label imagery that contains rip currents from many different geographic locations. While an expert can visually spot rip currents, it can be challenging for non-experts (Brannstrom et al. 2015). However, various ML methods can detect rip currents, as demonstrated by recent works (de Silva et al. 2023; de Silva et al. 2021; Maryan et al. 2019). Mobile apps empowered by ML can assist nonexpert data collectors by visually showing them the detected rip currents using bounding boxes or similar visualization through the live camera feed.

The same concept as the example above applies to data collection in other fields, such as biological sciences, marine life, geomorphology, weather-related phenomena, etc. The ML-empowered apps allow nonexpert participants to learn and correctly detect the subjects through on-the-field assistance in these data collection scenarios. This is especially helpful for relatively hard-to-recognize or -differentiate objects such as rip currents.

RELATED WORKS

CITIZEN SCIENCE PLATFORMS

We examined several popular citizen science app creation platforms enabling people-powered mobile app development (Liu et al. 2021). Earlier, we mentioned Zooniverse, which features projects from diverse domains (Barber 2018; Simpson, Page, and De Roure 2014). One of the Zooniverse projects is OceanEYEs (Ra et al. 2022), which seeks volunteers to count and label fish, if there are any, in millions of unlabeled images collected from the ocean. Anecdata (Disney et al. 2017) is another free platform like Zooniverse, where both are primarily web portals with companion mobile apps. SPOTTERON (Liu et al. 2021) is a similar platform that exclusively functions through mobile apps with a uniform, easily customizable graphical user interface (GUI) for various projects. However, the base systems of these general-purpose citizen science app creation platforms do not support complex operations like ML model integration. The Citizen Science Association (Citizenscience.org, 2023) also maintains a list of major citizen science platforms, complete with a comparison table highlighting their most prominent features. ML-assisted data collection is not included in this comparison, as this feature is not common on these platforms (Cybertracker 2022; Ispotnature 2022; Epicollect 2023).

CITIZEN SCIENCE APPS WITH MACHINE LEARNING

Many custom-built citizen science applications have ML capabilities. We previously mentioned that iNaturalist (Horn et al. 2018) has server-side ML capabilities, and functions like a social network to connect nature observers. Leafsnap (Kumar et al. 2012), a mobile app for automatic plant species identification, is another example of a citizen science app that utilizes computer vision. Despite having many powerful features, Leafsnap's ML processing is performed on a cloud server, like iNaturalist, after the participants upload their images. Leafsnap's server-side ML processing takes 5.4 seconds per image, which is not fast enough to produce real-time results (Kumar et al. 2012). Although modern high-end servers perform much faster ML operations, server-side ML is not feasible for many real-time applications, especially citizen science apps intended for use in remote locations. Wildme.org (Wildme 2023), FathomNet (Katija et al. 2022; Fathomnet 2023), and many other web-only citizen science platforms also use server-side ML and have no mobile apps. Many existing applications were not designed as general-purpose citizen science apps, so while some are open source, developing a new app using any of them as a starting point would require significant programming expertise. Khan et al. (2021) presented a short paper discussing the integration of ML into citizen science projects. However, their approach required significant programming expertise to be effectively utilized at that time. A few apps, such as Pl@ntNet, incorporate client-side ML (Goëau et al. 2013; Plantnet 2022). Pl@ntNet is one of the most popular science apps and was also among the first to include client-side ML support, allowing it to work without an internet connection. However, as it is specifically designed for detecting plants, it cannot be used for other citizen science projects. However, similar to Seek by iNaturalist, it is specifically designed for detecting plants and cannot be used for other applications (Horn et al. 2018).

MOBILE APP CREATION PLATFORMS

The development of mobile apps involves a challenging process that includes various phases such as platform selection, writing, debugging, optimizing code, creating user interfaces, simulation, testing, and support (Joorabchi, Mesbah, and Kruchten 2013). Few customizable mobile app creation and deployment platforms, such as App Movement (Garbett et al. 2016), provide generic templates for the community to build apps. Model-driven development (MDD) (Balagtas-Fernandez and Hussmann 2008) has been adopted for mobile app development to simplify the process, reducing technical complexity and costs significantly. For example, Gharaat et al. (2021) proposed an MDD framework that enables novice app developers to model location-based apps by code transformation. Although MDD has improved the app creation process for developers, it does not support app customization and adding advanced features, such as ML, without writing code. Yiqun Li, Guo, and Chin (2015) proposed a platform to create mobile augmented reality apps without programming, but none of these systems support the integration of ML capabilities in the app.

Mobile applications are generally classified into three categories: native apps, hybrid apps, and mobile web apps (Huy and Vanthanh 2012; Umuhoza and Brambilla 2016). Native mobile apps are created specifically for a single platform (e.g., Google's Android or Apple's iOS) and leverage the hardware and software of that platform

to enhance the user experience. Hybrid apps, on the other hand, run through web browsers after installing on devices like native apps and are typically created like webpages. Hybrid apps are more capable of streamlining the development process but are not as fast or reliable as native apps. Web apps are adaptive websites that change layout, outlook, and accessibility when accessed from a mobile device. Due to the technical complexity of integrating client-side ML models, native apps are the most feasible option for creating ML-powered apps. ML model training platforms such as Roboflow (Ciaglia et al. 2022), Lobe.ai (Lobe 2023), Ultralytics HUB (Ultralytics 2023), etc., have basic app templates that can be used with the trained models. TensorFlow Lite (Abadi et al. 2015) provides similar simplified app templates. However, these templates are too basic for creating any real app without writing code and making substantial modifications. We explored various app-making platforms without coding available for mobile and PC platforms (Buildfire 2023; Appsgeyser 2023; Appypie 2023; Andromo 2023). However, these platforms primarily function as GUI makers with standard basic functionalities, such as text inputs and outputs, loading graphics, maps, calendars, websites, accessing system camera apps, etc. Integrating the complex process of loading ML models and providing computer vision functionalities is not available on any of these platforms.

MACHINE LEARNING MODELS FOR COMPUTER VISION ON MOBILE DEVICES

Typically, mobile devices have limited computational resources and power, which are major constraints for running ML models on such devices. Many recent research projects have focused on creating optimal models for computer vision tasks, in terms of accuracy, speed, resource efficiency, scalability, robustness, and generalizability. They employ deep learning models, which can be categorized into two types based on their underlying structures: one-stage and two-stage. The trend discovered in the current research describes how highly efficient one-stage paradigms will prioritize the prediction speed in frames per second (FPS). In contrast, two-stage models strive to achieve the best per-frame accuracy by employing a filtering stage and predicting stage. Owing to the limited computational resources of mobile devices, using two-stage models can be costly. As a solution, employing lightweight one-stage models for data collection and analysis on these devices can enable integration into a wider range of mobile applications that have computational constraints (Khan et al. 2021).

Modern vision tasks, such as object detection, image classification, semantic segmentation, etc., are mostly done using convolutional neural networks (CNN). The previously discussed categorization is about all ML models in general; for object detection specifically, the two categories of CNN-based ML models to choose from are (1) region-based detectors and (2) single-shot detectors (SSD). The computational resource-intensive two-stage region-based detectors, such as Faster R-CNN (Ren et al. 2016), have a region-proposal stage and a classifier stage. The limited computational power of today's mobile devices precludes their direct use. However, they have been utilized for real-time object detection via remote GPU servers (Lee et al. 2017). But these server-dependent systems are suitable only for deployment in locations with network access. SSD attains object detection using a single-stage CNN (W. Liu et al. 2016). YOLO (Adarsh, Rathi, and Kumar, 2020), EfficientDet (Tan, Pang, and Le 2020), and SSDs such as SSD MobileNet (Sandler et al. 2018; Chiu et al. 2020) are designed to perform in real time, sacrificing some accuracy (Huang et al. 2017). Additionally, Sun et al. (Sun et al. 2020) empirically showed that the SSD MobileNetv2 required the least amount of memory, thus making the SSD MobileNetv2 preferable for processing on mobile platforms. Similarly, variants of YOLOv8 (Ultralytics 2023), EfficientNet (Tan and Q. Le 2019), Inception (Xia, Xu, and Nan 2017), and MobileNet (Chiu et al. 2020) are optimized for real-time image classification on mobile devices. Likewise, YOLOv8-seg (Ultralytics 2023), U-Net MobileNetv2 (Siddique et al. 2021), MobileNetv2-DeepLab-v3 (Chen et al. 2017), etc. provide real-time segmentation on portable devices. We select and use the ML models most suitable for computer vision mobile apps for citizen science applications.

SYSTEM DESIGN OF THE PLATFORM

Our citizen science app creation platform combines multiple technical components to build the overall system, which can be divided into three steps: (1) training dataset creation, (2) ML model training, and (3) mobile app (iOS or Android) building, all of which our platform automates. The workflow of our system is presented in Figure 1.

The dataset consists of individual images or images extracted from videos to train ML models for computer vision. The type of labels needed for the image data depends on the type of task, such as object detection or image classification. The training images are labeled using bounding boxes around the objects to train a model for object detection. Labeling for image classification involves organizing the images into classes and labeling each class. Our system provides the required tools with instructions for formatting, labeling, and creating the training dataset.

Next, an ML model compatible with our system needs to be selected from a list. For each project, sufficient initial

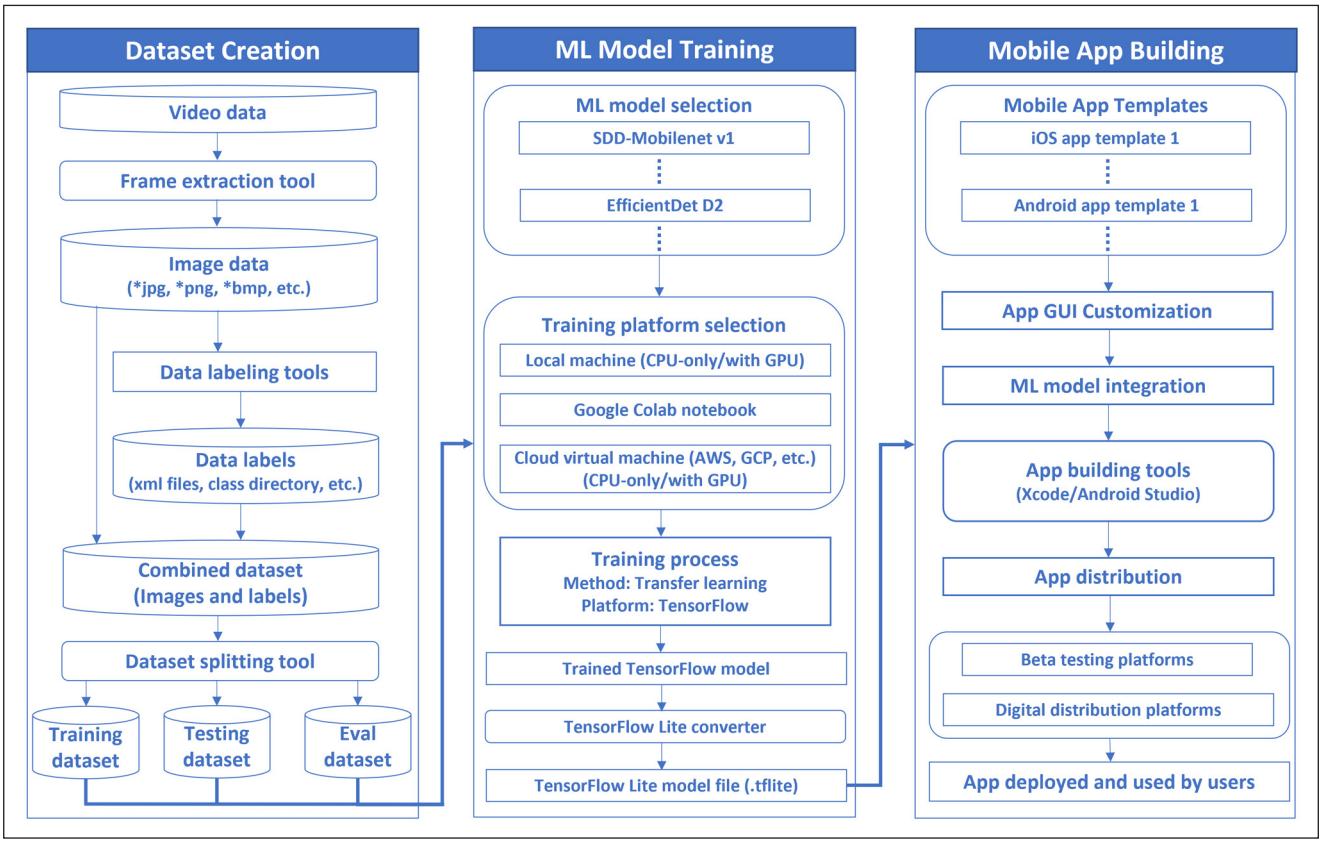

Figure 1 Overview and workflow of the components of our open-source citizen science app creation platform.

training data is needed to train a functioning model, which can be later improved via further training as more data are collected (Schwarz et al. 2018). For a project with an adequate dataset, the researchers can immediately train a model and quickly deploy the app. However, the project team must create a minimum training dataset for a project without initial data. Our platform includes the option to add an "Expert mode (No ML)" to the apps (Supplemental File 1: SmartCS App Studio User Manual). This feature allows volunteers to contribute to building up a dataset without relying on ML guidance. A back-end server collects these data, which can then be curated. After quality control, feedback can be provided to volunteers to facilitate learning, as well as for retraining models to improve their accuracy. Our system provides built-in guidance to run the training process on a local machine or a cloud server.

Finally, a template for the iOS or Android app needs to be selected from a collection of templates to fit the requirements of various citizen science projects. The template can be further customized to change GUI color, icon, logo, etc. Then, the app is built with the trained ML model integrated into it. The app's primary visual data collection tool is like a camera app with a live view with image and video capture function, where the detection results are shown using visualizations, such as bounding boxes and text labels. These visualizations help participants identify and record data on objects of interest. The models operate on mobile devices using native computational resources, without any server support (Figure 2). A server is necessary only for uploading the collected data. Tools for data uploading, user guides, and tutorials are also included in the templates.

IMPLEMENTATION

We created a desktop and a web-based version of our app creation platform to make it more accessible and versatile. For both versions, the apps are created through the three simple guided steps. Our platform is implemented as an open-source tool, utilizing other open-source software and additional tools such as AnyLabeling, TensorFlow, PyTorch, and Colab Jupyter Notebook (Abadi et al. 2015; Tzutalin 2015). An important advantage of open source is that it enables users with programming skills to help improve the platform by writing code to update and create new features, creating new templates, etc.

The desktop version is convenient for the user who wants to download it and run all the steps on a single machine capable of handling the ML training and app compilation.

We created the desktop version to run on Windows, macOS, and Linux. The web-based version works as an online service, providing the user interface as a website, which can be accessed through any device that has a standard web browser (Figure 3). Another advantage of the web-based version is that different steps can run on different local or cloud machines optimized for the specific step. For example, data labeling can be done on a laptop, as low computational resources are required. Then, the labeled data can be transferred to a GPU-optimized local or cloud machine for ML training. Because of this flexibility, the web-based version requires executing a few commands on the console and some file transfer operations. On the desktop version, the steps can be performed entirely by clicking some buttons and following the prompts, making it easy to be used by computer users with any skill level. Neither version requires programming skills or coding expertise.

DATASET CREATION

The object detection ML model training process expects still images as training data. We created a tool for extracting frames from the videos as still images if the available training data consists of video files. The frame extraction rate can be adjusted depending on the motion change rate of objects of interest. The data is labeled using free, open-source tools such as LabelImg (Tzutalin 2015) or AnyLabeling (AnyLabeling 2023). Our platform also supports converting labels obtained using Amazon's Mechanical Turk (Paolacci, Chandler, and Ipeirotis 2010). Additionally, we a provide some custom tools that we created using Python

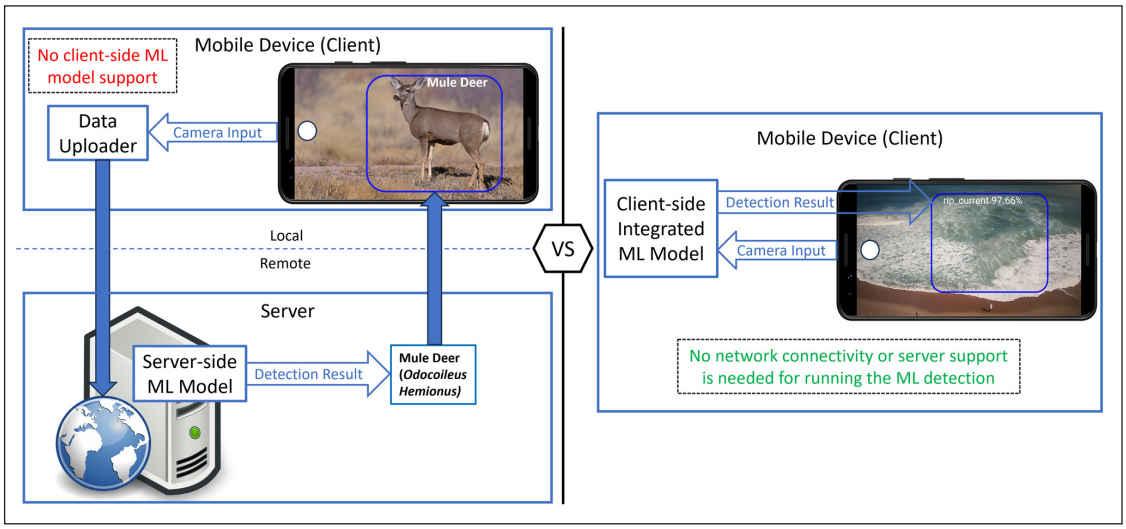

Figure 2 Server-side (*left*) versus client-side (*right*) machine learning (ML) models. We used client-side ML models in our implementation, which provided real-time object detection without any network connectivity or server-side processing requirements.

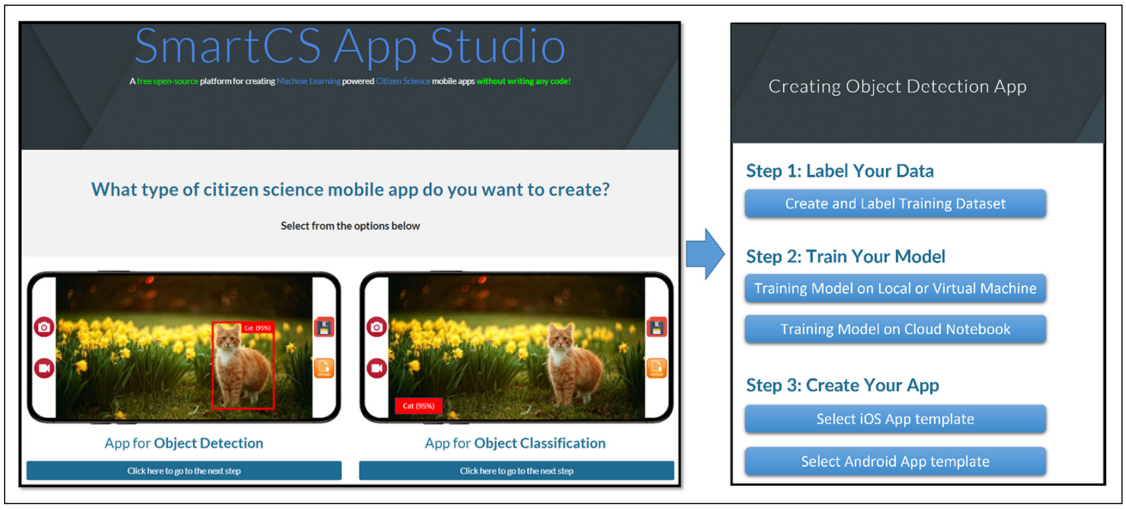

Figure 3 The web version of the platform was created for easy access and uses different feasible computational resources for different steps.

for additional dataset formatting, such as image format conversion, annotation file processing, etc. The datasets for the classification tasks are created by simply putting the images for each class into separate directories, where each directory's name works as the class names. Our system provides a tool to split the data into three sets, training, test, and evaluation, using a default ratio of 6:2:2 or user-defined splitting ratio (Reitermanova 2010).

MACHINE LEARNING MODEL TRAINING

This next step allows the user to train the model on a local computer, a cloud virtual machine, or a cloud Python notebook. Python notebooks (Supplemental File 1: SmartCS App Studio User Manual) available via Google Colab provide access to free GPU resources. We use small and lightweight models optimized for mobile devices from the TensorFlow Lite library for training (Abadi et al. 2015). Our platform provides the options to select the most appropriate model based on the information provided in Supplemental File 2: Appendix (B), Table B.1 and the planned usage of the app (Sanchez, Romero, and Morales 2020). Note that detection accuracy of the utilized ML models is considered reliable based on its mean average precision (mAP) (Padilla, Netto, and Da Silva 2020). The models are then custom trained using the dataset created in the previous step by the transfer learning process (Alsing 2018). We provide tutorial documentation and video for model training (Supplemental File 1: SmartCS App Studio User Manual).

While the training runs, a console shows the loss function value for the current training step. The lower the loss, the better a model has learned to detect the objects (Wang et al. 2020). The training needs to run until the model converges, which is indicated when the loss function value drops below a certain threshold (Allen-Zhu, Li, and Song 2019).

The training time depends on many factors such as the ML model, hardware (CPU only or GPU, amount of system memory, etc.) used for the training, and the training dataset (size, number of classes, etc.) (Lim, Loh, and Shih 2000; Jordan and Mitchell 2015). Further discussion about guidance on setting up these training parameters in the platform is included in Supplemental File 2: Appendix (B). ML model training for the case studies presented in this paper took about 2 to 3 days using inexpensive CPU-only machines, which can be done within a few hours using more expensive GPU machines.

MOBILE APP BUILDING

After training, the models are converted to TensorFlow Lite format, compatible with the platform's iOS and Android app templates. The app is built using the platform-specific build tool (Xcode for iOS or Android Studio for Android) and uploaded to digital distribution platforms for deployment. A paid developer account is required for Android and iOS to distribute the apps publicly through the official app distribution services. However, for initial app testing, it can be freely installed on a mobile device by connecting it to the USB cable to the computer where it was built. For easy deployment of apps on the Google Play Store and iOS App Store, we integrated Fastlane (Katz 2018) into our platform. Fastlane is an open-source platform that automates beta deployments and releases for mobile Android and iOS apps.

For user accounts, citizen science projects, and app management on our platform, we utilized Firebase Authentication (Moroney 2017), an open-source user account management system. It facilitates managing user identities and authentication securely using various sign-in methods, including email and password, anonymous, and federated identity providers.

RESULTS

This section presents a wide variety of citizen science apps created using SmartCS. The apps use ML models to aid users in collecting data for citizen science projects or as educational tools. Three apps, RipSnap, Seal vs. Sea Lion, and Vehicle Object Detection, were created for university research projects. In contrast, three other apps, Recycle This, TidalNow, and Sk.in, were created by high school students with no prior programming experience. Due to space constraints, we provide details on two of these example apps and briefly describe the other four use cases in Supplemental File 2: Appendix (A).

USE CASE: RECYCLE THIS

The practice of recycling is essential for the environment and the future of our planet (Al-Salem, Lettieri, and Baeyens 2009). Data collection on recyclable objects is necessary for optimizing recycling processes. However, there is a lack of clarity about what and how to recycle, with surveys showing that up to 62 percent of Americans lack recycling knowledge (Waste360 2023). The mobile app Recycle This, created by a high school student, incorporates ML for real-time detection and collection of data on common household recyclables (Figure 4). The project focused on classifying objects like glass bottles, plastic containers, cardboard, paper, and aluminum cans, and 2,500 training images were sourced from the public image datasets (Kaggle 2022; Images.cv 2023; Roboflow 2023). In addition to being a data collection tool, the app also acts as an educational tool by providing information and clarification on the recyclability of everyday waste objects. With widespread distribution, it can raise public awareness.

SmartCS enabled the creation of the app without writing codes, and subsequent studies were published as conference papers (Yeh and Khan 2022).

USE CASE: RipSnap

The importance of rip current detection was discussed earlier. The citizen science app RipSnap is based on the idea of CoastSnap (Harley et al. 2018), where app users contribute snapshots of coastlines from fixed docking stations to study coastal erosion and other processes. RipSnap extends the idea of collecting ML-detected videos of rips with location metadata. Data collected through RipSnap can help validate a rip current forecast model (Dusek and Seim 2013). The app's primary aim is to enhance beach safety by educating users about the presence of rip currents (Figure 5). A lightweight ML model integrated into this app assists nonexpert participants in identifying and collecting these valuable rip current data. The training dataset consists of 3,360 labeled images of two classes of rip currents, a combination of datasets from de Silva et al. (2021), and additional data collected using drones and beach monitoring webcams.

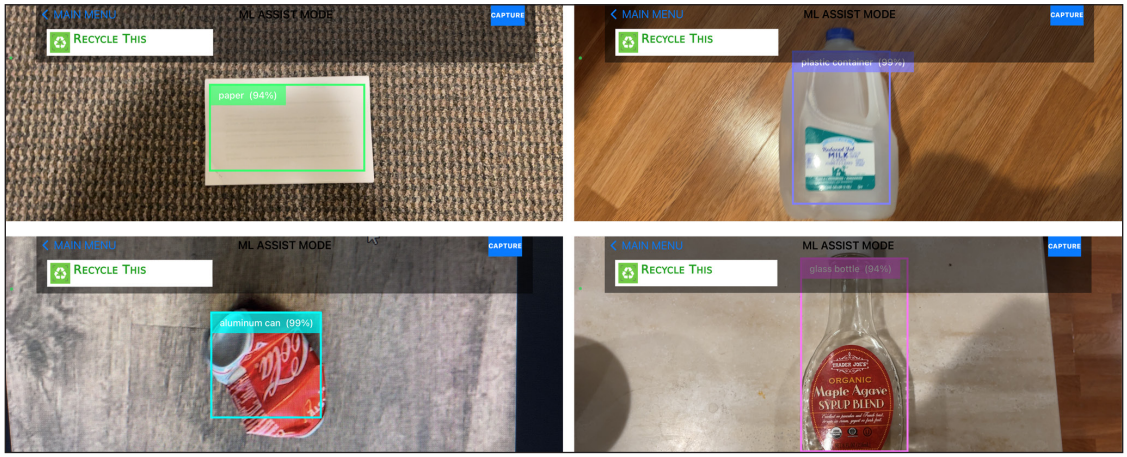

Figure 4 This figure illustrates the type of materials that the "Recycle This" app can detect: papers, aluminum cans, plastic containers, and glass bottles.

Figure 5 Appearance of the RipSnap app with examples of rip currents detected by the app. The location of the rip current is visualized using the red bounding box with the label and the confidence score of detection.

FEEDBACK

We collected user feedback from two user groups: (1) researchers who used the platform to create their apps, and (2) beta testers who used the created apps. The research questions and objectives are described before each study (Lazar, Feng, and Hochheiser 2017). The main goals of these user studies were usability testing of the app creation platform and establishing the effectiveness of the created apps. The findings of the user studies are discussed in this section.

USER STUDY 1: APP CREATORS

Our research question for this user study is, "Will people without programming experience be able to create citizen science apps using SmartCS?" The objective was to collect feedback and evidence demonstrating that individuals without programming experience can successfully create citizen science apps using SmartCS.

This study gathered insights from 10 high school students, with 5 male and 5 female participants, all of whom did not have prior programming experience and were selected among science internship program participants by an independent committee. These students, ranging in age from 14 to 17 years, engaged as researchers to develop their own applications utilizing SmartCS. These students were tasked with individually creating apps, a challenge that all except one were able to successfully complete. The process of app creation took them between 1 to 2 weeks, a timeframe that varied depending on the complexity of data labeling and the time required for machine learning model training, which in turn was influenced by the number of available GPUs. More details about the apps are provided in Supplemental File 2: Appendix (A).

For this study, we collected both quantitative and qualitative data from users. Participants' experiences and interactions with SmartCS were recorded using an online form, facilitating the collection of quantitative data through Likert scales. Additionally, the feedback form enabled the gathering of qualitative data through multiple open-ended responses. We also carefully observed participants' interactions with the system to gain deeper insights into their user experience. We observed that users could easily learn to use the platform by following the provided user guides and tutorials. Whenever users required additional help beyond what was available in the existing resources, we updated the guides and tutorials accordingly. Their main task was to learn how to navigate the platform's GUI. These observations helped us identify patterns and challenges within the user interface and interaction design that were not immediately apparent through quantitative feedback alone.

Figure 6 summarizes the quantitative user feedback on the app creation platform. Despite the small sample size, we can obtain some insights from the survey. The users were quite satisfied with the "Free to Use" aspect, as everyone gave it the highest rating. The "Final App as Per Expectation" feature has the lowest mean score,

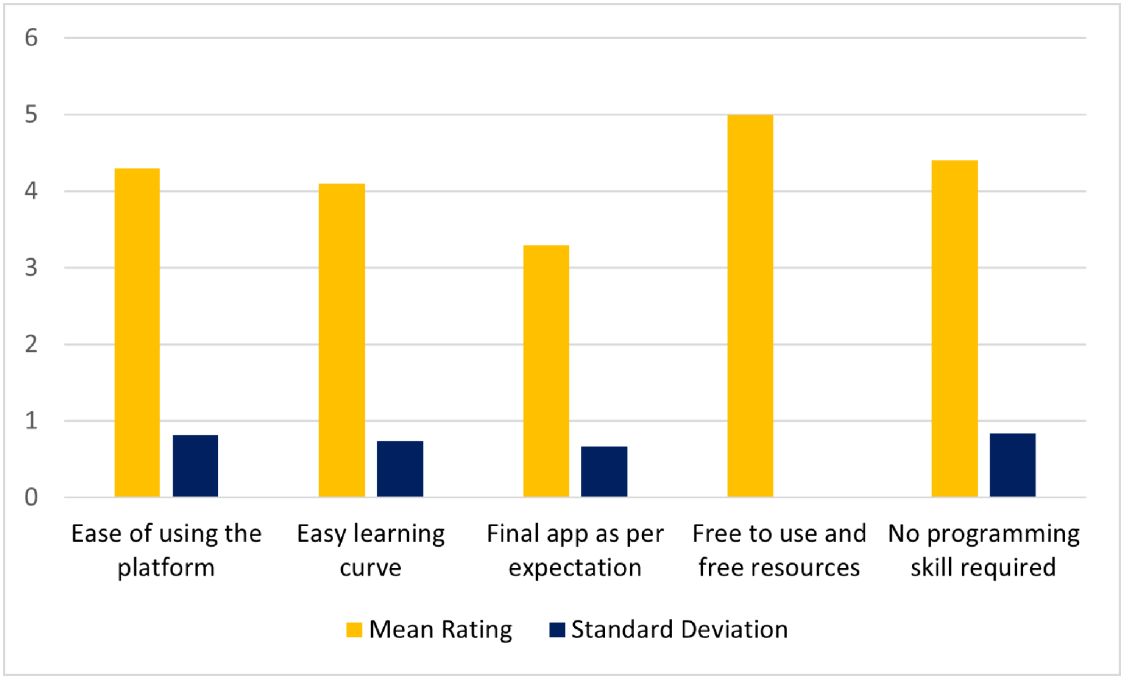

Figure 6 Summary of scores from the user study by app creators. Participants were asked to rate their experience of using the platform on a 5-point Likert scale from "Poor" (a score of 1) to "Excellent" (a score of 5).

indicating the diversity of expectations among users. However, this may be due to the limited templates and customization options available in the current version, as suggested by the users' qualitative feedback regarding their unmet expectations. Users expressed the need for more application-specific features and customizations tailored to each app's unique purpose. However, providing such a high level of customization is challenging with a general-purpose tool, requiring a balance between ease of use and flexibility. This limitation can be partially addressed by providing more templates. The "No Programming Skill Required" question has the highest variability, which may be due to the different levels of computer exposure among the users. Overall, this feedback suggests that the platform is user-friendly and convenient for nonexpert users.

USER STUDY 2: APP USERS

The research question for user study 2 is, "Are the citizen science apps created using SmartCS useful and easy to use?" The objective was to collect feedback and evidence demonstrating that citizen science apps developed with SmartCS, featuring simple designs and ML guidance, are both useful and easy to use.

Three apps, RipSnap, Recycle This, and Tidal-Now, underwent user testing with 38, 21, and 30 testers, respectively. Participants provided feedback on their user experience through an online form (Figure 7). Users generally found the apps easy to use. The GUI received the lowest rating, likely due to their simplistic appearance. We plan to improve upon this in future work. ML capabilities and usefulness were generally rated highly (either 4 or 5) across all three apps. These results indicate that apps created with SmartCS are user-friendly and serve their intended purpose well. This study was conducted through an online user study via open beta testing, where the apps were distributed using Apple's beta testing platform, TestFlight.

QUALITATIVE FEEDBACK

We gathered qualitative feedback from participants of both user studies through open-ended comments and verbal interviews. A primary suggestion from users who created apps was to provide more resources, help files, and video tutorials to guide them through the process. A recurring observation across all study groups was the simplicity and limited features of the user interface. However, this is more of a design choice rather than a technical limitation. Most participants strongly agreed that they appreciated the inclusion of ML for all citizen science use cases. They were impressed with the apps' ability to detect complex phenomena such as rip currents in real time.

CONCLUSION

This article presents SmartCS, a platform for building mobile apps with client-side ML-based guidance for citizen science without writing code. The apps created using the platform enhance data collection quality and efficiency

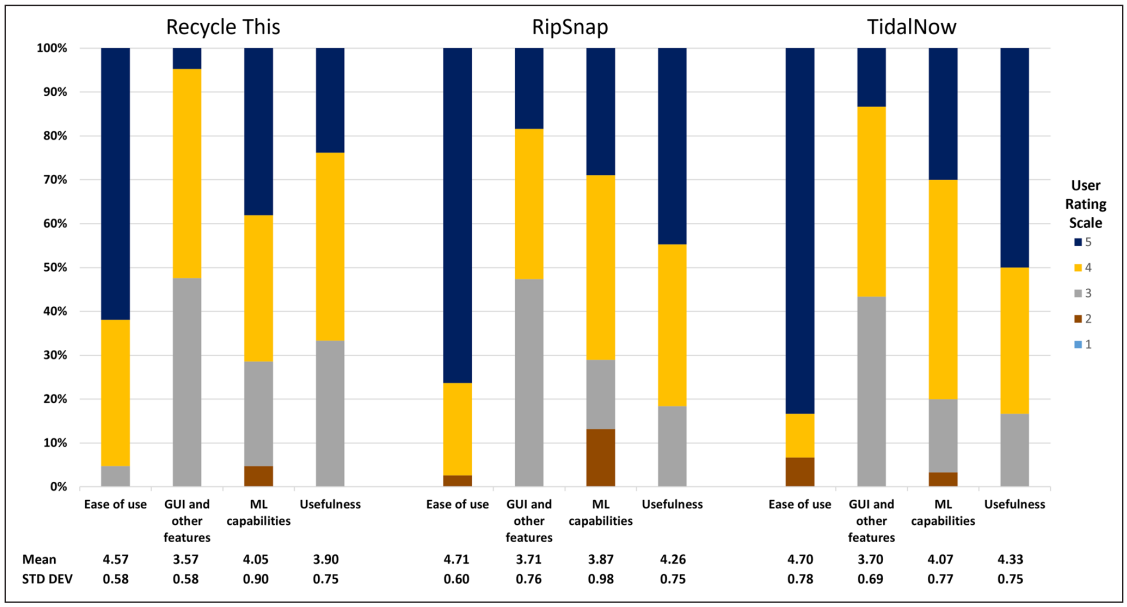

Figure 7 Summary of scores from the user study by users of three apps. Participants were asked to rate their experience of using the platform on a 5-point Likert scale from "Poor" (a score of 1) to "Excellent" (a score of 5). GUI: graphical user interface, ML: machine learning, STD DEV: standard deviation.

through ML-based guidance, even without internet connectivity in the field. The ML guidance also allows the apps to function as educational tools for participants who may not be familiar with the subject of the data being collected. We demonstrate the use of the authoring platform with six example use cases. We also present user studies to illustrate the app creation platform's usability and the effectiveness of the created apps for citizen science applications, highlighting its potential to engage a broader audience in citizen science activities. The feedback suggests areas for improvement, such as offering more resources and enhancing the user interface, but overall, the platform received positive feedback for its usability and the inclusion of ML capabilities.

The current apps enable expert users to utilize the apps without ML support, while nonexpert users can benefit from ML assistance for data collection. We plan to facilitate a seamless transition between ML-supported and non-ML modes through in-app guidance in future developments. It is important to note that the ML guidance used in the apps is not infallible and can produce false positive and false negative detections. It would be interesting to see if a collaborative approach between humans and ML can perform even better than with ML only. In such an arrangement, the human will have the option of overriding the ML detection in instances where they are confident and rely on ML detections otherwise. We believe that it is possible to improve overall accuracy and automate further the data collection process. Furthermore, the data collected when humans overrode ML suggestions can be used to improve and refine the ML model further, which we plan to investigate in the future.

DATA AVAILABILITY STATEMENT

The web-based version of SmartCS is available at https://smartctsc.github.io/SmartCS/. The entire codebase is accessible as open source in a public repository located at https://github.com/SmartCtSc.

SUPPLEMENTAL FILES

The supplemental files for this article can be found as follows:

- **Supplemental File 1.** SmartCS App Studio User Manual. DOI: https://doi.org/10.5334/cstp.642.s1
- **Supplemental File 2.** Appendix. DOI: https://doi.org/10.5334/cstp.642.s2

ETHICS AND CONSENT

This research (Study HS-FY2023-62) has been approved by the Office of Research Compliance Administration (ORCA), University of California, Santa Cruz, and informed consent is obtained from all participants involved in the study.

ACKNOWLEDGEMENTS

The authors thank the high school students from the UCSC Science Internship Program, the SmartCS apps user study participants, lifeguards, and the volunteers. We also acknowledge the support from the Google Cloud Research Credits program and AWS Cloud Credit for Research.

FUNDING INFORMATION

This work is partially funded by the following grants: Southeast Coastal Ocean Observing Regional Association (SECOORA) sub-award from NOAA award number NA20NOS0120220, the US Coastal Research Program (USCRP) through a Sea Grant award number NA23OAR4170121, and a grant from the UCSC Center for Coastal Climate Resilience. The statements, findings, conclusions, and recommendations are those of the authors and do not necessarily reflect the views of SECOORA, NOAA, or UCSC. The content of the information provided in this publication does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred.

COMPETING INTERESTS

The authors have no competing interests to declare.

AUTHOR CONTRIBUTIONS

F.H.K. contributed to the conceptualization, methodology, software development, validation, formal analysis, investigation, data curation, original draft writing, manuscript reviewing and editing, and visualization. A. de S. provided validation, formal analysis, and participated in original draft writing, reviewing and editing the manuscript. G.D. contributed to validation, formal analysis, investigation, resource management, data curation, original draft writing, manuscript reviewing and editing, supervision, and funding acquisition. J.D. was responsible for validation, formal

analysis, investigation, writing the original draft, reviewing and editing the manuscript, visualization, and supervision. A.P. contributed to the conceptualization, methodology, validation, formal analysis, investigation, writing of the original draft, reviewing and editing of the manuscript, visualization, supervision, and funding acquisition.

AUTHOR AFFILIATIONS

Fahim Hasan Khan orcid.org/0000-0003-3130-6259
University of California Santa Cruz, US

Akila de Silva orcid.org/0000-0002-7553-7270
University of California Santa Cruz, US

Gregory Dusek orcid.org/0000-0003-0289-1356
NOAA/National Ocean Service, US

James Davis orcid.org/0000-0002-1413-2616
University of California Santa Cruz, US

Alex Pang orcid.org/0000-0003-0720-7162
University of California Santa Cruz, US

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M. and **Ghemawat, S.** (2015) TensorFlow: Large-scale machine learning on heterogeneous systems Available at https://www.tensorflow.org (Last accessed 10 June 2024).

Adarsh, P., Rathi, P. and **Kumar, M.** (2020) YOLO v3-Tiny: Object Detection and Recognition using one stage improved model. In: *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*. Coimbatore, India on 6 March 2020, pp. 687–694. DOI: https://doi.org/10.1109/ICACCS48705.2020.9074315

Al-Salem, S.M., Lettieri, P. and **Baeyens, J.** (2009) Recycling and recovery routes of plastic solid waste (PSW): A review. *Waste Management*, 29(10), pp. 2625–2643. DOI: https://doi.org/10.1016/j.wasman.2009.06.004

Allen-Zhu, Z., Li, Y. and **Song, Z.** (2019) A convergence theory for deep learning via over-parameterization. In: *International Conference on Machine Learning*. Long Beach, CA on 24 May 2019, pp. 242–252.

Alsing, O. (2018) *Mobile object detection using TensorFlow Lite and transfer learning*. MA thesis. KTH, School of Electrical Engineering and Computer Science (EECS).

Andromo. (2023) Create an App for Android and iOS without Coding. Available at https://www.andromo.com/ (Last accessed 10 June 2023).

AnyLabeling. (2023) AnyLabeling – No-Code Labeling Tool. Available at https://anylabeling.nrl.ai/ (Last accessed 10 June 2023).

AppsGeyser. (2023) Create an App with No Coding Skills. Available at: https://appsgeyser.com (Last accessed 10 June 2023).

Appypie. (2023) Create an App with Appy Pie's App Builder. Available at: https://www.appypie.com (Last accessed 10 June 2023).

Aristeidou, M. and **Herodotou, C.** (2020) Online citizen science: A systematic review of effects on learning and scientific literacy. *Citizen Science: Theory and Practice*, 5(1), pp. 1–12. DOI: https://doi.org/10.5334/cstp.224

Balagtas-Fernandez, F.T. and **Hussmann, H.** (2008) Model-driven development of mobile applications. In: *2008 23rd IEEE/ACM International Conference on Automated Software Engineering*. L'aquila, Italy on 15 September 2008, pp. 509–512. DOI: https://doi.org/10.1109/ASE.2008.94

Barber, S.T. (2018) The Zooniverse is expanding: Crowdsourced solutions to the hidden collections problem and the rise of the revolutionary cataloging interface. *Journal of Library Metadata*, 18(2), pp. 85–111. DOI: https://doi.org/10.1080/19386389.2018.1489449

Brannstrom, C., Brown, H.L., Houser, C., Trimble, S. and **Santos, A.** (2015) "You can't see them from sitting here": Evaluating beach user understanding of a rip current warning sign. *Applied Geography*, 56, pp. 61–70. DOI: https://doi.org/10.1016/j.apgeog.2014.10.011

Buildfire. (2023) Create an App with BuildFire's App Maker. Available at: https://www.buildfire.com (Last accessed 10 June 2023).

Castelle, B., Scott, T., Brander, R.W. and **McCarroll, R.J.** (2016) Rip current types, circulation and hazard. *Earth Science Reviews*, 163, pp. 1–21. ISSN: 0012-8252. DOI: https://doi.org/10.1016/j.earscirev.2016.09.008

Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K. and **Yuille, A.L.** (2017) Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, *40*(4), pp. 834–848. DOI: https://doi.org/10.1109/TPAMI.2017.2699184

Chiu, Y.C., Tsai, C.Y., Ruan, M.D., Shen, G.Y. and **Lee, T.T.** (2020) August. Mobilenet-SSDv2: An improved object detection model for embedded systems. In: *2020 International conference on system science and engineering (ICSSE)*. Kagawa, Japan on 31 August 2020, pp. 1–5. DOI: https://doi.org/10.1109/ICSSE50014.2020.9219319

Ciaglia, F., Zuppichini, F.S., Guerrie, P., McQuade, M. and **Solawetz, J.** (2022) Roboflow 100: A rich, multi-domain object detection benchmark. *arXiv preprint* arXiv:2211.13523. DOI: https://doi.org/10.48550/arXiv.2211.13523

Citizenscience.org. (2023) Platforms for hosting participatory science projects – Citizen Science Association. Available at: https://citizenscience.org/platformsfor- hosting-participatory-science-projects (Last accessed 10 June 2023).

Cybertracker. (2022) CyberTracker: The Most Efficient Way of Field Data Collection. Available at: https://cybertracker.org (Last accessed 10 June 2022).

de Silva, A., Mori, I., Dusek, G., Davis, J. and **Pang, A.** (2021) Automated rip current detection with region based convolutional neural networks. *Coastal Engineering*, 166, p. 103859. DOI: https://doi.org/10.1016/j.coastaleng.2021.103859

de Silva, A., Zhao, M., Stewart, D., Hasan, F., Dusek, G., Davis, J. and **Pang, A.** (2023) RipViz: Finding Rip Currents by Learning Pathline Behavior. *IEEE Transactions on Visualization and Computer Graphics*. DOI: https://doi.org/10.1109/TVCG.2023.3243834

Disney, J., Bailey, D., Farrell, A. and **Taylor, A.** (2017) Next generation citizen science using Anecdata. org. *Maine Policy Review*, 26(2), pp. 70–79. DOI: https://doi.org/10.53558/PHKW8135

Dusek, G. and **Seim, H.** (2013) A probabilistic rip current forecast model. *Journal of Coastal Research*, 29(4), pp. 909–925. DOI: https://doi.org/10.2112/JCOASTRES-D-12-00118.1

Eitzel, M., Cappadonna, J., Santos-Lang, C., Duerr, R., West, S.E., Virapongse, A., Kyba, C., Bowser, A., Cooper, C., Sforzi, A. and **Metcalfe, A.** (2017) Citizen science terminology matters: Exploring key terms. *Citizen science: Theory and practice*, pp. 1–20. DOI: https://doi.org/10.5334/cstp.96

Epicollect. (2023) Epicollect5: Free and Easy-to-use Mobile Data Collection. Available at: https://five.epicollect.net (Last accessed 10 June 2023).

Fathomnet. (2023) FathomNet – An Open-Source Image Database for Ocean Exploration. Available at: https://fathomnet.org (Last accessed 10 June 2023).

Garbett, A., Comber, R., Jenkins, E. and **Olivier, P.** (2016) App movement: A platform for community commissioning of mobile applications. In: *Proceedings of the 2016 CHI conference on human factors in computing systems*, San Jose, California on 7 May 2016, pp. 26–37. DOI: https://doi.org/10.1145/2858036.2858094

Gensini, V.A. and **Ashley, W.S.** (2010) An examination of rip current fatalities in the United States. *Natural Hazards*, 54(1), pp. 159–175. DOI: https://doi.org/10.1007/s11069-009-9458-0

Gharaat, M., Sharbaf, M., Zamani, B. and **Hamou-Lhadj, A.** (2021) ALBA: a model-driven framework for the automatic generation of android location-based apps. *Automated Software Engineering*, *28*, pp.1–45. DOI: https://doi.org/10.1007/s10515-020-00278-3

Goëau, H., Bonnet, P., Joly, A., Bakić, V., Barbe, J., Yahiaoui, I., Selmi, S., Carré, J., Barthélémy, D., Boujemaa, N. and **Molino, J.F.** (2013) Pl@ ntnet mobile app. In: *Proceedings of the 21st ACM international conference on Multimedia*, Chengdu, China on 20 October 2021, pp. 423–424. DOI: https://doi.org/10.1145/2502081.2502251

Haklay, M., Dörler, D., Heigl, F., Manzoni, M., Hecker, S., Vohland, K., Vohland, K., Land-Zandstra, A., Ceccaroni, L., Lemmens, R. and **Perelló, J.** (2021) What is citizen science? The challenges of definition. *The science of citizen science*, 2, pp. 13–33. DOI: https://doi.org/10.1007/978-3-030-58278-4_2

Harley, M., Kinsela, M., Sánchez-García, E.S. and **Vos, K.** (2018) December. CoastSnap: crowd-sourced shoreline change mapping using smartphones. In: *AGU Fall Meeting Abstracts*, (Vol. 2018), pp. EP52D–26.

Horn, G. V., Mac Aodha, O., Song, Y., Cui, Y., Sun, C., Shepard, A., Adam, H., Perona, P. and **Belongie, S.** (2018) The inaturalist species classification and detection dataset. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, Salt Lake City, Utah on 18 June 2018 pp. 8769–8778. DOI: https://doi.org/10.48550/arXiv.1707.06642

Howard, L., van Rees, C.B., Dahlquist, Z., Luikart, G. and **Hand, B.K.,** (2022) A review of invasive species reporting apps for citizen science and opportunities for innovation. *NeoBiota*, *71*, pp. 165–188. DOI: https://doi.org/10.3897/neobiota.71.79597

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S. and **Murphy, K.** (2017) Speed/accuracy trade-offs for modern convolutional object detectors. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, Honolulu, Hawaii on 21 July 2017 pp. 7310–7311. DOI: https://doi.org/10.48550/arXiv.1611.10012

Hummer, P. and **Niedermeyer, C.** (2018) February. Don't walk alone: Synergy effects for citizen science created through adaptive platform design in SPOTTERON. In: *Austrian Citizen Science Conference 2018*, Salzburg, Austria on 3 February 2018, p. 66.

Huy, N.P. and **Vanthanh, D.** (2012) December. Evaluation of mobile app paradigms. In: *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia*, Bali, Indonesia on 3 December 2012, pp. 25–30. DOI: https://doi.org/10.1145/2428955.2428968

Images.cv. (2023) Images.cv: Your Machine Learning and Data Science Community. Available at: https://images.cv (Last accessed 10 June 2023).

Ispotnature. (2022) iSpot Nature: Your place to share nature. Available at: https://www. ispotnature.org (Last accessed 10 June 2022).

Joorabchi, M.E., Mesbah, A. and **Kruchten, P.** (2013) October. Real challenges in mobile app development. In: *2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, Baltimore, MD on 10 October 2013, pp. 15–24. DOI: https://doi.org/10.1109/ESEM.2013.9

Jordan, M.I. and **Mitchell, T.M.** (2015) Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), pp.255–260. DOI: https://doi.org/10.1126/science.aaa8415

Kaggle. (2022) Kaggle: Your Machine Learning and Data Science Community. Available at: https://www.kaggle.com (Last accessed 10 June 2022).

Katija, K., Orenstein, E., Schlining, B., Lundsten, L., Barnard, K., Sainz, G., Boulais, O., Cromwell, M., Butler, E., Woodward, B. and **Bell, K.L.** (2022) FathomNet: A global image database for enabling artificial intelligence in the ocean. *Scientific reports*, *12*(1), p.15914. DOI: https://doi.org/10.1038/s41598-022-19939-2

Katz, D. (2018) *Continuous delivery for mobile with fastlane: automating mobile application development and deployment for iOS and Android*. Packt Publishing Ltd.

Khan, F.H., de Silva, A., Dusek, G., Davis, J. and **Pang, A.** (2021) October. Authoring platform for mobile citizen science apps with client-side ml. In: *Companion Publication of the 2021 Conference on Computer Supported Cooperative Work and Social Computing*, Virtual Event, USA on 23 October 2021, pp. 89–94. DOI: https://doi.org/10.1145/3462204.3481743

Kumar, N., Belhumeur, P.N., Biswas, A., Jacobs, D.W., Kress, W.J., Lopez, I.C. and **Soares, J.V.** (2012) Leafsnap: A computer vision system for automatic plant species identification. In: *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision*, Florence, Italy on 7 October 2012, pp. 502–516. DOI: https://doi.org/10.1007/978-3-642-33709-3_36

Lazar, J., Feng, J.H. and **Hochheiser, H.** (2017) *Research methods in human-computer interaction*. Morgan Kaufmann.

Lee, J., Wang, J., Crandall, D., Šabanović, S. and **Fox, G.** (2017) April. Real-time, cloud-based object detection for unmanned aerial vehicles. In: *2017 First IEEE International Conference on Robotic Computing (IRC)*, Taichung, Taiwan on 10 April 2017. DOI: https://doi.org/10.1109/IRC.2017.77

Lemmens, R., Antoniou, V., Hummer, P. and **Potsiou, C.** (2021) Citizen science in the digital world of apps. *The science of citizen science*, *461*. DOI: https://doi.org/10.1007/978-3-030-58278-4_23

Li, Y., Guo, A. and **Chin, C.L.** (2015) A platform for mobile augmented reality app creation without programming. In: *SIGGRAPH Asia 2015 Mobile Graphics and Interactive Applications*, Kobe, Japan on 2 November 2015, pp. 1–1. DOI: https://doi.org/10.1145/2818427.2818452

Lim, T., Loh, W. and **Shih, Y.** (2000) A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning*, 40(3), pp. 203–228. DOI: https://doi.org/10.1145/2818427.2818452

Liu, H.Y., Dörler, D., Heigl, F. and **Grossberndt, S.** (2021) Citizen science platforms. *The science of citizen science*, 22, pp. 439–459. DOI: https://doi.org/10.1007/978-3-030-58278-4_22

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y. and **Berg, A.C.** (2016) Ssd: Single shot multibox detector. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands on 11 October 2016*, pp. 21–37. DOI: https://doi.org/10.1007/978-3-319-46448-0_2

Lobe. (2023). Lobe: An Easy-to-Use Machine Learning Tool. Available at: https://lobe.ai (Last accessed 10 March 2023).

Maryan, C., Hoque, M.T., Michael, C., Ioup, E. and **Abdelguerfi, M.** (2019) Machine learning applications in detecting rip channels from images. *Applied Soft Computing*, *78*, pp. 84–93. DOI: https://doi.org/10.1016/j.asoc.2019.02.017

Moroney, L. (2017) Using authentication in firebase. *The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform*, pp. 25–50. DOI: https://doi.org/10.1007/978-1-4842-2943-9_2

Newman, G., Wiggins, A., Crall, A., Graham, E., Newman, S. and **Crowston, K.** (2012) The future of citizen science: emerging technologies and shifting paradigms. *Frontiers in Ecology and the Environment*, *10*(6), pp. 298–304. DOI: https://doi.org/10.1890/110294

Odenwald, S. (2019) Smartphone sensors for citizen science applications: Radioactivity and magnetism. *Citizen Science: Theory and Practice*, 4(1), p. 18. DOI: https://doi.org/10.5334/cstp.158

Padilla, R., Netto, S.L. and **Da Silva, E.A.** (2020) July. A survey on performance metrics for object-detection algorithms. In: *2020 international conference on systems, signals and image processing (IWSSIP)*, Niterói, Brazil on 1 July 2020, pp. 237–242. DOI: https://doi.org/10.1109/IWSSIP48289.2020.9145130

Paolacci, G., Chandler, J. and **Ipeirotis, P.G.** (2010) Running experiments on amazon mechanical turk. *Judgment and Decision making*, *5*(5), pp. 411–419. DOI: https://doi.org/10.1017/S1930297500002205

Philip, S. and **Pang, A.** (2016) June. Detecting and visualizing rip current using optical flow. In: *Proceedings of the Eurographics/IEEE VGTC Conference on Visualization: Short Papers*, Goslar, DEU on 06 June 2016, pp. 19–23. DOI: http://doi.org/10.2312/eurovisshort.20161155

Plantnet. (2022) PlantNet: The Plant Identification App. Available at: https://plantnet. org (Last accessed 10 June 2022).

Qin, Z., Li, Z., Zhang, Z., Bao, Y., Yu, G., Peng, Y. and **Sun, J.,** (2019) ThunderNet: Towards real-time generic object detection on mobile devices. In: *Proceedings of the IEEE/CVF international conference on computer vision*, Seoul, Korea on 27 October 2019, pp. 6718–6727. DOI: https://doi.org/10.1109/ICCV.2019.00682

Ra, H., Richards, B.L., Rollo, A., Miller-Greene, D. and **Taylor, J.** (2022) Keeping Track of Hawaii's Bottomfish Populations With the Help of Citizen Scientists. *Fisheries*, *47*(11), pp. 510–515. DOI: https://doi.org/10.1002/fsh.10812

Reitermanova, Z. (2010) June. Data splitting. In: *WDS* (Vol. 10, pp. 31–36). Prague: Matfyzpress.

Ren, S., He, K., Girshick, R. and **Sun, J.** (2016) Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and*

machine intelligence, *39*(6), pp. 1137–1149. DOI: https://doi.org/10.1109/TPAMI.2016.2577031

Roboflow. (2023) Roboflow: Give your software the power to see objects in images and video. Available at: https://roboflow.com (Last accessed 10 June 2023).

Roche, J., Bell, L., Galvão, C., Golumbic, Y.N., Kloetzer, L., Knoben, N., Laakso, M., Lorke, J., Mannion, G., Massetti, L. and **Mauchline, A.** (2020) Citizen science, education, and learning: Challenges and opportunities. *Frontiers in Sociology*, *5*, p. 613814. DOI: https://doi.org/10.3389/fsoc.2020.613814

Sanchez, S.A., Romero, H.J. and **Morales, A.D.** (2020) May. A review: Comparison of performance metrics of pretrained models for object detection using the TensorFlow framework. In: *IOP Conference Series: Materials Science and Engineering* (Vol. 844, No. 1), p. 012024. IOP Publishing. DOI: https://doi.org/10.1088/1757-899X/844/1/012024

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and **Chen, L.C.** (2018) Mobilenetv2: Inverted residuals and linear bottlenecks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. Salt Lake City, UT on 18 June 2018, pp. 4510–4520. DOI: https://doi.org/10.1109/CVPR.2018.00474

Schwarz, J., Czarnecki, W., Luketina, J., Grabska-Barwinska, A., Teh, Y.W., Pascanu, R. and **Hadsell, R.** (2018). Progress & compress: A scalable framework for continual learning. In: *International conference on machine learning*. Stockholm, Sweden on 10 July 2018, pp. 4528–4537.

Siddique, N., Paheding, S., Elkin, C.P. and **Devabhaktuni, V.** (2021) U-net and its variants for medical image segmentation: A review of theory and applications. *IEEE Access*, *9*, pp. 82031–82057. DOI: https://doi.org/10.1109/ACCESS.2021.3086020

Simpson, R., Page, K.R. and **De Roure, D.** (2014) April. Zooniverse: observing the world's largest citizen science platform. In: *Proceedings of the 23rd international conference on world wide web*. Seoul, Korea on 07 April 2014, pp. 1049–1054. DOI: https://doi.org/10.1145/2567948.2579215

Sun, C., Zhan, W., She, J. and **Zhang, Y.** (2020) Object detection from the video taken by drone via convolutional neural networks. *Mathematical Problems in Engineering*, 2020(1), p. 4013647. DOI: https://doi.org/10.1155/2020/4013647

Tan, M. and **Le, Q.** (2019) Efficientnet: Rethinking model scaling for convolutional neural networks. In: *International conference on machine learning*. Long Beach, California on 9 June 2019, pp. 6105–6114.

Tan, M., Pang, R. and **Le, Q.V.** (2020) EfficientDet: Scalable and efficient object detection. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Virtual Event, USA on 14 June 2020, pp. 10778–10787. DOI: https://doi.org/10.1109/CVPR42600.2020.01079

Tzutalin. (2015) LabelImg — a graphical image annotation tool and label object bounding boxes in images. Available at: https://github.com/heartexlabs/labelImg (Last accessed 10 June 2023).

Ultralytics. (2023) YOLOv8: A New State-of-the-Art Computer Vision Model. Available at: https://yolov8.com/ (Last accessed 10 June 2023).

Umuhoza, E. and **Brambilla, M.** (2016) Model driven development approaches for mobile applications: A survey. In: *International Conference on Mobile Web and Information Systems*. Dublin, Ireland on 11 December 2016, pp. 93–107. DOI: https://doi.org/10.1007/978-3-319-44215-0_8

Wang, Q., Ma, Y., Zhao, K. and **Tian, Y.** (2020) A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, pp. 1–26. DOI: https://doi.org/10.1007/s40745-020-00253-5

Waste360. (2023) Covanta Survey: Americans Don't Know How to Recycle. Available at: https://www.waste360.com/waste-recycling/covanta-survey-americans-don-t-know-how-to-recycle- (Last accessed 10 June 2024).

Wildme. (2023) Wild Me – Machine Learning for Wildlife Research. Available at: https://www.wildme.org (Last accessed 10 June 2023).

Xia, X., Xu, C. and **Nan, B.** (2017) Inception-v3 for flower classification. In: *2017 2nd International Conference on Image, Vision and Computing*. Chengdu, China on 2 June 2017, pp. 783–787. DOI: https://doi.org/10.1109/ICIVC.2017.7984661

Yeh, C. and **Khan, F.H.** (2022) December. Citizen Science Mobile Apps with Machine Learning for Recyclable Objects. In: *2022 International Conference on Computational Science and Computational Intelligence (CSCI)*, Las Vegas, NV on 14–16 December 2022, pp. 1539–1542. DOI: https://doi.org/10.1109/CSCI58124.2022.00273

TO CITE THIS ARTICLE:
Khan, FH, de Silva, A, Dusek, G, Davis, J and Pang, A. 2024. SmartCS: Enabling the Creation of Machine Learning–Powered Computer Vision Mobile Apps for Citizen Science Applications without Coding. *Citizen Science: Theory and Practice*, 9(1): 14, pp. 1–16. DOI: https://doi.org/10.5334/cstp.642

Submitted: 21 May 2023 **Accepted:** 15 May 2024 **Published:** 02 July 2024

Citizen Science: Theory and Practice is a peer-reviewed open access journal published by Ubiquity Press.