UNIVERSITY OF CALIFORNIA
SANTA CRUZ

IMPROVING EFFICIENCY AND QUALITY OF DATA COLLECTION WITH MACHINE LEARNING AND CITIZEN SCIENCE

A dissertation submitted in partial satisfaction
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE AND ENGINEERING

by

Fahim Hasan Khan

September 2024

The Dissertation of Fahim Hasan Khan
is approved:

Professor Alex Pang, Chair

Professor James Davis

Dr. Gregory Dusek

Peter Biehl
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

List of Tables

Abstract

Improving Efficiency and Quality of Data Collection with Machine Learning and Citizen Science

by

Fahim Hasan Khan

Working with data is a fundamental and essential aspect of computer science, particularly in machine learning (ML), data science, AI applications, scientific analysis, and decision-making. Efficiency in data collection is crucial, as many scientific investigations, including those in computer science, rely on large volumes of data. Additionally, data quality significantly influences the overall effectiveness and performance of systems and algorithms. Citizen science facilitates public participation in scientific research, contributing to data collection, analysis, and reporting. This dissertation addresses two main challenges in the data collection process: improving efficiency and ensuring data quality. To tackle these challenges, I propose an approach that integrates ML with citizen science to enhance data collection. This synergy can improve data collection efficiency and quality, as ML algorithms assist citizen science participants in accurately identifying relevant data, filtering out label noise, and validating gathered data. Primarily, I focus on the potential of using computer vision ML models to guide and automate the collection process of visual data, such as images and videos. In this dissertation, I introduce a set of systems designed to improve the

data collection process, including SmartCS, a platform for creating ML-powered citizen science applications without writing code; RipFinder, a mobile application that uses ML to guide the collection of rip current data; and RipScout, a drone-based system for the automated collection of rip current data. These systems address data quality earlier in the collection pipeline, rather than gathering and cleaning data afterward. Another contribution of my dissertation is engaging the general public in scientific research, demonstrated through my work on involving young students in research through these systems. Overall, my approach and developed systems advance the state of the art in modern data collection processes by uniquely combining citizen science and ML, demonstrating their significance in enhancing data quality and efficiency.

To My Parents and My Family

Acknowledgments

I express my deepest gratitude to my dissertation Reading Committee, Professor Alex Pang, Professor James Davis, and Dr. Gregory Dusek, for their invaluable guidance, insightful feedback, and unwavering support throughout my research. Their time, expertise, and encouragement have been instrumental in shaping the direction and quality of this work. I also sincerely appreciate the Computer Science and Engineering Department and Graduate Division at the University of California, Santa Cruz (UCSC), for fostering a supportive and enriching academic environment.

This research was made possible by the generous funding and support from the Southeast Coastal Ocean Observing Regional Association (SECOORA) through a sub-award from NOAA (NA20NOS0120220), the US Coastal Research Program (USCRP) administered by the US Army Corps of Engineers (USACE), Department of Defense, through a Sea Grant (NA23OAR4170121), and the UCSC Center for Coastal Climate Resilience. The contents, findings, and conclusions are those of the author and do not necessarily reflect the views, positions, or policies of SECOORA, NOAA, UCSC, or the government, and no official endorsement should be inferred. I also acknowledge the support from the Google Cloud and AWS Cloud Credit for Research programs.

I am fortunate to have had the support of my colleagues Dr. Akila de Silva, Dr. Emily Lovell, Jiahao Luo, Donald Stewart, Issei Mori, Minghao Liu, Mona Zhao, Marzia Binta Nizam, and Vanshika Vats. Their collaboration, insights, and feedback played a significant role in bringing this work to fruition. I thank all my collaborators,

interns, volunteers, and participants who contributed to various projects as part of my research. I also thank the peer reviewers for providing valuable feedback on my papers. I also acknowledge the creators of the writing and editing tools, specifically Grammarly, Overleaf, and GPT-4, which I used to improve the readability and formatting of this dissertation.

Lastly, and most importantly, I wish to express my profound appreciation to my family, especially my parents and my wife, for their unwavering love, patience, and encouragement. Their belief in me has inspired my journey, and their support has been my greatest source of strength through every challenge.

The text of this dissertation includes reprints of the following previously published material:

- **Khan, Fahim Hasan**, Akila de Silva, Gregory Dusek, James Davis, and Alex Pang. "SmartCS: Enabling the Creation of Machine Learning–Powered Computer Vision Mobile Apps for Citizen Science Applications without Coding." Citizen Science: Theory and Practice 9, no. 1 (2024).

- **Khan, Fahim Hasan**, Emily Lovell, Akila de Silva, Gregory Dusek, James Davis, and Alex Pang. "WIP: Citizen Science Tools with Machine Learning as a Pathway to Engage High School Students in Research." In 2024 IEEE Frontiers in Education Conference (FIE), pp. 1-5. IEEE, 2024.

The co-author listed in these publications directed and supervised the research, which forms the basis for the dissertation.

Chapter 1

Introduction

Working with data is a basic and important aspect of computer science and engineering, modern scientific research, and technological innovation. **Data collection** is the systematic process of gathering and measuring information from different sources to obtain a thorough and precise understanding of a specific subject. It supports critical processes in scientific analysis, decision-making, and various other modern scientific endeavors, such as building machine learning (ML) models and artificial intelligence (AI) applications. Efficient and high-quality data collection is essential for the advancement of these fields. However, it presents various challenges that require innovative solutions.

Efficiency in data collection refers to the ability to gather accurate and reliable data swiftly and with minimal resource expenditure. It involves optimizing processes to reduce time, costs, and effort while maintaining high-quality results. **Data quality**

relates to the accuracy, reliability, and relevance of data in fulfilling its intended purpose. High-quality data is complete, consistent, and error-free, facilitating effective decision-making and analysis.

Efficient collection of high-quality data is challenging for various reasons, such as the complexities of handling large volumes of different types of data from diverse sources, ensuring data integrity, and managing real-time data streams. Maintaining accuracy and completeness of data while minimizing latency in real-time applications requires sophisticated technology and systems. Resource constraints, including cost and the need for skilled personnel, increase these challenges. Additionally, ensuring user engagement, managing device and computational limitations, and addressing legal, ethical, and environmental considerations add further complexity to the data collection process. All these factors combined make collecting high-quality data efficiently complex and challenging.

The motivation for my work is driven by the need to use new technologies to solve the challenges of collecting data, which is a crucial part of modern science and innovation. As the need for high-quality data grows, especially in fields like ML and AI, it has become very important to make data collection methods more efficient and accurate. My dissertation focuses on using citizen science and ML to make data collection easier and more accessible for everyone, not just experts. By creating new tools and methods, this work aims to get more people involved in scientific research, make the data collected more reliable, and apply these improvements to real-world

issues like detecting dangerous rip currents, thereby contributing to scientific progress.

To address the data collection challenges, a primary focus of this work is the development and implementation of SmartCS, a novel platform designed to enable the creation of ML-powered computer vision mobile apps for citizen science applications without requiring any coding or programming skills. SmartCS leverages the computational capabilities of modern mobile devices to perform complex computer vision tasks using ML to guide non-expert participants in collecting high-quality visual data in various research fields. This platform addresses the challenges faced by citizen scientists, including the need for accurate data labeling and the limitations of server-dependent ML systems in remote locations.

Additionally, this dissertation investigates the role of ML in engaging the general public in research activities through the development of citizen science tools. We selected a group of high school students as a representative sample of the general public. The high school students were chosen for this study due to their availability, eagerness to learn, and status as good representatives of lay users, given their early stage of learning. Moreover, by integrating ML within these tools, students gain hands-on experience with advanced technologies, fostering their interest in Science, Technology, Engineering, and Mathematics (STEM) careers and enhancing their understanding of scientific methods. This approach benefits the students and contributes valuable data to ongoing research projects.

While data collection is essential for many applications in various fields, this

dissertation focuses on a specific application to implement and evaluate our methods. It concentrates on the application of ML for rip current detection and automated data collection, an area of critical importance due to the significant dangers these currents pose to beachgoers worldwide. Rip currents are dangerous, strong, and fast-moving currents that can pull even experienced swimmers away from the shore, often leading to drownings and fatalities. Identifying rip currents from an ML and computer vision perspective involves object detection and segmentation. Developing an effective rip current detection application necessitates real-time, often on-device, processing. Additionally, rip currents are challenging to detect due to their amorphous and transient nature, making them difficult for even advanced computer vision algorithms to identify. Because of these challenges and design considerations, the proposed solutions in this dissertation can be easily translated and applied to other data collection problems across various domains.

Among the proposed approaches, the development of the mobile application RipFinder, an expansion of a SmartCS app, illustrates the practical use of ML for real-time, client-side detection of rip currents, providing a tool that operates effectively even without internet connectivity. Additionally, this dissertation introduces RipScout, a drone-based system designed for the automated collection of rip current data. Furthermore, it presents the design and deployment of a WiFi camera-based system for automated rip current data collection, demonstrating the usefulness and effectiveness of ML in enhancing environmental monitoring and safety.

Given the extensive challenges associated with data collection and the scarcity of focused research on improving its efficiency and quality, we defined two key research questions (RQs) based on the literature review (Chapter 2).

- **RQ 1:** How can efficiency in large-scale data collection be improved through the integration of advanced technologies and methodologies?
- **RQ 2:** How can we ensure and enhance the quality and integrity of data throughout the collection, processing, and utilization stages?

The contributions of this dissertation are summarized below:

- SmartCS, A platform for creating ML-powered citizen science applications that enables users to develop applications without writing code.
- Investigating strategies to engage high school students (as a representative group for the general populace) in research through the use of SmartCS and other tools to create ML-enhanced citizen science apps.
- RipFinder, A mobile application that utilizes ML to guide the collection of rip current data.
- RipScout, A drone-based system designed for the automated collection of rip current data.
- The design and implementation of a wireless network camera-based system for automated data collection.

In this dissertation, Chapter 2 discusses previous works and challenges on data collection. Chapter 3 presents the overview of approach presented in this dissertation. Chapter 4 introduces the SmartCS platform, with details of its design and implementation for creating ML-integrated mobile apps for citizen science and some user studies. Chapter 5 investigates the use of ML in citizen science tools to engage high school students with case studies and user feedback. Chapter 6 focuses on RipFinder, a mobile app for real-time rip current detection. Chapter 7 covers RipScout, a real-time rip current detection and automated data collection system using drones or Uncrewed Aircraft Systems (UAS). Chapter 8 presents the system design for real-time rip current detection and automated data collection using a network camera in a remote location. Finally, Chapter 9 provides concluding remarks and outlines planned future work.

Through these chapters, the dissertation aims to showcase the potential of combining citizen science, mobile technology, and ML to enhance public participation in scientific research, improve data collection processes, and provide educational opportunities.

Chapter 2

Related Work

Data collection is fundamental to scientific research and decision-making processes across various fields. Systematic data collection is essential for both qualitative and quantitative research [267]. Analyzing these data allows researchers and decision-makers to obtain insights, make predictions, and develop new technologies. High-quality data collection ensures the reliability and validity of research findings, facilitating advancements in science and technology [265]. Selecting appropriate data collection methods and strategies is necessary to obtain relevant and accurate data for specific research purposes [15, 121, 194, 233]. The careful approach to data collection enables effective scientific research and technical development.

In computer science and engineering, data collection is a central focus. For example, ML models depend heavily on large datasets to generate accurate predictions and identify patterns. The quality and amount of data significantly impact the

performance and reliability of these models. Efficient and high-quality data collection is crucial for the progression of AI and other data-driven technologies. The success of ML algorithms is primarily related to the availability of large and well-curated diverse datasets, which provide the necessary breadth and depth for training robust models [123]. Furthermore, as highlighted in numerous studies, high-quality data collection practices improve model performance and reduce biases [29, 93, 265]. By focusing on developing better data collection methodologies, the field can continue to address complex problems with innovative solutions with greater accuracy and efficiency, further strengthening the critical role of data in technological advancement.

In this chapter, I provide a high-level and broad discussion of previous works and the framing of data collection challenges, followed by more detailed and relevant background work in the subsequent chapters.

2.1 Previous Works

As a foundational task in scientific research, data collection has long presented significant challenges [267]. Over time, various strategies such as crowdsourcing [121], automation, and mixed methods [15, 29] have been implemented to enhance the data collection process. These challenges and strategies differ across research domains and evolve with technological advancements. For example, the challenges and solutions in medical research [165] may differ from those in environmental science [190].

Technological progress has continuously improved data collection methods [143],

enhancing both quality and efficiency, particularly with the advent of computer science and digital tools [233] and ML. While ML has revolutionized the field, it also introduces challenges in ensuring data quality and efficiency [211]. High-quality data is crucial for training deep learning models, yet maintaining this quality remains a major obstacle [269].

Data can be collected through various methodologies, including surveys, interviews, observations, experiments, sensors, digital tools, and crowdsourcing, tailored to meet specific research contexts and objectives [51, 192]. Recent advancements in mobile and wearable technology, such as smartphones and fitness trackers, have made real-time data collection possible, benefiting studies in healthcare, environmental monitoring, and user behavior [191]. IoT devices and remote sensing have further transformed fields like environmental science, allowing automated, continuous data gathering for better analysis and decision-making [96].

For example, in agriculture, IoT devices and smart sensors are used for monitoring soil moisture, temperature, air quality, and crop health, optimizing irrigation and fertilization [251]. These types of applications also improve productivity and sustainability in various other fields [259]. Systems developed using these technologies allow for comprehensive and precise data collection, facilitating better analysis and decision-making for environmental management and conservation efforts [101].

Social research has also advanced with online surveys and social media platforms that gather data on public opinion and behavior [25]. Platforms like YouTube [88],

Instagram [108], and TikTok [226] provide rich visual data that can be used to study cultural trends, communication patterns, and user engagement [103]. Collaborative efforts like Wikipedia, OpenStreetMap, Google Street View, etc. rely on crowdsourcing to enhance data accuracy [102]. Citizen science efforts make use of crowd participation to gather large volumes of data from various locations [92]. The subsequent chapters (Chapters 3 to 5) present a more comprehensive discussion of citizen science's role in data collection.

In technology development, data is essential for training ML models in computer vision, natural language processing, and autonomous systems. Platforms like Amazon Mechanical Turk and Google Colab make data gathering and labeling more accessible [20, 30]. On the other hand, AI and ML also help automate data categorization, error detection, and quality control, streamlining research processes and reducing errors [185].

For instance, in healthcare, AI tools analyze patient data from medical records, wearables, and imaging, enabling early diagnosis and personalized treatment plans [247]. AI can also automate tasks such as transcribing audio recordings, analyzing social media content, and categorizing images, thereby streamlining workflows and reducing the time required for data processing [71]

2.1.1 Visual Data Collection

This dissertation centers on the collection of visual data, such as images and videos, often supplemented with metadata like location and elevation [92]. Visual data is added with observations and annotations to create comprehensive datasets that are widely used in areas like biodiversity monitoring, autonomous driving, urban planning, and coastal engineering. For example, in rip current studies, images, and videos are crucial for detection and analysis [35, 53]. Metadata integration, such as GPS coordinates, further enhances the contextual understanding of visual data for more accurate analysis [68].

The evolution of visual data collection began with early methods like remote sensing, aerial photography, and video, which laid the foundation for today's advanced systems. Starting in the mid-20th century with technologies like Landsat 1 in 1972, continuous Earth monitoring became possible, driving progress in land use and environmental analysis [249]. Aerial photography, initially used in military reconnaissance, later found applications in environmental and urban studies [48]. Early applications in environmental monitoring include Cowardin et al.'s [49] wetland classification using aerial photography. Integrating visual data into GIS in the 1980s marked a major leap in spatial analysis, as highlighted by pioneers like Tomlinson [245]. Lillesand and Kiefer's work was a very useful resource on the use of visual data for environmental monitoring [162]. However, one of the major limitations of these early visual data collection systems was the need for more well-thought strategies to ensure efficiency and quality in the data collection process.

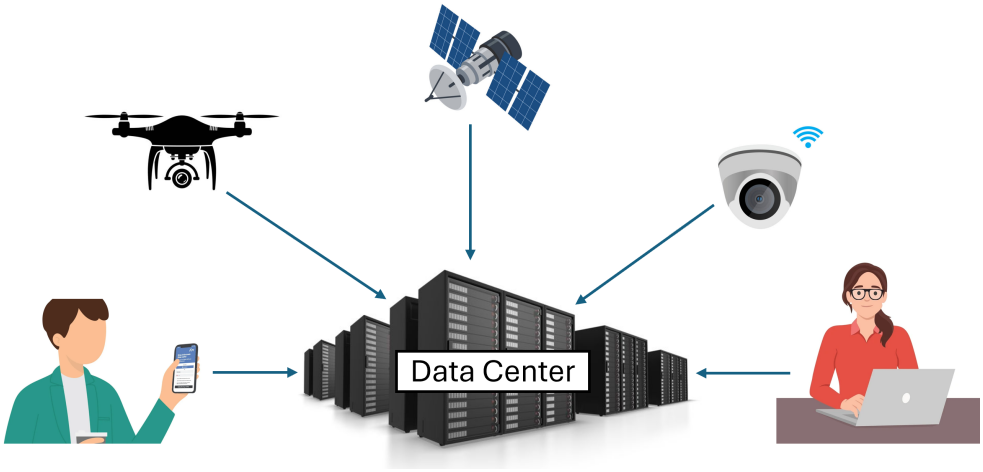


Figure 2.1: Diverse data sources collect large amounts of data stored in data centers.

Recent advancements in high-resolution cameras, sensors, and computer vision have transformed visual data collection. Drones, satellites, and other devices now capture detailed imagery across vast areas, supporting diverse applications from environmental monitoring to urban planning [181]. In healthcare, advances in medical imaging, combined with AI and ML, enhance disease detection and patient monitoring [165]. Similarly, autonomous systems depend heavily on visual data from cameras, LiDAR, and radar for safe navigation and decision-making [159].

These technological advancements have greatly improved the accuracy and efficiency of data collection. However, there remains a gap in research focused on simultaneously improving both accuracy and efficiency in data collection. This dissertation aims to develop innovative solutions and systems that contribute to closing this gap.

2.1.2 Efficiency in Data Collection

Effective data collection is essential for handling the large volumes of data gathered and produced in today's digital age. (Figure 2.1). As organizations and researchers increasingly rely on large datasets to derive meaningful insights, improving the efficiency of data collection becomes even more critical. While there are various challenges related to efficiency in data collection, a few challenges related to this dissertation, identified through the literature review, are briefly discussed in the non-exhaustive list below. These points are also relevant to Research Question 1 presented in Chapter 1.

1. **Data Variety and Complexity:** The data collected comes in various formats and complexities, making it difficult to manage and process efficiently. The diversity of data types, from structured data in databases to unstructured data such as text, images, and video, requires sophisticated methods to integrate and analyze effectively [143]. A study by Katal et al. provides an overview of tools and methodologies for handling complex and varied data in large-scale collections [128].

2. **Resource and Technological Constraints:** Limited resources and outdated technologies can obstruct data collection. The vast amount of data collected rapidly and continuously requires scalable and modern technological infrastructures to manage effectively [150]. Hilbert et al. highlight the constraints and capacities of current technologies in managing large-scale data [104].

3. **Real-Time Data Collection:** Collecting data in real time requires robust systems capable of handling continuous data streams. Real-time data analytics needs systems that can process and analyze data on the fly, providing immediate insights and responses [83]. A survey by Yasumoto et al. presents a system for real-time stream processing, addressing the challenges of IoT data collection [278].

4. **User Participation:** Engaging users effectively to contribute data can be challenging. Incentivizing and maintaining user engagement in citizen science and crowdsourcing projects are critical for successful data collection efforts [270]. Kittur et al. discuss methods to engage and motivate users in data collection tasks [144].

5. **Legal and Regulatory Compliance:** Ensuring compliance with data protection laws and regulations adds another layer of complexity. Understanding and following laws and legal frameworks to ensure ethical data practices and protect user privacy is crucial [282]. A study by Tene et al. discusses the regulatory landscape and compliance strategies for large-scale data collection [241].

2.1.3 Data Quality

Maintaining data quality and integrity is essential for the reliability of research results and technical applications. The importance of data quality is well-documented in the literature, emphasizing the need for accurate, consistent, reliable, complete, and usable data. Studies by Wang and Strong highlight that data quality is multi-

dimensional and that these key aspects are fundamental for effective decision-making and operational efficiency [234, 265]. Other studies extensively discussed the critical nature of these dimensions in ensuring data quality across various contexts and applications [195, 202]. Some aspects of data quality relevant to this dissertation, particularly in relation to Research Question 2 presented in Chapter 1, are briefly discussed below.

1. **Accuracy:** Data should accurately represent the intended information without errors or noises. Ensuring accuracy is vital, as inaccurate data can lead to incorrect conclusions and faulty decision-making [265]. For example, Chen et al. discuss methods for improving data accuracy through enhanced data analytics and intelligence systems [38].

2. **Consistency:** Data should be standard across different systems and formats, ensuring reliable access and analysis. Consistent data facilitates smooth integration and comparison [202]. Madnick et al. discuss the role of data semantics in ensuring data consistency across different systems [172].

3. **Reliability:** Reliable data can be depended upon for making sound decisions and conducting operations. Reliability is essential for trust in data, as noted by Strong, Lee, and Wang [234]. Research by Fisher et al. highlights the importance of reliable data in critical decision-making processes [79].

4. **Completeness:** Data should be complete and free of missing information that

could affect its usefulness. Incomplete data can lead to biased analyses and compromised results [195]. For example, the DaQuinCIS Project focuses on methods to ensure data completeness in information systems [217].

5. **Usability:** Data should be easy to understand, process, and use by its target audience. Usability ensures that data can be effectively utilized for its intended purpose, enhancing the overall value of the data [195]. Eppler et al. discuss various approaches to enhancing data usability for different audiences [70].

Maintaining the quality of data presents significant challenges due to several factors: the large volume of data generated and collected [150]; the diversity of data sources [143]; human and system errors [202]; and data degradation over time [195]. First, the vast amount of gathered data can be overwhelming, making the efforts to maintain consistent quality across datasets complicated [150]. Large volumes can lead to data quality issues due to several factors. For example, the increased complexity and variety of data sources make it challenging to standardize and validate data effectively [84, 143]. Additionally, as the volume of data grows, the likelihood of encountering incomplete, duplicate, or erroneous data increases, making it harder to ensure accuracy and consistency [41]. This can eventually reduce the overall data quality and the reliability of any analyses performed on it [202]. Data degradation is a critical issue. As data ages, it may become outdated or irrelevant, thereby decreasing its utility [195,265]. This trend is particularly relevant for time-sensitive data sources like financial market data, weather information, and social media trends. For example, weather data becomes

obsolete quickly as new data updates are required for accurate forecasting [89]. Social media data, used to understand public sentiment or trends, can also become irrelevant quickly as public opinion changes fast [22, 250]. Finally, human and system errors during data entry, collection, or processing can introduce inaccuracies, negatively affecting data integrity and reliability. Overcoming these challenges is essential to maintain data as a reliable basis for decision-making and analysis in scientific research.

2.2 Comparison of My Work with Prior Research

My research is different from previous related works because it focuses on improving both the efficiency and quality of data collection simultaneously. Earlier studies have generally concentrated on either advancing the technology for more efficient data gathering or developing better methodologies to enhance data quality. For example, as discussed earlier in this chapter, many research efforts have been aimed at creating advanced sensors and automated systems that make the data collection process faster and more efficient. Others have focused on engaging communities and creating protocols to gather reliable data from non-experts. However, these approaches often considered efficiency and quality as separate goals, addressing them independently rather than as interconnected factors that can be optimized together.

In contrast, my work takes a more comprehensive approach by using the tools and systems I developed to improve both efficiency and quality at the same time. This dual focus not only speeds up the data collection process but also ensures high-quality data

through real-time validation and visual feedback mechanisms. The RipScout system is a key example of this approach, integrating advanced ML models with practical tools like drones. By considering efficiency and quality together, my research offers a novel perspective and methodology, closing the gap left by previous works.

Chapter 3

Approach

In this dissertation, my approach uniquely integrates ML with citizen science to create a scalable, cost-effective, and efficient data collection system. Unlike traditional methods, this approach leverages the power of citizen science to involve a vast network of volunteers, significantly increasing the scale of data collection without the need for specialized training. Additionally, the SmartCS tool enables the creation of ML-powered mobile apps without coding, reducing development costs and making advanced data collection tools accessible to a broader audience [138]. By automating data collection using ML-guided UAVs and an extensive network of cameras, my approach reduces the need for human intervention, thus lowering operational costs. The use of ML algorithms ensures higher data quality by filtering out noise and validating submissions, improving the precision and dependability of the gathered data. Furthermore, this approach fosters public engagement and education in scientific

research, adding a valuable dimension to the scientific and technical community [23]. Overall, my approach offers a novel alternative to traditional data collection methods by being more effective in ensuring high-quality data.

3.1 ML and Data Collection

ML offers powerful tools for improving the efficiency and quality of data collection. ML algorithms can automate the identification, categorization, extraction, and prediction of valuable information from large datasets [146]. This automation enhances efficiency and accuracy in data collection processes, particularly in tasks involving complex pattern recognition and classification [154]. For example, convolutional neural networks (CNNs) have shown remarkable success in image recognition and classification tasks, significantly improving data annotation processes [147].

The relationship between ML models and data is inherently circular and symbiotic, with each component continuously influencing and improving the other (Figure 3.1). This relationship is characterized by a continuous feedback loop where high-quality, diverse data are essential for training effective ML models, and in turn, these models facilitate more efficient and accurate data collection and annotation [62, 123]. As ML models learn from the data, they generate insights and predictions that highlight areas needing improvement, leading to the refinement of data quality and quantity [98]. This iterative process involves deploying models to interact with real-world data, which generates new data that further trains and refines the models [220]. Consequently, as

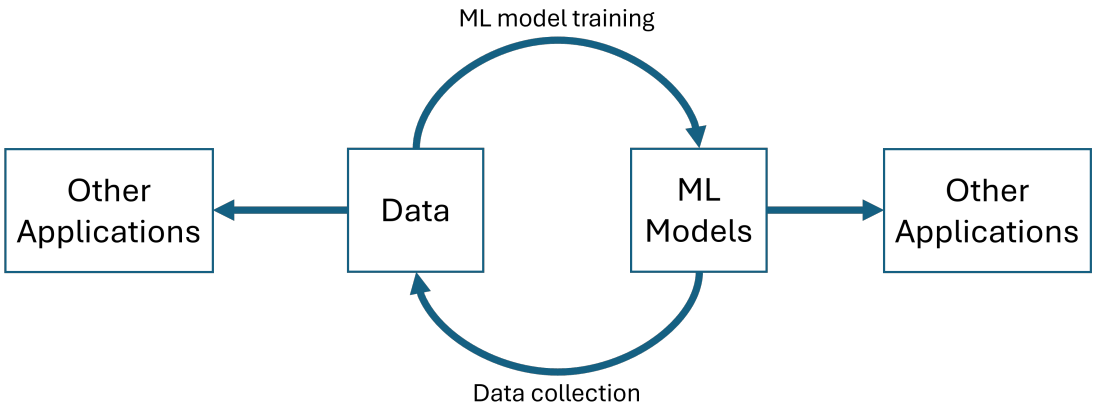


Figure 3.1: The circular relationship between data and ML models is illustrated here. Some examples of "Other Applications" include person identification, plant and animal detection and counting, autonomous driving, image recognition, and segmentation.

models evolve and improve, they enable more advanced data collection and analysis techniques, creating a cycle of mutual enhancement and progressive advancement in ML [145]. Even though this concept was suggested in many of the previous works referenced above, the novel aspect of my work is leveraging this circular relationship between data and ML models in the systems I designed and implemented to improve the data collection process, as explained in the following chapters.

A notable real-life example of this circular relationship is the Segment Anything Model (SAM) developed by Meta AI [141]. SAM was trained on a dataset comprising 11 million images and 1.1 billion annotation masks, which were initially generated using various ML models. A small sample of this dataset was then verified by human experts to ensure accuracy. SAM's primary application is to generate segmentation masks from any image, facilitating tasks such as automated and ML-guided annotation of data. Consequently, SAM is not only utilizing data for its training but also generating

new data that can be used to train additional ML models, perpetuating the cycle of mutual enhancement between models and data.

Data Labeling: Data labeling is necessary for a range of applications, such as computer vision, NLP (natural language processing), and speech recognition. For example, when building a computer vision system, we first need to label images, pixels, or key points or create a bounding box around a digital image to generate our training dataset. Images can be classified by type (like product vs. lifestyle images), content, or segmented at the pixel level. This labeled data can then be used to develop a computer vision model capable of automatically classifying images, detecting object locations, identifying key points in an image, or segmenting an image [9]. In ML, data labeling involves tagging raw data (such as images, text files, videos, etc.) with one or more descriptive labels to provide context that a ML model can learn from. For instance, labels may specify whether a photo contains a bird or car, what words were spoken in an audio recording, or if an x-ray shows signs of a tumor [9].

The quality of annotations is hard to control, and obtaining reliable labels from citizen science platforms can be difficult due to label noise. Incorrect or mislabeled data introduce label noise. Frénay et al. [81] provides a detailed survey on various types of label noise, identifying sources such as insufficient information, expert mistakes, subjective classification, and encoding or communication problems. Although "noise" has different meanings in various branches of science, in this text, "label noise" refers to observed labels that are classified incorrectly [81].

Another goals of this work is to improve data quality by reducing label noise through the implementation of data collection systems with ML guidance and automation. The ML guidance assists human data collectors in recognizing the correct data and providing accurate labels, thereby reducing label noise and enhancing overall data quality.

3.2 Citizen Science

Citizen science, a special form of crowdsourcing, involves the participation of volunteers, often non-experts, in scientific research. These volunteers collect and/or analyze data, contributing to a wide range of projects. Citizen science benefits both researchers and participants. Researchers can collect data that they otherwise would not be able to, while participants learn about the subject they are engaged with. Most citizen science projects requires data collection. Considering the diverse range of projects, providing some level of expertise or guidance for beginners can significantly enhance the quality of the gathered data.

Many citizen science projects rely heavily on visual data, like photographs or videos of various subjects. These data are often collected from all over the world, including remote locations. For instance, with iNaturalist, a smartphone app available to everyone, users can gather data and learn about various plant and animal species [257].

Citizen science platforms are increasingly going mobile with emerging technologies

and shifting paradigms [180]. Modern smartphones, equipped with multiple cameras and an array of sensors, can be used in combination to provide more information than just photos and videos. Utilizing these additional capabilities of smartphones can enhance data collection in citizen science projects. Additionally, the increasing availability of low-cost UAS makes aerial imagery from public contributions a possibility for researchers. Citizen science participation usually involves data collection, annotation, classification, and other tasks. An example of a data collection task is volunteers using smartphones to take photos of plant species in their local area for a biodiversity project, then uploading the images to a central database for further analysis. An annotation or labeling task involves identifying and labeling these species in the images, often adding additional information or comments. Classification entails categorizing the species to assist biologists in their studies. Another example task is volunteers participating in a coastal cleanup event, recording the types and quantities of trash collected, which are then analyzed to understand pollution patterns and develop strategies for reducing marine debris. There is an opportunity to engage participants more intimately by improving modeling or prediction through confirmation or refutation.

3.2.1 Citizen Science for Data Collection

Data collection in citizen science projects has various objectives, such as monitoring or performing analysis. In many cases, ML algorithms are used for post-collection

analysis of the data. For example, in the CoastSnap project, beachgoers take photos of the coastline from fixed camera mounts and upload them to a central database [100]. After the data collection phase, ML algorithms are used to analyze these images. The computer vision models can automatically detect and track changes in shoreline position, identify features such as sand dunes and vegetation, and measure beach width and slope over time. This post-collection analysis allows researchers to efficiently process large volumes of visual data, monitor coastal erosion and accretion, and make informed decisions about coastal management and preservation strategies.

As shown in Figure 3.1, data can also be used to further train the models used in the citizen science platform to assist participants. However, traditional neural network architectures struggle with catastrophic forgetting, making it challenging for them to learn a series of tasks consecutively. Catastrophic forgetting is the phenomenon where a neural network, when trained sequentially on multiple tasks, tends to forget the knowledge learned from previous tasks as it learns new ones [82]. This happens because the model parameters are updated to optimize the performance on the current task, often at the expense of the performance on previous tasks. In summary, the network overwrites its existing knowledge with new information, leading to a significant drop in accuracy on earlier tasks. This challenge is particularly problematic in scenarios where continuous learning from an evolving data stream is required, such as in lifelong learning applications or when deploying models in dynamic environments [142]. Since new data are continuously collected in citizen science projects, using these data

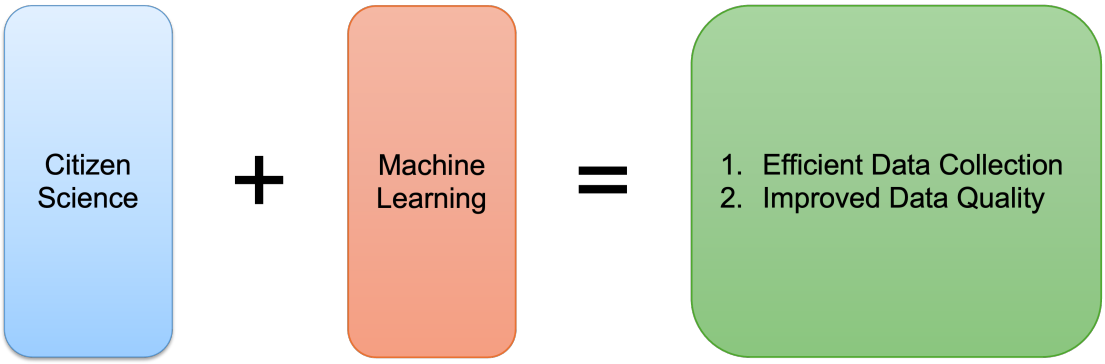


Figure 3.2: This dissertation hypothesizes that integrating ML with citizen science enhances data quality by guiding volunteers, reducing label noise and errors, and validating submitted data, with the systems described in later chapters designed to test this hypothesis.

efficiently for ML training without forgetting previous training can be challenging [56].

3.3 Synergy of ML and Citizen Science

Combining ML with citizen science creates a powerful synergy for enhancing data collection. ML algorithms can process and analyze the large volumes of data generated by citizen scientists, identifying patterns and insights more quickly and accurately than human analysts. The hypothesis of this dissertation is that integrating ML with citizen science enhances data quality by providing guidance to volunteers, minimizing label noise and errors, and validating the data they submit (Figure 3.2). The systems and studies described in the subsequent chapters are designed to prove this hypothesis.

Furthermore, ML can enhance engagement and participation in citizen science projects by providing tools for data visualization, real-time feedback, and personalized experiences for participants. Conversely, the diverse and extensive data collected

Figure 3.3: Visual data collection platforms used in this dissertation: (1) Stationary (left), and (2) Mobile (middle and right).

through citizen science initiatives are invaluable for training and refining ML models, making them more robust and accurate.

Efficient data collection with ML and citizen science represents a modern approach to scientific research and data gathering. It leverages cutting-edge technology and widespread public engagement to tackle complex challenges in data collection, offering scalable, cost-effective, and inclusive methods for gathering high-quality data. This approach not only advances scientific research but also democratizes the process, involving the public in meaningful scientific endeavors.

3.4 Visual Data Collection Platforms

The main focus of this dissertation is visual data (images, videos, etc.), which is the primary type of data collected by the systems I have proposed and implemented based on the concepts and hypotheses discussed in the previous sections. Typically, this type of data is collected using various types of cameras mounted on different platforms, such as smartphones, drones, satellites, microscopes, and standalone handheld cameras.

From the perspective of my research (Figure 3.3), visual data collection platforms can be categorized into two types: (1) Stationary or fixed camera platforms (Chapter 8), and (2) Mobile camera platforms (Chapter 4, 6, and 7).

3.4.1 Stationary or Fixed Camera Platforms

Stationary or fixed visual data collection platforms can be deployed for various applications. For instance, surveillance cameras are extensively used for security and monitoring in public spaces, businesses, and residential areas [268]. Traffic cameras are installed at intersections and along roadways to monitor traffic flow, capture violations, and enhance safety. Webcams, typically linked to computers or networks, are used for activities like live streaming, video calls, and remote surveillance [263]. Wildlife cameras are strategically placed in natural habitats to observe and record wildlife behavior and environmental changes [32]. Environmental monitoring stations use fixed cameras to track weather conditions and pollution levels [151].

These fixed camera platforms share several common features. They are typically deployed in fixed locations and are often part of remote or networked systems that allow for continuous or periodic monitoring. The deployment of these cameras requires careful planning to ensure they are positioned to effectively capture the desired data, whether for security, traffic management, wildlife observation, environmental monitoring, scientific research, or industrial processes. The effectiveness of these systems depends on strategic placement and proper maintenance to ensure reliable data

collection over time.

3.4.2 Mobile Camera Platforms

Various mobile visual data collection platforms are utilized today, including handheld cameras, smartphones, tablets, drones (UAS), body-worn cameras, wearable devices like smart glasses and AR headsets, vehicle-mounted cameras, and robots. Specifically, my research concentrates on two main types of mobile cameras: (1) cameras integrated into smart devices (Chapters 4 and 6), and (2) cameras mounted on UAS (Chapter 7). Although the systems presented in Chapters 6, 7, and 8 were developed for rip current data collection, they can be easily adapted for a wide variety of use cases. We selected rip current data collection as a pilot project to demonstrate the application and capabilities of our approach and systems.

3.4.2.1 Smartphones and Smart Devices

Smartphones and smart devices have become essential tools for visual data collection due to their widespread availability, portability, and advanced technological capabilities. Equipped with high-resolution cameras, these devices enable users to capture high-quality images and videos easily. The integration of additional sensors, such as high-precision location/GPS sensors, accelerometers, gyroscopes, 3D/depth cameras, and LiDAR, enhances the contextual data that can be collected alongside visual information, making these devices highly versatile for a range of applications.

In research and citizen science projects, smartphones and smart devices allow users to gather and share data in real-time, facilitating large-scale data collection efforts across diverse geographical locations [18, 111, 183]. Furthermore, the expansion of user-friendly applications such as CoastSnap [100], iNaturalist [183], and Pl@ntNet [196] enables non-experts to participate in data collection, thereby democratizing access to scientific research and fostering community engagement. With their ability to support advanced functionalities, such as ML and augmented reality, smartphones and smart devices are not only revolutionizing visual data collection but also expanding the possibilities for analysis and interpretation of the collected data [33, 77, 257].

A common feature of smartphones and smart devices is their mobility and flexibility, allowing data collection in various settings and conditions. However, they often face constraints related to computational power and battery life, which can limit the complexity of real-time data processing and analysis performed on the devices themselves.

3.4.2.2 Uncrewed Aerial Systems (UAS)

Recent developments in UAS have made these platforms valuable for data collection in inspection, surveillance, mapping, and 3D modeling [181]. They offer a low-cost alternative to manned aerial photogrammetry, particularly in short- and close-range domains. Rotary or fixed-wing UAS are much cheaper and easier to operate, even by amateur pilots [260]. Most modern UAS can fly in manual, semi-automated, and

autonomous modes, and can be equipped with additional sensors beyond high-quality cameras [61].

UAS can perform photogrammetric data acquisition with amateur or SLR digital cameras, producing outputs like terrain models, elevation models, contour lines, textured 3D models, and vector information for large areas [181]. Their accessibility makes UAS ideal for a variety of citizen science projects [240, 273]. While some UAS have basic object detection and obstacle avoidance features, adding customized ML-assisted data collection capabilities can further extend their applications in specialized projects [19, 58, 182, 244].

A key feature of UAS is their maneuverability and ability to access hard-to-reach areas and perform data collection over large regions. However, like smartphones and smart devices, they face constraints related to computational power and battery life, limiting flight duration and the complexity of real-time data processing onboard.

3.5 Translating Approach to Systems

The subsequent chapters detail the application of this approach to various systems. The next chapter introduces the SmartCS platform, which enables the development of a citizen science app with integrated ML to enhance the efficiency and quality of data collection. Following this, I assess the effectiveness of my approach by addressing four research questions. The chapter on the RipFinder app advances this concept by employing sophisticated techniques to combine ML and citizen science for efficient,

high-quality rip current data collection. The RipScout system builds upon RipFinder by incorporating drones guided by ML to further improve the quality and efficiency of rip current data collection. Finally, I present the implementation and development of a versatile system designed to efficiently collect various types of high-quality data, extending beyond rip current data.

Chapter 4

SmartCS: Enabling the Creation of ML-Powered Computer Vision Mobile Apps for Citizen Science Applications without Coding

In this chapter, I introduce SmartCS, a platform that enables the creation of citizen science apps with ML support quickly and without the need for coding skills [138]. 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 subjects, thereby enabling high-quality data collection and accurate annotation, which reduces label noise. Several citizen science apps created

using SmartCS are also presented, some of which were conceived and developed by high school students.

4.1 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 [14, 97, 261]. 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 [67]. 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 [257].

With emerging technologies and shifting paradigms, citizen science platforms are also becoming mobile, making it easier for participants to collect visual data [180]. Mobile devices, such as smartphones and tablets, equipped with multiple cameras and advanced processing capabilities, can perform complex computer vision (CV) tasks using machine ML [197]. 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 non-experts. Mobile apps with integrated 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 [209].

Citizen science platforms such as Zooniverse [228], SPOTTERON [111], Anecdata [59], etc., provide the service to build citizen science apps for crowdsourced research projects [166]. 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 [228]. 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 [257].

This chapter 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 [158]. Furthermore, the investment required for app design and development often spans from tens to hundreds of thousands of US dollars [184]. The cost of crafting apps, especially those with an extensive range of features can hinder innovation [107]. 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 validated 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.

4.2 Motivation

Here, we discuss the challenges participants face in research data collection for rip current detection [193]. 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 [35, 86]. 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 [26]. However, various ML methods can detect rip currents, as demonstrated by recent works [53, 54, 174]. Mobile apps empowered by ML can assist non-expert 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 non-expert 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.

4.3 Related Works

4.3.1 Citizen Science Platforms

We examined several popular citizen science app creation platforms enabling people-powered mobile app development [166]. Earlier, we mentioned Zooniverse, which features projects from diverse domains [18,228]. One of the Zooniverse projects is OceanEYEs [198], which seeks volunteers to count and label fish, if there are any, in millions of unlabeled images collected from the ocean. Anecdata [60] is another

free platform like Zooniverse, where both are primarily web portals with companion mobile apps. SPOTTERON [166] 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 [46] 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 [47, 52, 69, 115].

4.3.2 Citizen Science Apps with ML

Many custom-built citizen science applications have ML capabilities. We previously mentioned that iNaturalist [257] has server-side ML capabilities and functions like a social network to connect nature observers. Leafsnap [149], 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 took 5.4 seconds per image, which is not fast enough to produce real-time results [149]. 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 [271], Fathomnet [77, 129], 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. [136] 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 [90, 196]. 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 [257].

4.3.3 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 [122]. Few customizable mobile app creation and deployment platforms, such as App Movement [85], provide generic templates for the community to build apps. Model-driven development (MDD) [17] has been adopted for mobile app development to simplify the process, reducing

technical complexity and costs significantly. For example, [87] 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. [160] 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 [112, 255]. 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 [45], Lobe.ai [168], Ultralytics HUB [254], etc., have basic app templates that can be used with the trained models. TensorFlow Lite [1] 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 [11, 31, 275, 276]. 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.

4.3.4 ML 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. Due 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 [136].

Modern vision tasks, such as object detection, image classification, semantic segmentation, etc., are mostly done using convolutional neural networks (CNN). As the previously discussed categorization is about all ML models in general, specifically for object detection, 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 [205], have a region-proposal stage and a classifier stage. The limited computational power of today's mobile devices precludes their direct use. On the other hand, they have been utilized for real-time object detection via remote GPU servers [155]. However, these server-dependent systems are only suitable for deployment in locations with network access. On the contrary, SSD attains object detection using a single-stage CNN [155]. YOLO [118], EfficientDet [239], and SSDs, such as SSD MobileNet [44, 216], are designed to perform in realtime sacrificing some accuracy [109]. Additionally, Sun et al. [235] 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 [254], EfficientNet [237], Inception [277], and MobileNet [44] are optimized for real-time image classification on mobile devices. Likewise, YOLOv8-seg (Ultralytics 2023), U-Net MobileNetv2 [227], MobileNetv2-DeepLab-v3 [40], 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.

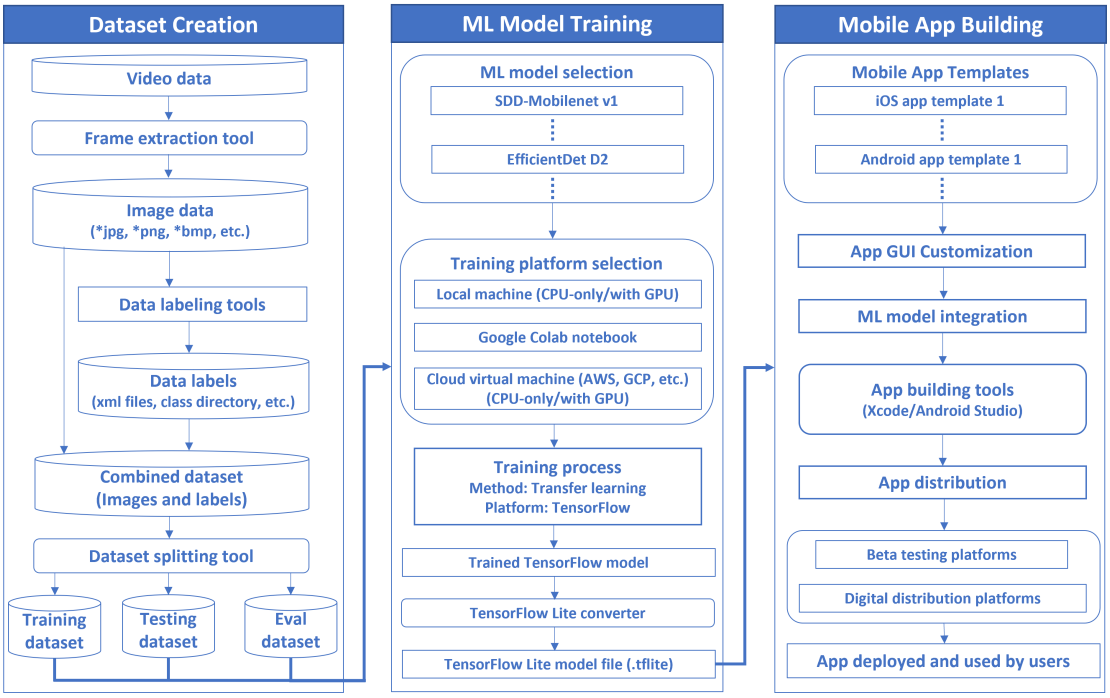


Figure 4.1: Overview and workflow of the components of our open-source citizen science app creation platform.

4.4 System Design of the Platform

Our citizen science app creation platform combines multiple technical components to build the overall system. The main parts of our proposed system are three steps: (1) Training dataset creation, (2) ML model training, and (3) Mobile app (iOS or Android) building. The workflow of our system is presented in Figure 4.1. Furthermore, our platform automates these steps.

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 training data is needed to train a functioning model, which can be later improved via further training as more data are collected [219]. 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. 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

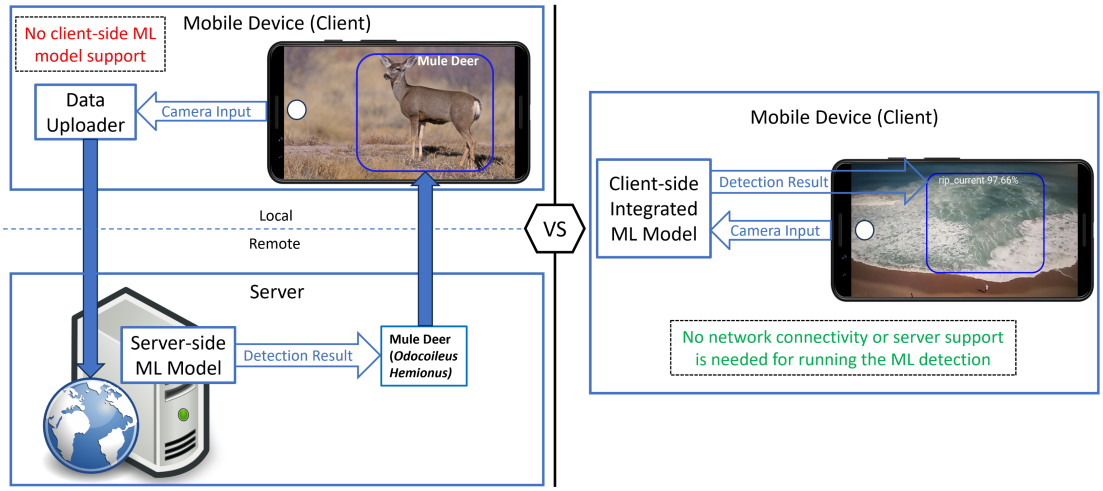


Figure 4.2: Server-side (left) vs client-side (right) ML models. We used client-side ML models in our implementation, which provided real-time object detection without any network connectivity or server-side processing requirements.

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 4.2). A server is only necessary for uploading the collected data. Tools for data uploading, user guides, and tutorials are also included in the templates.

4.5 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 tools such as AnyLabeling, TensorFlow, PyTorch, and Colab Jupyter Notebook [1,253]. 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.

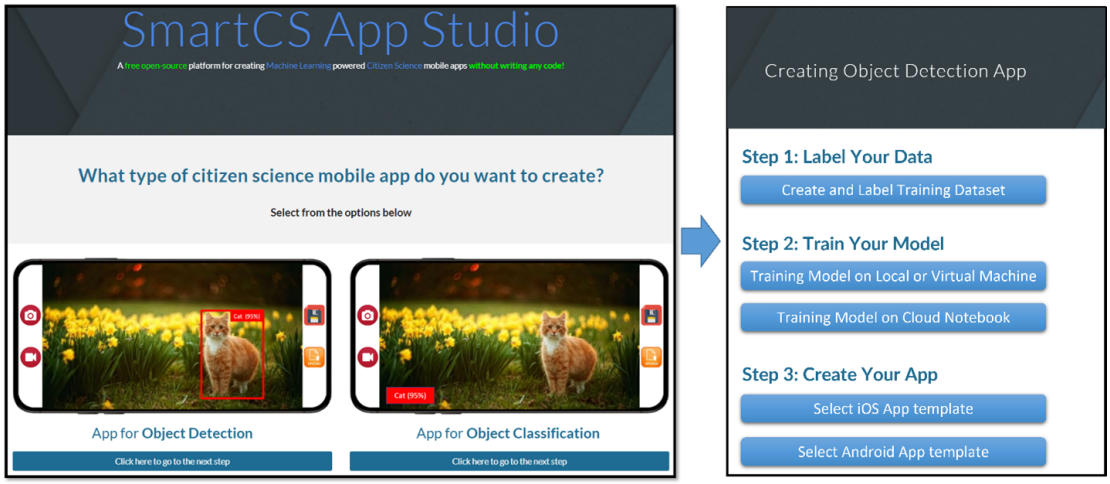


Figure 4.3: The web version of the platform was created for easy access and uses different feasible computational resources for different steps.

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 4.3). 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.

4.5.1 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 [253] or AnyLabeling (AI Curious 2023). Our platform also supports converting labels obtained using Amazon's Mechanical Turk [189]. Additionally, we a provide some custom tools that we created using Python 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 [204].

4.5.2 ML 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 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 [1]. Our platform provides the options to select the most appropriate model based on the information provided in Table 4.1 and the planned usage of the app [215]. Note that detection accuracy of the utilized ML models is considered reliable based on its Mean Average Precision (mAP) [186]. The models are then custom trained using the dataset created in the previous step by the transfer learning process [8]. We provide tutorial documentation and video for model training.

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 [264]. The training needs to run until the model converges, which is indicated when the loss function value drops below a certain threshold [6]. 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.) [123, 163]. Further discussion about guidance on setting up these training parameters in the platform is included in the Supplemental Contents section at the end of this chapter. ML model training for the case studies presented in this chapter took about 2 to 3 days using inexpensive CPU-only machines, which can be done within a few hours using more expensive GPU machines.

4.5.3 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 [132] 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 [178], 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.

4.6 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 the Supplemental Contents section at the end of this chapter.

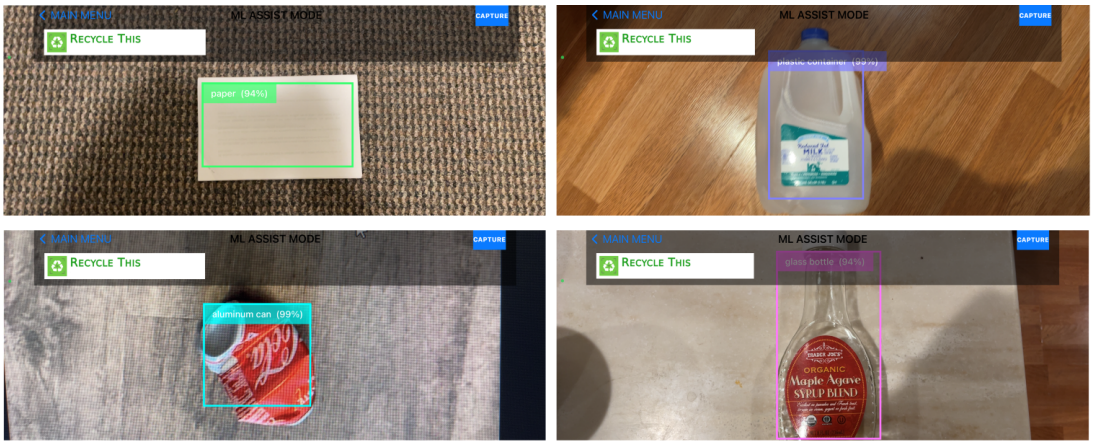


Figure 4.4: This figure illustrates the type of materials that the "Recycle This" app can detect papers, aluminum cans, plastic containers, and glass bottles.

4.6.1 Use Case: Recycle This

The practice of recycling is essential for the environment and the future of our planet [4]. 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 (Informa Markets, 2019). 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 4.4). The project focused on classifying objects like glass bottles, plastic containers, cardboard, paper, and aluminum cans. 2,500 training images were sourced from the public image datasets [114, 125, 208]. 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 [279].

Figure 4.5: 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.

4.6.2 Use Case: RipSnap

The importance of rip current detection was discussed earlier. The citizen science app RipSnap is based on the idea of CoastSnap [100], 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 [65]. The app's primary aim is to enhance beach safety by educating users about the presence of rip currents (Figure 4.5). A lightweight ML model integrated into this app assists non-expert 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 [53], and additional data collected using drones and beach monitoring webcams.

4.7 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 [152]. 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.

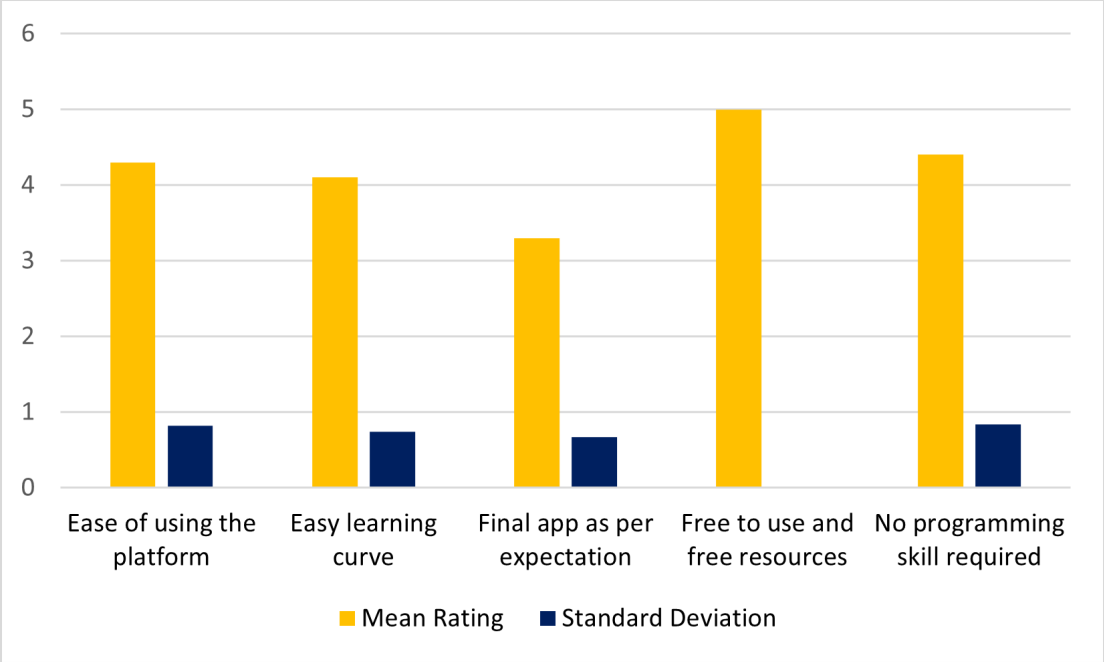


Figure 4.6: Summary of scores from the user study by app creators. Participants were asked to rate their experience of using the platform on a 5-point Likert scale from "Poor" (a score of 1) to "Excellent" (a score of 5).

4.7.1 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 is 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 ML model training, which in turn was influenced by the number of available GPUs. More details about the apps are provided in the Supplemental Contents section at the end of this chapter.

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 4.6 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, 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 non-expert users.

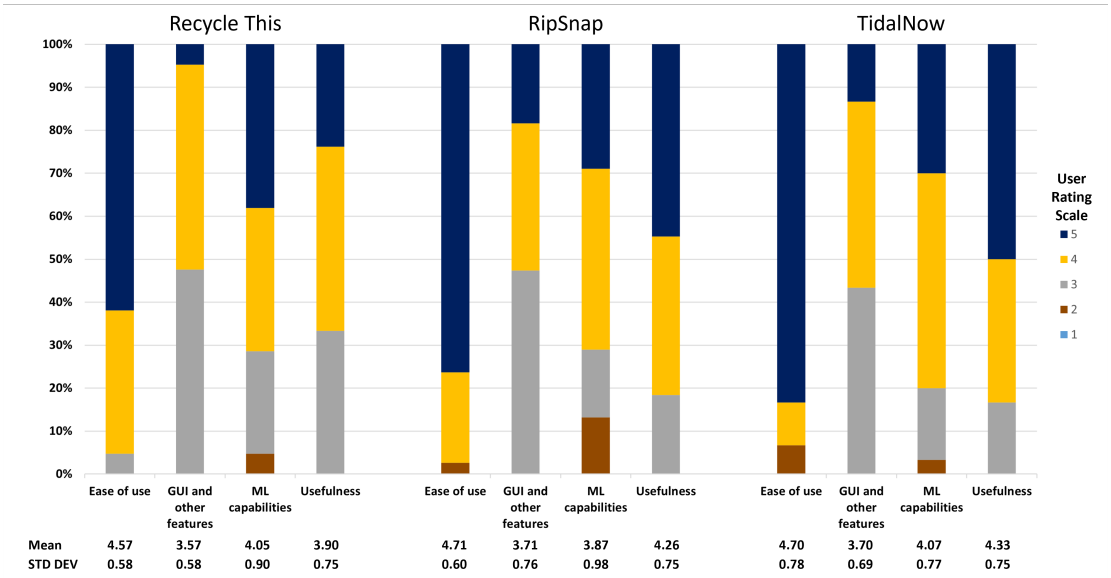


Figure 4.7: Summary of scores from the user study by users of three apps. Participants were asked to rate their experience of using the platform on a 5-point Likert scale from “Poor” (a score of 1) to “Excellent” (a score of 5).

4.7.2 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 is 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 4.7). 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.

4.7.3 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.

4.8 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 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 non-expert 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 just with ML. 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.

Ethics and Consent

This research 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.

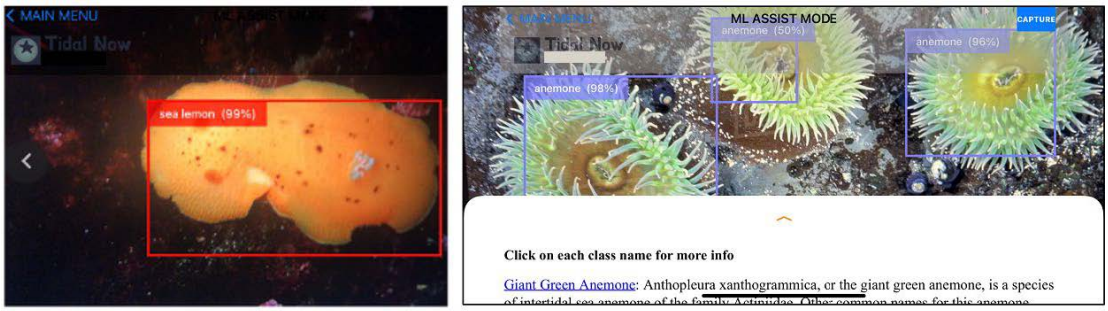
Supplemental Contents

Other Use Cases

This section briefly describes the other apps created using our platform.

TidalNow

There are different dimensions of animal biodiversity (species richness, phyletic richness, and functional diversity) in tide pools. Activities involving observation of wild organisms in the tide pool can provide recreational and learning opportunities [74]. TidalNow is a citizen science mobile app developed by a high school student. The app uses an ML model to identify different types of saltwater marine species in tide pools. The ML model integrated into the app is trained to detect five different species: giant green anemone, ochre stars, lined chiton, sea lemons, and black turban snails. The ML model for this app was trained using about 600 images for each class. Although apps like iNaturalist or Google Lens can recognize these specimens, they require server-side processing and internet connectivity. However, many tide pools are located on beaches with limited or no internet access. As this app works without internet connectivity, it works perfectly fine in these remote locations (Figure 4.8).



a) b)

Figure 4.8: The TidalNow App is shown here. Similar to other apps, this app (a) shows the detected object using a bounding box, (b) the selected template has a built-in pull-up panel that was customized to present additional information about the detected objects.

Sk.in

Skin conditions are more prevalent than other illnesses in all countries worldwide [5]. Some skin diseases can be lethal [7]. Although the advancement of lasers and photonics based medical technology has made it possible to diagnose skin diseases much more quickly and accurately, the cost of such diagnosis is still very expensive [5]. So, there is a lot of research interest in detecting skin diseases using ML-based computer vision [5,201,224,231]. Even though some of these recent works demonstrate very accurate skin disease detection using CNN, there are not many works that can do this in real-time on mobile platforms. Sk.in is a mobile application that utilizes ML object detection to categorize dermal conditions as bacterial, fungal, parasitic, viral infections, or allergic reactions in real-time. Sk.in intends to increase the efficiency of diagnosing and treating generalized skin conditions for the public and is designed for everyday use. Developed primarily as an educational tool by a high school student, this app also facilitates data collection on skin infections across a diverse demographic. The ML model can also be trained to detect more skin diseases, such as melanoma and other types of skin cancers [7]. Figure 4.9 (a)-(c) displays the app's capability of detecting bacterial infection, allergy, and viral infection.

Seal vs Sea Lion

Biodiversity analysis is important for many research groups, such as those with a focus on biological science, aquaculture, marine biology, etc. Researchers may need

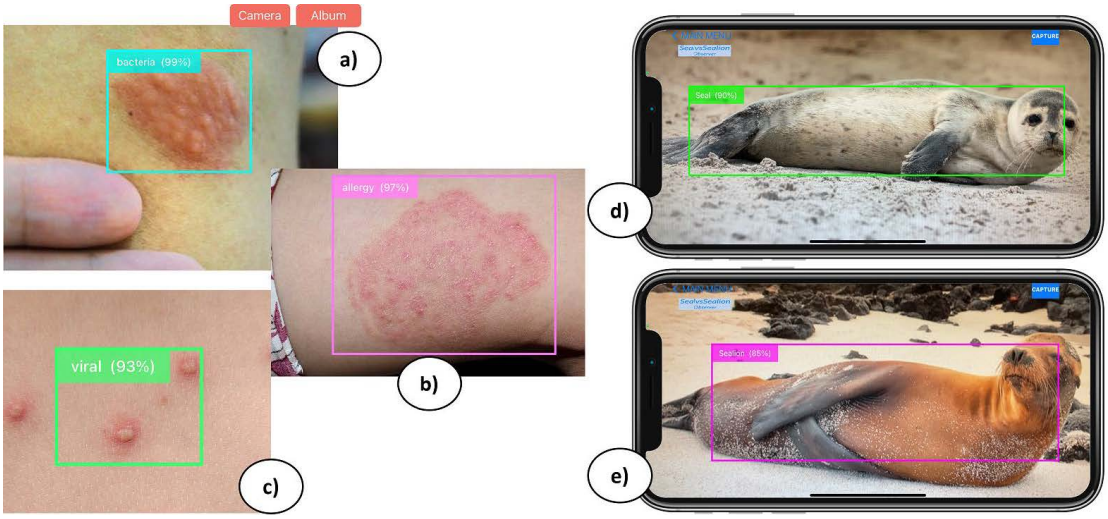


Figure 4.9: (a)-(c) Demonstrates that the "Sk.in" App can detect bacterial infection, allergy, and viral infection. (d)-(e) Shows example results from the sea lions and seals detection and differentiation app, where detected seals are highlighted using green bounding boxes and sea lions are shown using magenta bounding boxes.

to collect data about some endangered species; other times, they need data to analyze the biodiversity in some specific area [272, 273]. In this use case, we trained a model with images of sea lions and seals to demonstrate our app's usability for these types of research projects. Many sea lion species are considered endangered [43] and collecting data about them is needed for marine biology research and conservation groups [27]. However, differentiating between seals and sea lions can be challenging for non-expert participants [273]. Using our ML-powered app, the participants can detect and differentiate between these two species (Figure 4.9 (d-e)). With further training data and re-training the model, this app can be modified to detect and differentiate among various sub-species [99]. The same concept can be applied to create educational and data collection apps about other animal species.

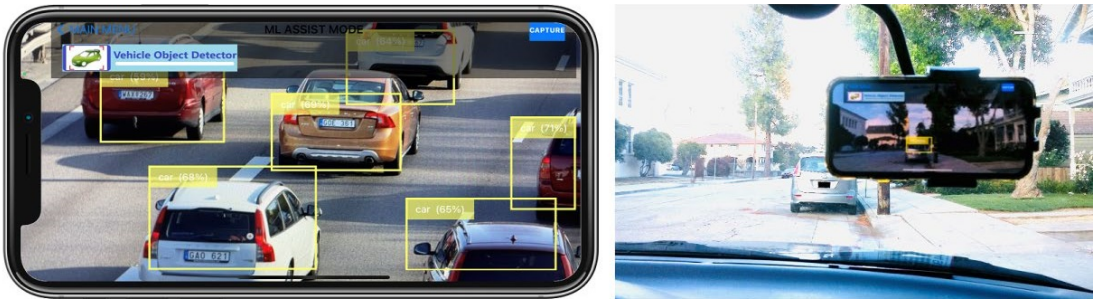


Figure 4.10: The vehicle object detection app is shown here. The app is used for collecting vehicle data by mounting it on the windshield of a car.

Vehicle Object Detection

The vehicle detector is a citizen science mobile app for collecting video data about road objects relevant to autonomous vehicle research. It was created as part of an autonomous vehicle research project in collaboration between a university research group and a company from two different countries. The app can be used for road object detection and data collection by mounting the phone next to the windshield of a car. Various standardized datasets exist for autonomous vehicle research, such as Kitti, Waymo, NuScene, etc. [126]. However, these are collected using arrays of advanced sensors mounted on specialized vehicles [117]. This app facilitates small-scale experiments and data collection worldwide, employing a simple and inexpensive mobile setup (Figure 4.10). Even using the app with multiple mobile devices to collect multiview datasets would be much cheaper than using the traditional autonomous vehicle data collection setup. Therefore, it can be used to collect data using citizen science from different developing countries about various exotic vehicles, such as three-wheelers, rickshaws, etc. These vehicles are not commonly seen in developed countries

and, thus, not present in the standard datasets [130].

The other apps developed with the platform include those for detecting plant leaf diseases, identifying beach debris, recognizing types of building architecture, assessing fingernail conditions, counting blood cells, and classifying ultrasound image types.

Summary of Used ML models

Table 4.1: This table presents a summary of the ML models tested and supported on our platform.

Model Name	**Inference Speed (ms)**	**mAP for COCO objects**	**Mobile model size (MB)**
SSD MobileNet v1	48	29.1	5
SSD MobileNet v2	39	28.2	5
EfficientDet D0	39	33.6	6
EfficientDet D1	54	38.4	8
EfficientDet D2	67	41.8	11
YOLOv8m	32	50.2	49

The Table 4.1 lists the ML models tested on our platform, enabling app creators to select the most appropriate model for training [1]. The information provided here helps the app creators to select a model for training. The faster the inference speed (second column), the app gains better real-time performance. The higher mean average precision (mAP) in the third column represents better precision for detecting objects. Comparing the second and third columns shows that higher precision requires

more inference time, leading to slower than real-time performance. The app creator needs to decide about the trade-off between these two. The fourth column shows the approximate final size of the converted trained model, which may impact performance on older devices with lower computational resources.

In our platform's current selection of compatible models for mobile devices released up to 2023, YOLOv8m is the largest recommended model that runs smoothly on a typical consumer mobile device, with a saved weight size of 49 MB. However, even though YOLOv8m shows better performance metrics on benchmarks, we found that EfficientDet D2 offers more stable performance overall during our testing with a few current-generation smartphones (Apple's iPhones and Google's Pixels). Over time, with the introduction of more powerful mobile devices and larger compatible models, these can be included in our platform without significant modifications.

The number of images needed to train an object detection model can vary widely depending on several factors, such as the complexity of the task, data quality, and model architectures. Decent results can still be achieved even with hundreds of images per class. [223] suggest an inflection point of around 150-500 images per class, beyond which the earlier sharp performance gains start to level off. However, [21] argue that for optimum accuracy, having at least 2000 different images for each class is desirable. As a rough guideline, for a simple object detection task with a few object classes and relatively consistent object appearances and backgrounds, a few hundred to a few thousand images might suffice, especially if transfer learning is utilized. For more

complex tasks or a larger number of object classes, tens or even hundreds of thousands of images might be needed.

The number of classes supported for training depends on the model type and architecture. For instance, according to official documentation, EfficientDet can support up to 999 classes [239], whereas YOLOv8 (Ultralytics, 2023) does not have a defined hard limit for the number of classes. However, the choice of model determines the number of classes; it is not a limitation of our platform, SmartCS, since we can incorporate newer versions of models that support more classes.

Although transfer learning cannot be performed within the app due to technological limitations and the resource constraints of mobile devices, which render model training infeasible, models can be updated periodically with newly collected data by retraining them using transfer learning on more powerful machines or cloud servers. Additionally, we provide pre-trained models derived from well-known public image datasets, such as MS-COCO, to serve as a starting point for training on new datasets via transfer learning.

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.

Chapter 5

Citizen Science Tools with ML as a Pathway to Engage High School Students in Research

Efficient data collection can be achieved by creating tools with ML guidance to reduce incorrect data collection and label noise among a larger population. One approach to accomplish this is to make the creation process of these tools easy enough for non-programmer general user groups, such as high school students [139]. This chapter outlines an approach to engage high school students in research using citizen science tools embedded with ML models. In the context of fostering early engagement in scientific research among high school students, this chapter explores SmartCS and other supporting tools. Our experience suggests that this approach enabled students to

learn aspects of computer science and engineering, particularly in ML model training and mobile application software development. It also allowed them to experience firsthand the significant role citizen science can play in collecting and analyzing scientific data.

5.1 Introduction

In recent years, there has been growing emphasis on integrating research methodologies and scientific thinking into earlier stages of education, enriching high school students' learning experiences and preparing them for future academic and professional challenges [258]. This engagement not only fosters a passion for science and technology but also profoundly influences students' academic paths, enhancing critical thinking, problem-solving skills, and a sense of scientific contribution [248]. This chapter presents an approach that integrates citizen science applications with ML techniques to actively involve high school students in meaningful research activities.

Citizen science allows the public to participate in research projects. In a typical citizen science scenario, the public collects data using smartphone applications (commonly known as apps), and professional researchers utilize this data in their studies [148]. For example, apps can be developed to collect images of coastal sea life for use by ocean science researchers, as well as apps for users to collect images of leaves for use by botanists. These apps can be augmented with ML to provide automated guidance for complex data collection tasks [136, 183]. By leveraging a platform

for the codeless creation of ML-enhanced apps, high school students can engage in impactful citizen science research projects without needing extensive technical training or programming knowledge.

This work focuses on developing citizen science apps that collect and analyze visual data (images and videos), leveraging computer vision ML models [236]. This approach provides students with hands-on, socially relevant, and impactful exposure to ML. Recent advancements in large language models, such as Generative Pre-trained Transformers (GPTs), have made them suitable for various applications, including citizen science [80]. However, applying ML for computer vision-based guidance presents greater challenges due to the subjectivity of visual data compared to linguistic data. Object detection or identification using ML models is a fundamental technique in computer vision [285], which can be especially useful in citizen science applications where participants benefit from visual cues. To this end, we utilized the codeless platform SmartCS [138] and its supporting tools to create citizen science apps with object detection capabilities. This enables high school students to learn about ML through the application of computer vision in creating citizen science tools while engaging in research.

This chapter explores the following research questions (RQs):

1. **RQ1:** How does integrating ML within citizen science tools influence high school students' engagement and interest in scientific research?

2. **RQ2:** How effective is using a codeless platform in teaching high school

students complex concepts such as data curation, ML, and mobile application development?

3. **RQ3:** What learning outcomes are observed among high school students who develop ML-powered mobile applications through the SmartCS platform?

4. **RQ4:** What impact do student-developed ML-powered apps have on public participation in citizen science projects?

5.2 Related Work

5.2.1 Citizen Science in High School Education

In the past, there have been various efforts to integrate citizen science into the learning process at the high school level [222]. However, it is much less prevalent than in higher education [106, 230]. For high school students, citizen science offers learning opportunities and engagement across various scientific fields [134]. It also serves as an early introduction to and motivation for STEM careers [105]. Its benefits have been demonstrated for students as young as those in the fifth [135] and eighth grades [188]. Also, there have been initiatives for driving innovation through project-based learning for social good, participated in by middle and high school students [173]. This project aims to further bridge the gap between high school education and advanced scientific research by leveraging the power of citizen science.

5.2.2 ML in High School Education

Teaching fundamental AI concepts and techniques, including ML as a sub-field, has traditionally been confined to higher education [214]. Recently, computing education has begun to be included in high school curricula worldwide, incorporating some advanced topics [133, 171]. For instance, [262] introduced a sandbox methodology for teaching computing-based data science to high school students. However, computer science course content at this educational level rarely covers AI or ML [110]. In the last year, though, there has been an influx of GPT-4-based applications for education, including those aimed at high school students [131]. Even though opportunities for high school students to explore ML and its societal implications have been investigated, approaches for teaching them about ML and its application in research remain very limited [127]; this is something we strive to address with our work.

5.2.3 ML and Citizen Science

Limited but successful applications of ML have been observed in the field of Citizen Science in the past, including mobile apps like iNaturalist [183], PlantNet [90], and LeafSnap [149]. However, the untapped potential of using ML in citizen science remains vastly unexplored [138]. Echeverria et al. [66] demonstrated that the citizen science platform iNaturalist is a valuable tool for carrying out collaborative projects in secondary education. Our work aims to build on past successes by integrating ML to improve the accessibility of citizen science projects. This could potentially lead

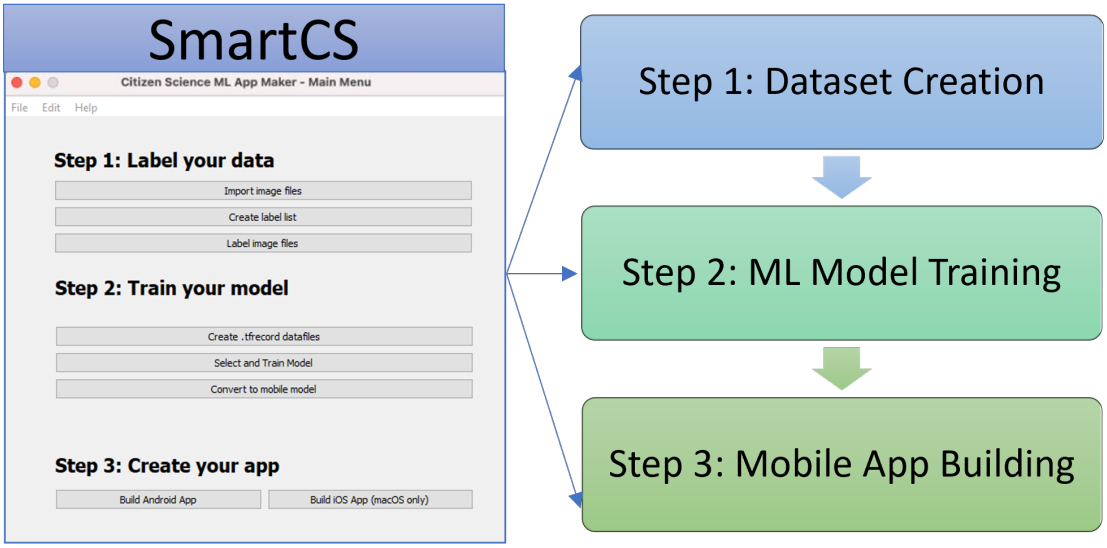


Figure 5.1: The graphical user interface of SmartCS facilitates the three steps required to create an ML-powered citizen science mobile app.

to a new paradigm in high school educational methodologies, making learning more interactive and engaging.

5.3 Methodology and Research Setting

5.3.1 Participant Selection

The high school students were recruited through a summer science internship program offered by an R1 institution. A research mentor, typically a graduate student, provides the students with guidance throughout the project. It was clearly communicated in the participant call that no programming skills were required for participation in the project. We enrolled twelve participants, comprising seven male and five female students, ranging in age from 14 to 17 years. These participants were

from diverse backgrounds, including individuals from both the USA and India.

5.3.2 Structure

Our initiative provided a comprehensive research experience in three phases: (1) conceptualization of citizen science projects, (2) development, and (3) deployment of mobile apps. Table 5.1 summarizes the learning goals and tools used in each phase.

5.3.2.1 Conceptualization Phase

In this phase, students explore the application of ML to citizen science and identify potential research areas through an open-ended approach. Guided by mentors, students define the scope and objectives of their projects—such as designing an app to detect whether a given object is recyclable. The mentor ensured that the projects were feasible to develop within the given two-month timeframe and that students were reviewing relevant literature while ensuring they received sufficient information about the tasks at hand. They learned how to conceptualize a science project by reviewing current literature, identifying a research problem, and proposing a solution that integrates computer vision, ML, and citizen science. Thus, this phase introduced them to research methodology.

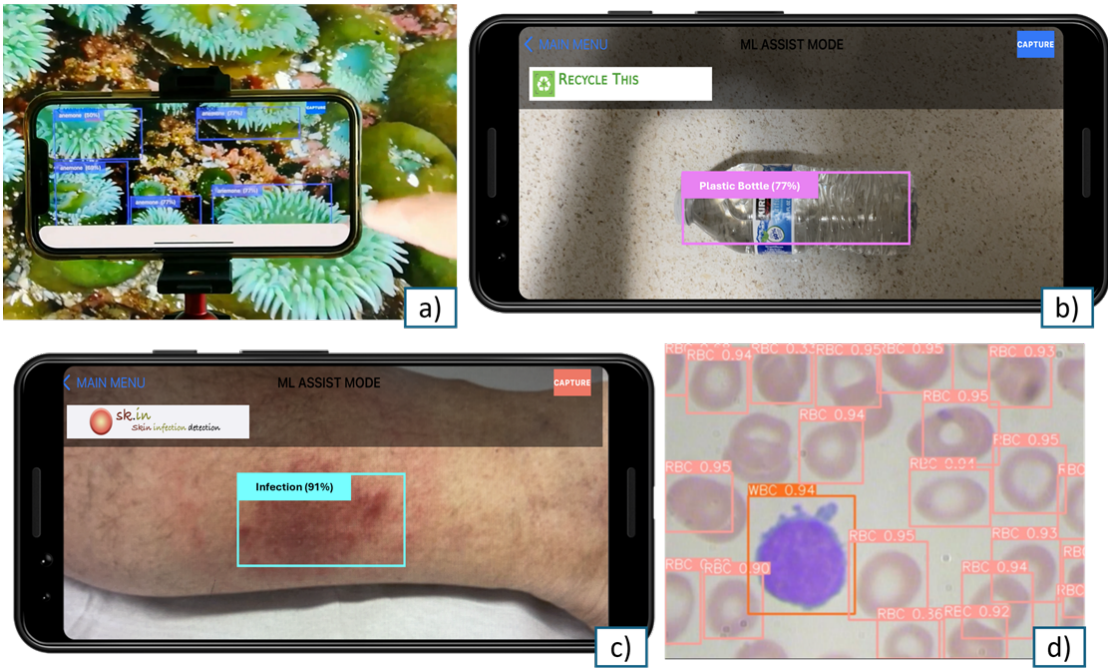


Figure 5.2: Some of the apps created by the high school students include: (a) Tidepool species identification, (b) Recyclable object detection, (c) Skin infection identification, and (d) Blood cell type identification.

5.3.2.2 Development Phase

The primary tool for developing ML-powered applications was the SmartCS App Studio and its supporting tools. The development phase, guided by SmartCS, involves three main steps: dataset creation, ML model training, and mobile app building (Figure 5.1). This phase includes training on ML basics, data collection and processing, and app development. Examples of student-created apps are shown in Figure 5.2.

Dataset Creation: Students gathered and labeled data, ensuring it was appropriately formatted for ML model training. For instance, for the tidepool species identification app, this involved collecting many clear images of tidepool species suitable for computer-vision-assisted identification. This step helped them understand the importance of quality data and how it directly impacts the effectiveness of ML models. Data was sourced from existing (often multiple) public datasets or collected manually, depending on availability. Open-source tools, such as LabelImg and AnyLabeling, were used to label images. If any conversion of dataset formats was needed, the free tool Roboflow was used.

ML Model Training: This step teaches students the fundamentals of ML algorithms and the model training process. Participants used prepared datasets to train at least one lightweight ML model suitable for mobile devices, selecting from SSD-Mobilenet, EfficientDet, and YOLO models [285]. Training was conducted on a local computer or via a remote server using tools like Google Colab, Amazon Web Services, and

Table 5.1: Summary of Learning Activities by Phases and Steps.

Phase	**Step**	**Learning Goals**	**Tools Learned**
Conceptualization	-	Study literature and state-of-the-art to conceptualize the idea of a project	Google Scholar
Development	Dataset Creation	Creating a dataset or repurposing a public dataset	SmartCS, LabelImg, AnyLabeling, Roboflow
	ML Model Training	Training an ML model using local PC/cloud computing	SmartCS, Google Colab, AWS, Ultralytics Hub
	Mobile App Building	Building an app for iOS and/or Android phones	SmartCS, Xcode, Android Studio
Deployment	-	Recruit users, distribute the apps, and collect user feedback	TestFlight, Google Drive, Google Forms

Ultralytics Hub. This experience introduced students to ML concepts within the context of computer vision applications.

Mobile App Building: Students integrated their trained ML models into an iOS and/or Android mobile app, using pre-developed templates provided by SmartCS. The learning objective here was to comprehend how ML models can be applied in real-world applications through mobile app development. This step also served as students' introduction to software engineering. They either used Xcode for iOS app creation, or Android Studio for Android app creation (Figure 5.2). In this step, students' visions for their apps came to life. For example, an app could allow a user to identify a tidepool species in real-time using their smartphone camera, with the image then uploaded for further research use.

5.3.2.3 Deployment Phase

The mobile apps were distributed among beta tester users, and students collected and analyzed user feedback. This served as their introduction to software testing, teaching them how to gather user feedback and use it to refine their projects. They also gained exposure to the human-computer interaction process.

5.4 Preliminary Results and Discussion

Here, we reflect on our preliminary observations as guided by the RQs. We evaluated outcomes through observations of student learning during the program, informal student comments, measures of project completion success, and beta testing of two student apps.

RQ1: How does integrating ML within citizen science tools influence high school students' engagement and interest in scientific research?

We observed that integrating ML within citizen science tools significantly boosts high school students' engagement and interest in scientific research, perhaps by making learning more interactive and impactful. We hypothesize that the ML-enhanced app's real-time feedback using object identification to enable learning and guide data collection makes the scientific process more exciting and understandable. This unique approach to designing science projects may help maintain the curiosity and excitement of the students creating the apps. Moreover, hands-on application of cutting-edge technology like ML exposes them to modern scientific methods, potentially sparking a lasting interest in STEM careers. By contributing to real-world research through state-of-the-art yet user-friendly ML tools (as listed in Table 5.1), students can see the practical impact of their work, enhancing their sense of contribution and the relevance of their educational activities.

RQ2: How effective is using a codeless platform in teaching high school students complex concepts such as data curation, ML, and mobile application development?

We hoped that using a codeless platform to teach high school students complex concepts would enable them to rapidly plan and engage in meaningful scientific research without requiring extensive background knowledge. The number of functional applications students have created, as summarized in Table 5.2, suggests this may indeed be a successful approach. This direct engagement with technology may enhance students' understanding through experiential learning. By making these tools accessible, students from diverse backgrounds can develop crucial digital age skills such as data curation, understanding ML algorithms, and gaining app development experience.

Qualitative feedback from participants also underscores the program's value. For instance, Student S7 remarked, "*I really enjoyed the program as it was beginner-friendly. Having never done any ML work previously, it showed the benefits and possibilities, while making it easy to label, create, and turn models into apps.*" S9 commented, "*The app creation platform expedited the process of creating the ML app and allowed me to easily deploy the ML models trained to perform real time detection. Furthermore, the app creation platform allowed us to control various parameters, which would improve the model performance.*" We find it especially noteworthy that students effectively used ML terminology to describe their experiences.

Table 5.2: List of projects created by the high school students.

Student	**Project**	**ML Model**	**Phone App**	**Publication**
S1	Tidepool species identification	Yes	Yes	-
S2	Recyclable object detection	Yes	Yes	Yes
S3	Skin infection identification	Yes	Yes	-
S4	Road vehicle type recognition	Yes	Yes	-
S5	Plant disease detection	Yes	-	-
S6	Beach debris identification	Yes	Yes	-
S7	Building architecture identification	Yes	Yes	-
S8	Fingernail conditions assessment	Yes	Yes	-
S9	Blood cell type identification	Yes	Yes	Yes
S10	Ultrasound images type classification	Yes	-	-
S11, S12	Differentiate between seals and sea lions	Yes	Yes	-

RQ3: What learning outcomes are observed among high school students who develop ML-powered mobile applications through the SmartCS platform?

High school students who participated in developing ML-powered mobile applications using the SmartCS platform were expected to achieve several key learning goals and learn some crucial software tools, as listed in Table 5.1. For example, using AnyLabeling, which allows data labeling with AI support from YOLO and Segment Anything, students can learn how ML models support real-life computer vision tasks. Additionally, the process fosters critical thinking, problem-solving, collaboration, and communication skills. These competencies enhance academic aptitude and prepare students for future STEM careers.

The apps successfully created by the students (Table 5.2) suggest that students have likely met the learning goals, including having developed the skills to use the tools outlined in Table 5.1. Notably, two students published their work at IEEE conferences following the two-month program [116, 279]. Publishing was not an intended goal of our initiative, as it is uncommon for high school students to publish, so this is especially suggestive of a transformative learning experience. This highlights both the long-term positive impact of our initiative and the potential of high school students to contribute to scientific research. Additionally, it is notable that five out of twelve students who have since graduated high school have already enrolled in STEM undergraduate programs.

RQ4: What impact do student-developed ML-powered apps have on public participation in citizen science projects?

To explore this RQ, we pilot tested the usability of two selected apps. This usability testing aimed to demonstrate the effectiveness of ML-based object detection apps in helping participants capture useful data. We recruited 20 participants, consisting of 10 males and 10 females, aged between 18 to 54. They were asked to use student-created apps to conduct (1) tidepool species identification and (2) recyclable object detection. We conducted this test in a lab, which enabled us to determine the participants' success rates and collect their feedback.

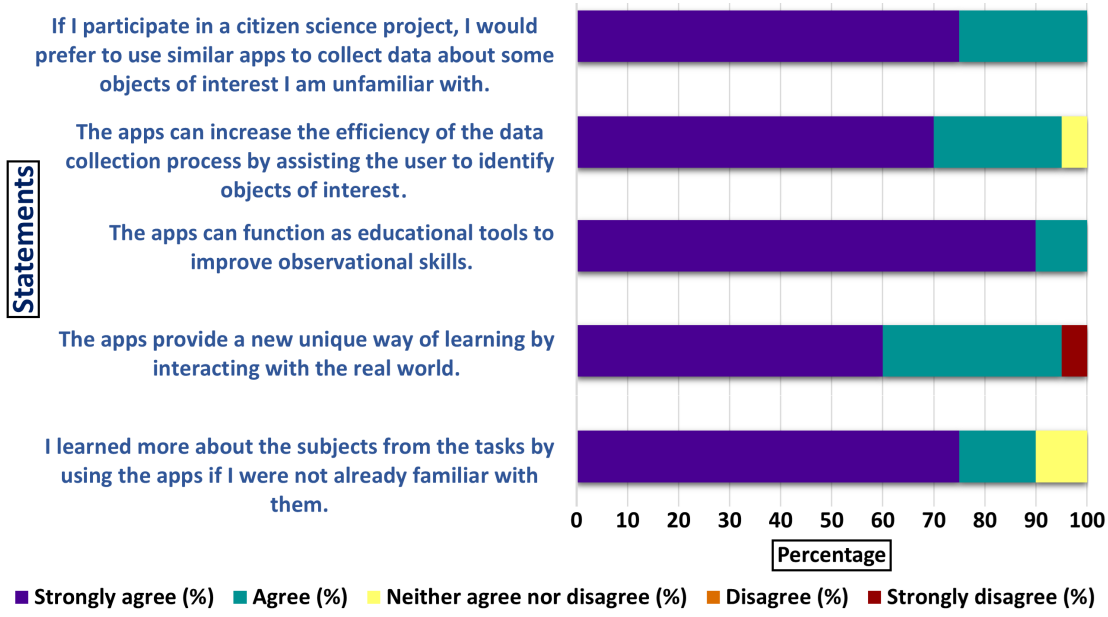


Figure 5.3: Summary of feedback from users on their experience of using the apps. Here, in the statements, "the apps" refer to the two selected apps created by the students.

Feedback from app users (Figure 5.3) indicates that ML guidance improved participants' experiences. They acknowledged the positive impact of ML-powered

apps on citizen science, highlighting the object identification feature as helpful for data collection. Additionally, participants viewed these apps as educational tools that provide interactive learning experiences, suggesting that ML-powered apps can significantly enhance public participation in citizen science by making data collection more accessible and engaging.

5.5 Conclusion and Future Work

This chapter demonstrates early effectiveness of integrating citizen science and ML to engage high school students in scientific research. A codeless platform, such as SmartCS, proved to be a valuable tool to this end, enabling students to create meaningful projects that contributed to both their educational growth and the scientific community.

Our future work will include formal evaluations of each RQ through rigorous user studies. We also plan to expand the scope of citizen science projects to various domains and ML applications beyond object identification. Additionally, we aim to explore applying this approach to younger students, including those in middle and elementary schools.

Ethics and Consent

This research 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.

Chapter 6

RipFinder: Real-Time Rip Current Detection on Mobile Devices

In this chapter, we introduce RipFinder, a mobile app with ML models trained to detect multiple types of rip currents. Chapter 3 briefly discussed RipSnap, an app created using SmartCS with a similar application; however, its features are limited to the templates provided by SmartCS. Additionally, the functionality of RipSnap and any other apps created by SmartCS is designed to be limited to client-side operations only. In contrast, we propose RipFinder as a more advanced client-server ML model-based computer vision system designed to leverage both small client-side models and large server-side models. While an app such as RipSnap can be created without knowing how to write code, developing the advanced features for RipFinder requires more advanced knowledge of programming, ML, and other aspects of computer science. Unlike

the basic data collection app RipSnap, the advanced data collection app RipFinder enhances rip current detection accuracy by utilizing multiple ML models such as EfficientDet and YOLO on the client side and larger models like Vision Transformers (ViT) on the server side. This approach leads to more efficient data collection and significantly reduces label noise, ensuring higher precision in identifying rip currents. This chapter presents RipFinder's overall design and discusses the challenges inherent to the rip current detection system.

6.1 Introduction

Rip currents are dangerous, strong, fast-moving currents that pull swimmers away from the shore, often leading to drownings and fatalities. They pose a significant hazard to beachgoers and can easily overpower even strong, experienced swimmers. Rip currents are a global issue, affecting coastlines around the world [179, 206, 283]. In the United States alone, they account for an estimated 100 drownings a year [86]. Rip currents can form suddenly and without obvious signs, which can catch swimmers off guard. While there are general conditions that can lead to their formation, predicting exactly when and where they will appear is challenging. Furthermore, rip currents are created through various mechanisms and, as a result, exhibit different visual characteristics. This complexity of occurrence and variability in appearance makes them difficult to identify [35]. Consequently, many beachgoers lack the essential knowledge and awareness needed to recognize and avoid these perilous currents.

Rip current detection techniques are significantly important because of their potential to save lives. As a public safety issue, the implications extend beyond swimmers. Lifeguards, rescue teams, and even bystanders who try to help can also be put in danger. If rip currents could be detected reliably, then beachgoers and lifeguards could be alerted to the dangers in real time. This would likely result in a significant decrease in the number of rip current-related incidents and fatalities. By providing more accurate information about rip currents, the general public could make more informed decisions about when it is safe to enter the water, thereby enhancing overall public safety. The development and deployment of tools, such as rip current prediction models [65] or mobile apps that can detect and provide real-time alerts and tips about rip currents, could be instrumental in these efforts.

While rip currents can often be visually identified by experienced swimmers, surfers, lifeguards, and coastal scientists, traditional detection and data collection methods typically involve in-situ instrumentation, such as GPS-equipped drifters and current meters [153, 170]. However, recent studies have demonstrated that images and video can also be used for detecting rip currents. These approaches leverage computer vision and ML models for object detection to spot and identify these potentially dangerous phenomena [53, 55, 64, 174, 177, 193, 199, 200].

However, detecting and segmenting rip currents with high accuracy using ML methods presents unique challenges due to their amorphous and ephemeral nature. Given the potentially fatal nature of dangerous rip currents, their detection is a matter

of life and death. Thus, high accuracy and reliability are crucial for any rip current detection tool for issuing warnings and taking preventive actions to decrease the number of rip current-related incidents. Providing such capability for real-world use i.e. on mobile platforms adds another layer of technical challenge.

Many object detection ML models can detect rip currents, but the challenge lies in deploying these models in real time on mobile devices with limited power and computational resources. More accurate yet computationally resource-intensive, ML models cannot run directly on mobile devices. By sending the visual input for object detection to a remote server, it can be achieved on mobile devices. However, this approach is not always feasible, especially in beach locations where server connectivity is unavailable. Alternatively, mobile-optimized ML models can feasibly run using the limited computational resources of portable devices without server connectivity but at the cost of sacrificing accuracy.

To address these challenges, we introduce a mobile application, or app, designed to detect rip currents using ML models for computer vision. Users can identify potential rip currents in real-time by simply aiming their phone's camera toward the ocean. We propose a client-server system of object detection models to balance the trade-off between computational speed and accuracy. Depending on the mobile device's available computational resources and internet connectivity, this app employs one or more ML models to identify rip currents. If the device is relatively new with adequate computational resources, our app runs two different types of mobile-

optimized ML models to enhance the reliability of rip current detection. For older, resource-constrained devices, only one ML model is used. Moreover, when internet connectivity is available, part of the visual data is transmitted to a server for further verification of the detection using a more accurate large model. Our system combines client-server architecture with multiple ML model-based computer vision to enhance the accuracy and reliability of rip current detection. The novelty of our solution lies in its implementation of this combined system, allowing the app to function both with and without internet connectivity. Our app's versatility is especially invaluable in areas where lifeguards are absent or internet access is limited, establishing it as a crucial tool for public safety.

In addition to rip current detection, our app places a strong emphasis on educating users about the dangers of rip currents through informative in-app content and links to additional resources. Our aim is to empower beach enthusiasts with the knowledge necessary to make informed decisions, protecting themselves and others from these hazardous rip currents. Moreover, our app includes a citizen science feature, enabling users to contribute to scientific knowledge. This is done by encouraging them to record and share data such as geotagged images and videos, along with additional information about detected rip currents. Harnessing the collective power of app users, we can gather valuable data that improves our understanding of rip currents and helps verify existing rip current forecast models. Ultimately, this leads to the development of more effective safety measures and strategies.

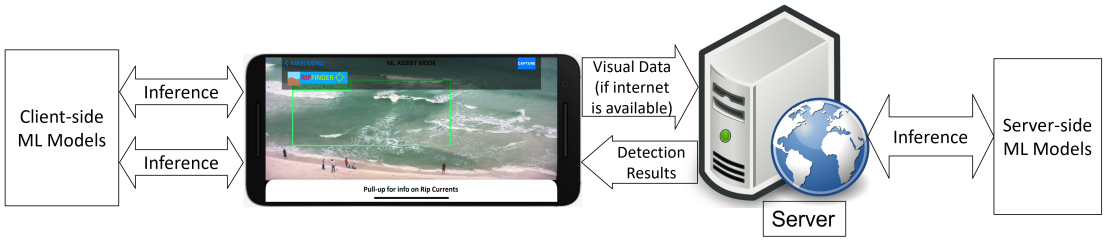


Figure 6.1: The high-level system architecture of RipFinder.

The contributions of this chapter are as follows:

- Introduction of RipFinder: a mobile app designed for real-time, vision-based rip current detection.
- Development of a client-server system tailored for the ML models utilized in the rip current detection app.
- A comprehensive analysis and comparison of state-of-the-art ML models for rip detection.

6.2 Related Work

6.2.1 Realtime Object Detection

Developing a mobile application for effectively and reliably identifying rip currents necessitates real-time object detection capabilities. Deep learning has revolutionized the field of object detection, as well as other computer vision tasks. Convolutional neural networks (CNNs) have become the standard method for these applications.

Numerous large and intricate models, such as Faster R-CNN—a two-stage region-based detector [205]—and DETR (Detection Transformers)—an object detector based on the Transformer architecture [34]—offer remarkable accuracy in object detection tasks. For instance, Faster R-CNN has been adeptly used for real-time object detection in drones by connecting to a remote GPU server [156]. However, these detectors often bear significant computational complexity, rendering them difficult to deploy on mobile or embedded platforms for real-time performance. An earlier server-based system named Glimpse, offering continuous, real-time object recognition for mobile devices, was introduced by Chen et al. [42]. Nonetheless, server-reliant systems prove impractical in locations devoid of internet connectivity.

Achieving accurate and reliable real-time object detection on mobile devices without depending on servers presents inherent challenges. Numerous efforts have been directed towards integrating deep learning methods on mobile devices by creating compact, mobile-optimized ML models. Typically, streamlined architectures, like one-stage CNNs, render the models lightweight, allowing them to function swiftly on mobile devices—making them an ideal choice for real-time object detection. The primary compromise for such efficiency is a minor decrease in accuracy relative to their more elaborate counterparts [109]. We scrutinized a range of mobile-optimized ML models to ascertain the best fit for our system. SSD-MobileNetV2 [216] stood out as one of the earliest trustworthy models tailored for mobile platforms. Among the contemporary one-stage models refined for mobile devices are variants of RT-

DETR [169], EfficientDet [239], and YOLO [120]. Our investigation encompassed a comprehensive evaluation of potential ML models suitable for real-time rip current detection using computer vision on mobile platforms.

6.2.2 Rip Current Detection with ML

Given its impact on public safety, the problem of automated rip current detection has been approached using various methods, some of which predate the emergence of deep learning techniques. For example, Philip et al. (2016) utilized optical flow on video sequences to discern the predominant flow towards the sea, aiding human observers in rip current detection [193]. Maryan et al. (2019) employed modified Haar cascade methods to detect rip currents from time-averaged images [174]. The concept of rip current detection via deep learning-based methods is not entirely new either. De Silva et al. (2021) were among the early adopters of deep learning methods for rip current detection, employing Faster R-CNN, a large model that achieved high accuracy [53]. They introduced a frame aggregation technique that bolstered detection accuracy for fixed-position cameras, but this technique was not suitable for moving cameras. Mori et al. (2022) offered a flow-based method to accentuate and depict rip currents for human observers [177]. However, this approach also demands a stationary camera and serves as a visualization tool rather than an automated detection system. In recent years, there have been several scholarly works about new deep learning model-based rip current detection techniques. For instance, Rashid et al. (2021) and Zhu et al.

(2022) presented RipDet [200] and YOLO-Rip [286], respectively. These lightweight rip current detection models, rooted in Tiny-YOLOv3 and YOLOv5s, belong to the smaller members of the YOLO family and are adept for environments with limited computational power. Rampal et al. (2022) showcased that the mobile-optimized, single-stage model SSD-MobileNetV2 can achieve performance metrics comparable to Faster R-CNN [199]. Furthermore, Dumitriu et al. (2023) explored and compared various iterations of YOLOv8 for rip current segmentation [64]. De Silva et al. (2023) unveiled RipViz, an innovation that examines 2D vector fields and interprets pathline behaviors to pinpoint rip currents [55]. Like that of Dumitriu et al. (2023), this method highlights the rip region's shape but identifies currents based on water movement rather than water appearance. Yet, while there is an assortment of effective rip current detection methods employing ML, a real-world application—such as a mobile app—primed for public safety and enhancing awareness for tangible societal impact remains elusive. This work endeavors to fill that void by devising a deployable mobile device-based real-time system for rip current detection.

6.3 System Design and Methods

6.3.1 System Architecture

Figure 6.1 presents an overview of the RipFinder system architecture. Our comprehensive system, designed to effectively identify and alert users of rip currents is

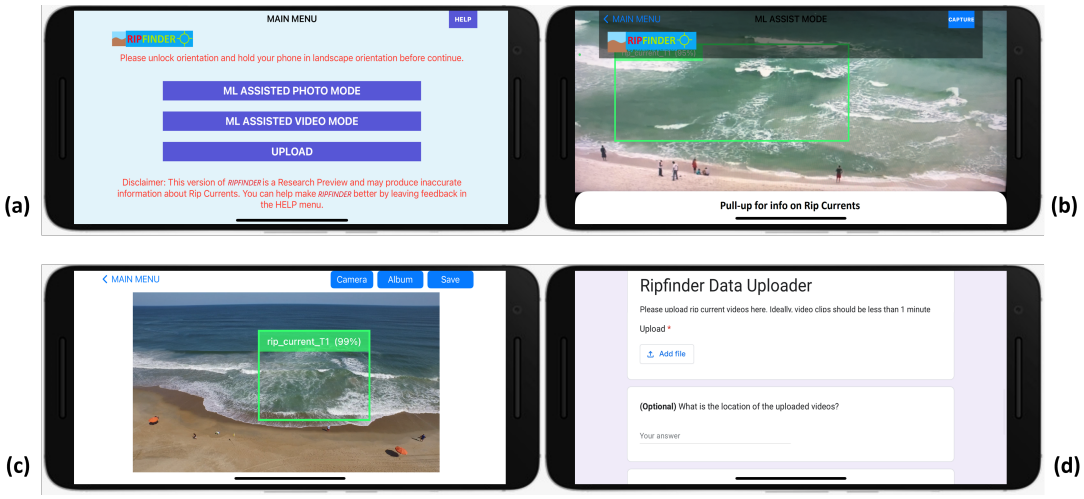


Figure 6.2: GUI of RipFinder App (a) Main menu, (b) Real-time detection from live camera view, (c) Detection from single image, (d) Data uploader for citizen science contribution.

organized into two primary components:

1. The client mobile app serves as the primary user interface. Within this app, we have integrated four ML models, each tailored specifically for mobile devices. As the device processes real-time visual input, these models evaluate the data and issue warnings if rip currents are detected. Depending on the device's processing power, the app can deploy either one or two ML models for detection. More modern devices with substantial resources can utilize two types of mobile-optimized ML models simultaneously, enhancing the reliability of rip current detection. In contrast, older devices with limited resources might default to a single model. Nevertheless, the ultimate decision to use one or two models rests with the user. When feasible, the app suggests users employ two models for optimal detection, but they retain the freedom to choose only one from the

available options if preferred.

2. Our system's server side employs complex ML models that demand significant computational resources and GPU capabilities, ensuring rip currents are detected with high accuracy. When a user captures an image or video via our mobile app, this data is sent to the server for in-depth analysis. After the server-side models process the data, the detection results are relayed back to the mobile app. Additionally, we offer the option to execute multiple models on the server, depending on its capabilities (number of CPUs and GPUs, system memory, etc.), enhancing reliability through redundancy.

Our system attempts to improve the reliability of rip current detection in a two-fold way. The use of two models enhances detection reliability on the client app, even though it demands more computational resources. Server-side models, being complex and larger, boast superior accuracy, thus ensuring that server-aided rip current detection is more reliable when internet access is available. The client-side model, meanwhile, operates using the on-device computational resources without the need for an internet connection. The results section further elaborates on the justification behind these two design choices. Thus, our system's design allows it to operate both online and offline.

Training datasets are essential for training both client-side and server-side ML models. We developed our dataset by utilizing the existing dataset from [53] and supplementing it with a large amount of our own data. Further details on the dataset and the ML model training process are explained in the implementation section.

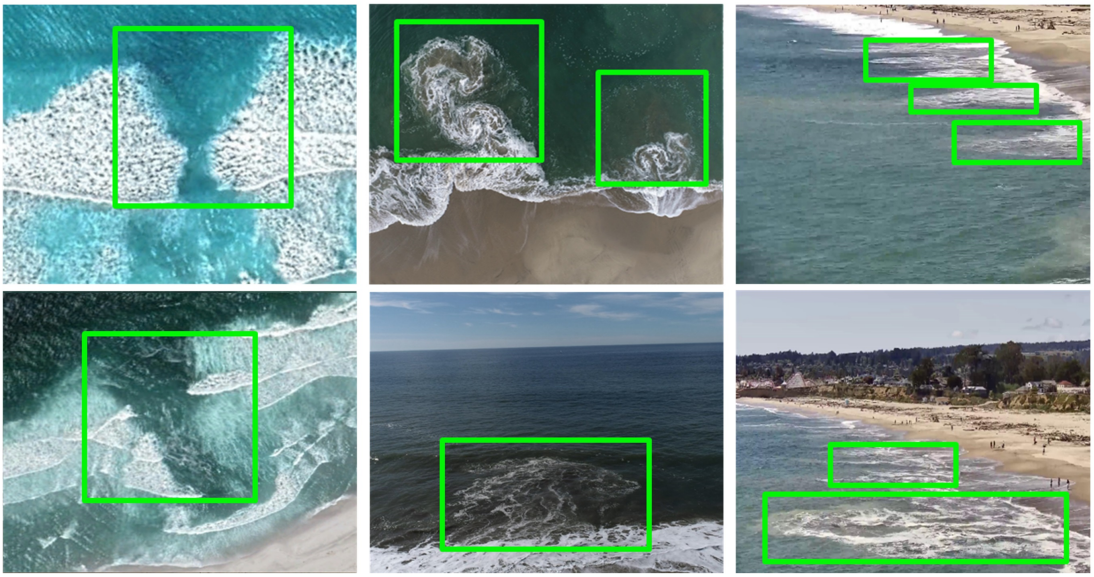


Figure 6.3: Some examples from our training dataset. The images from on the first column are from the dataset by [53]. The images on the second and third column are from the dataset we collected using a drone and a wireless rip activity monitoring camera respectively.

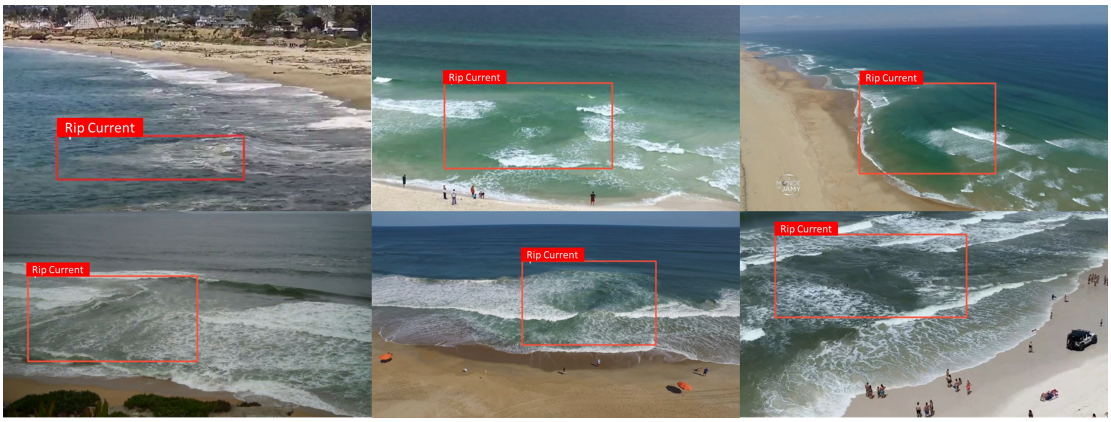


Figure 6.4: Some examples of detected rip currents from our test videos.

6.3.2 Mobile Apps

Figure 6.2 provides a visual representation of our mobile app's user interface, offering an intuitive, user-friendly environment. We created both an Android and iOS version of the mobile app. The application's design caters to a variety of user needs and includes the following features:

6.3.2.1 Live Camera and Visualization Tool.

The app offers a live camera feature to capture the seashore and serves as a real-time visualizer, placing bounding boxes around detected rip currents in the view, thus acting as an immediate warning system (Figure 6.2 (b)).

6.3.2.2 ML Model Selection.

From the in-app menu, users can choose the ML model for real-time rip current detection. On devices with higher computational resources, users have the option to turn on or off the use of two models in parallel for increased reliability.

6.3.2.3 Image and Video Recording.

The app enables real-time rip current detection and the recording of images and videos, letting users document and share potential rip currents with other beachgoers and rip current researchers.

6.3.2.4 Rip Current Detection Tool for Existing Images.

RipFinder app analyzes existing images on the phone to identify rip currents, offering retrospective insights to users (Figure 6.2 (c)).

6.3.2.5 Educational Resources.

Our app features an educational hub with resources on rip currents, accessible via a pull-up and help menu, ensuring users always have information at hand (Figure 6.2 (b)).

6.3.2.6 Data Upload Tool.

We integrated a data upload tool (Figure 6.2 (d)) for users to share geo-tagged rip current images and observations, fostering community collaboration and enhancing our dataset for improved algorithm refinement.

6.3.3 Client-side ML Models

In our application, RipFinder, we integrate several mobile-optimized ML models, all trained on a rip current dataset for client-side detection. These models have been tailored to ensure swift and efficient performance on mobile devices, which facilitates real-time rip current detection. The current version of RipFinder incorporates the following models:

6.3.3.1 YOLOv8n and YOLOv8m.

YOLOv8, the latest in the YOLO series known for fast object detection [119, 203], includes variants like YOLOv8n (nano) and YOLOv8m (medium) optimized for mobile devices. Its architecture facilitates single-pass detections, making it ideal for real-time applications such as rip current detection.

6.3.3.2 EfficientDet D0 and EfficientDet D2.

EfficientDet, known for its object detection prowess [239], has a unique scalable architecture that adjusts to computational resources, making it ideal for mobile use; it offers eight variants, D0 to D7, based on image size.

Of the four ML models at our disposal, the app selects one or two mobile-optimized models for rip current detection, contingent upon a device's computational prowess and internet connectivity. Modern, high-end devices employ two models, while the older, resource-constrained devices resort to just one. YOLOv8n and EfficientDet D0, due to their lesser computational demand, are ideally deployed as standalone models or in conjunction on dated or less competent mobile devices. In contrast, YOLOv8m and EfficientDet D2 are better aligned with newer devices boasting significant computational strength.

6.3.4 Server-side ML Models

Server-side, we engage a collection of high-performance ML models tailored for more resource-intensive computations. Given their demanding computational needs, these models are perfectly positioned for server-side deployment, capitalizing on robust hardware resources, including GPUs. For the server-side, we've selected:

6.3.4.1 YOLOv8l and YOLOv8x.

The YOLOv8 'l' (large) and 'x' (extra-large) variants [119] are more complex than their mobile-optimized versions, offering higher accuracy but requiring greater computational power, ideal for situations demanding utmost accuracy with ample resources.

6.3.4.2 RT-DETR (Real-Time Detection Transformer).

RT-DETR, a real-time adaptation of the DETR transformer-based object detection model [34, 169], maintains DETR's accuracy while ensuring faster performance. We trained its large and extra-large versions, RT-DETR-L and RT-DETR-X, for server-side use.

By leveraging these server-side models that can deliver high accuracy, we bolster the final verification of detected rip currents, reinforcing the reliability of our rip current detection tool.

Table 6.1: Comparison of the detection accuracy of the SOTA methods to select the best options for the client and server application.

Test Videos	**Client Side Models**						**Server Side Models**			
	EfficientDet			**YOLOv8**					**RT-DETR**	
	D0	**D1**	**D2**	**n**	**s**	**m**	**l**	**x**	**L**	**X**
Rip_test_video_1	1.00	1.00	1.00	0.94	0.72	0.99	0.99	0.93	1.00	1.00
Rip_test_video_2	0.99	0.86	1.00	0.01	0.01	0.05	0.20	0.05	1.00	0.99
Rip_test_video_3	0.86	0.84	0.79	0.58	0.30	0.71	0.46	0.53	0.90	0.93
Rip_test_video_4	0.27	0.79	0.72	0.00	0.00	0.04	0.00	0.00	0.85	0.89
Rip_test_video_5	0.73	0.91	1.00	0.76	0.50	1.00	1.00	1.00	1.00	1.00
Rip_test_video_6	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.86	1.00
Rip_test_video_7	0.99	1.00	1.00	0.19	0.35	0.93	1.00	1.00	1.00	1.00
Rip_test_video_8	0.70	0.71	0.71	0.00	0.00	0.00	0.15	0.29	0.76	0.80
Rip_test_video_9	1.00	1.00	1.00	0.21	0.24	0.62	0.71	0.63	1.00	1.00
Average Accuracy	0.73	0.79	0.91	0.30	0.24	0.48	0.50	0.49	0.93	0.96

6.4 Implementation

The various components of our system were implemented using the latest available technology.

6.4.1 Dataset

Our training dataset distinguishes between two types of rip currents based on their visual features. The first, termed **bathymetry-controlled rip**, is characterized by areas devoid of breaking waves, presenting as darker and calmer regions flanked by brighter waves. The second, known as **transient rip**, is identified by water discoloration due to sediment plumes that extend beyond the breaking waves. Though both classes represent rip currents, their visual features differ significantly. Detecting one type of rip current with an ML model trained on data from another type is unfeasible. Treating these two types as a single class compromises the effectiveness of the trained model. The label correlograms depicted in the figure illustrate the distinctions between the two classes based on the labeled regions of images from each class.

For the bathymetry-controlled rip current category, we utilized a dataset consisting of 1780 images made publicly available by [53]. For the transient rip current category, we curated a new dataset comprising 7565 labeled images. These were selectively extracted from videos captured by a drone, which focused on the visual signature of transient rip currents, and a Wi-Fi camera set up specifically for monitoring rip currents. We combined both datasets to train our model in the detection of the two rip current

types. This dataset was then divided in an 80:20 split for training and validation, with 80% allocated for training purposes and the remaining 20% used for validation. The efficacy of the trained models was assessed using a series of test videos. Figure 6.3 showcases a selection of images from our dataset.

6.4.2 ML Model Training

We conducted ML model training on an AWS cloud server equipped with 8 vCPUs, 61 GB of memory, and an NVIDIA Tesla V100 GPU boasting 16 GB of video memory. The EfficientDet models were trained using the TensorFlow library, while the YOLOv8 and RT-DETR models were trained with the Ultralytics library, which is based on PyTorch. All model trainings were initialized with a maximum of 500 epochs. For all versions of YOLOv8 and RT-DETR, a patience parameter of 50 was set. The patience parameter defines the number of epochs to wait before halting training via early stopping if there's no improvement in performance on a validation datasetSince the EfficientDet models do not allow for the definition of a patience parameter, we monitored convergence through TensorBoard and manually terminated the training once convergence was observed. All models converged within 300 epochs. We trained all models from scratch, instead of using transfer learning with MS COCO pretrained models from the ML libraries, to prevent negative transfer [266]. This decision was made because our rip current class data domain is distinct from any of the classes in the MS COCO2017 dataset [164].

Table 6.2: Comparison of ML Models: Performance Metrics and Resource Utilization.

ML Model Properties	EfficientDet D0	EfficientDet D2	YOLOv8 n	YOLOv8 m	YOLOv8 l	YOLOv8 x	RT-DETR-L	RT-DETR-X
Model Size on Server (MB)	13.70	18.50	6.00	49.60	83.60	130.40	63.00	129.00
Avg. FPS on Server	37	21	127	106	86	79	47	35
Model Size on Phone (MB)	4.23	7.04	6.00	49.60	83.60	130.40	63.00	129.00
Avg. FPS on iPhone 12 Pro	48	15	25	17	NA	NA	NA	NA
Avg. FPS on Pixel 6	26	8	29	18	NA	NA	NA	NA

6.4.3 Client Apps and Server

We developed the iOS version of the app in Swift using XCode, and wrote the Android version in Java with Android Studio. We tested the integrated mobile ML models on an iPhone 12 Pro and a Google Pixel 6. The server-side components were programmed in Python. We evaluated the server-side ML models on a desktop server equipped with a 16-core Intel Core i9 3.2 GHz CPU, 30 GB of memory, and an NVIDIA RTX3080 GPU boasting 10 GB of video memory.

6.5 Results

6.5.1 Performance Analysis of ML models

In this section, we present a performance analysis and comparison of state-of-the-art (SOTA) object detection models tailored for rip current detection. We compared ML models including EfficientDet D0, EfficientDet D1, EfficientDet D2, YOLOv8n,

YOLOv8s, YOLOv8m, YOLOv8l, YOLOv8x, RT-DETR-l, and RT-DETR-x. To gauge the accuracy of these models, we utilized nine test videos equipped with ground truth data. Four of these videos were selected for their relevance to our rip current detection objectives from the test set introduced by [53]. Additionally, three videos were drone-captured, while the last two originated from a wireless camera dedicated to rip current monitoring. Our accuracy assessment followed the methodology described in [53], where:

$$accuracy = \frac{correct\_labels}{total\_frames}$$

Frames were considered classified as correct if the detected bounding boxes had an Intersection over Union (IoU) score versus ground truth bounding boxes above 0.3. IoU is calculated as:

$$IoU = \frac{area\_of\_intersection}{area\_of\_union}$$

The comparison results are presented in Table 6.1 and some examples of detected rip currents are shown in Figure 6.4. Based on these results, we can justify the following two design choices we made.

6.5.1.1 Running two ML models to increase accuracy.

While running multiple models demands more computational resources, it enhances reliability. This design decision stems from the understanding that ML models

with varying architectures possess distinct strengths and shortcomings. Research by Mekhalfi et al. [175] indicates that models from the YOLO family tend to identify more objects, even if their precision varies. In contrast, EfficientDet provides more stable and accurate detection. In many cases, one of the models might not detect specific instances of rip currents, even if they were trained using the same data. For instance, although the rip current in "Rip test video 6" can be detected by EfficientDet D2, it isn't identified by any other mobile models. Thus, deploying two models ensures that a challenging-to-detect rip current is more likely to be detected on a more capable device. Additionally, since rip current detection pertains to safety, minimizing false negatives is more crucial than avoiding excessive false positives. Therefore, while employing two models might seem redundant for general applications, it is beneficial for the purpose of rip current detection.

6.5.1.2 Running ML models on both the client and server side.

More advanced and complex models, such as RT-DETR-L and RT-DETR-X, achieve higher accuracy but are limited to server execution. Thus, when an internet connection is available, server-assisted rip current detection becomes more reliable. The client-side models serves as the primary object detection mechanism, ensuring that rip current detection operates at the highest possible accuracy both with and without internet connectivity.

6.5.2 Evaluation and Selection

Among the ten (10) models highlighted in Table 6.1, we chose eight (8) for further evaluation. From the less accurate EfficientDet D0 and D1 variants, we selected only D0 because of smaller size. YOLOv8s was similarly excluded due to its poor accuracy. We evaluated the chosen models on a server equipped with a single GPU, an iPhone 12 Pro, and a Google Pixel 6 to determine the best-fit models for each platform. Our benchmarking of each model's performance focused on two primary metrics:

1. We evaluated the real-time responsiveness of each model by measuring the frames processed per second (FPS). This metric offers insights into the model's speed and its ability to detect rip currents in real-time scenarios. EfficientDet-D0 and YOLOv8n exhibited higher FPS on mobile devices, marking them as optimal choices for devices with limited computational capabilities. Meanwhile, the enhanced accuracy of EfficientDet-D2 positions it as a dependable option while still upholding real-time performance.

2. Each model's storage footprint need to be considered for embedding them in a mobile app, given that mobile devices have diverse storage capabilities and may also be running other apps simultaneously. Assessing a model's storage needs ensures that the application remains streamlined and does not overtax the device's memory. While the compactness of EfficientDet-D0 and YOLOv8n earmarks them as ideal for devices with resource constraints, the relatively small size and superior performance of EfficientDet-D2 make it a trustworthy option.

6.5.3 Model Performance Evaluation

EfficientDet-D0 and D2

- EfficientDet-D0: This model stood out for its high FPS, making it highly responsive on mobile devices. However, it occasionally struggled with identifying transient rip currents in complex backgrounds, leading to some false negatives.

- EfficientDet-D2: While slightly slower than D0, D2 provided higher accuracy, especially in distinguishing rip currents from similar-looking water patterns. This made it a more reliable option for detailed analysis despite its larger storage footprint.

YOLOv8 Variants

- YOLOv8n: This model demonstrated excellent real-time performance due to its compact size and speed. It was particularly adept at detecting well-defined rip currents but sometimes missed more subtle, transient currents.

- YOLOv8m: Balanced between speed and accuracy, this model handled both bathymetry-controlled and transient rips well. Its moderate storage requirements and consistent detection accuracy made it a strong candidate for mobile deployment.

- YOLOv8l and YOLOv8x: These larger models, used server-side, provided

superior accuracy, identifying even the faintest rip currents. However, their large size and computational demands limited their use to server environments.

- YOLOv8s: Excluded due to poor accuracy, especially in complex detection scenarios.

RT-DETR Variants

- RT-DETR-L and RT-DETR-X: These models, designed for server use, delivered high accuracy and reliability. They excelled in differentiating rip currents from similar patterns like wave shadows and sandbars. Their complex architecture, however, required substantial computational resources, justifying their server-side deployment.

6.5.4 Analysis for Client-Side Model Selection

Two Models vs. Three or More: The decision to use two models on the client-side was based on a balance between accuracy, computational resource demands, and processing time. While running more than two models could theoretically increase detection accuracy through model ensemble techniques, the incremental benefits were marginal compared to the substantial increase in resource consumption and latency.

- **Resource Consumption:** Each additional model significantly increased the server's CPU and GPU load, leading to higher operational costs and potential delays in processing, especially during peak usage times.

- **Latency:** Adding more models introduced additional latency, which could hinder the real-time aspect of rip current detection, a critical feature for user safety.

- **Redundancy and Reliability:** Using two models already provides a robust redundancy mechanism, ensuring reliable detection even if one model underperforms. The combination of YOLOv8l for broad detection and RT-DETR-L for detailed analysis offered a well-rounded approach.

Based on the comprehensive evaluation, EfficientDet-D2 and YOLOv8n were chosen for mobile deployment due to their balance of speed, accuracy, and compact size. For server-side implementation, YOLOv8l and RT-DETR-L were selected to maximize detection accuracy and reliability, ensuring that the system could effectively operate both online and offline.

Table 6.2 offers a comprehensive review of our findings from these cross-platform assessments. By juxtaposing results across platforms, we pinpointed models that meet both hardware constraints and app requirements to guarantee the most proficient rip current detection.

6.6 Conclusion

In this chapter, we introduce Ripfinder, a mobile app equipped with an ML-based computer vision tool designed to mitigate the safety hazards associated with rip currents, which are a leading cause of drownings globally. Ripfinder features

a sophisticated system that ensures rip current detection even in the absence of internet connectivity, making it indispensable in regions without lifeguards or reliable internet coverage. This capability is crucial for enhancing beach safety in remote and underserved areas.

Beyond its detection capabilities, Ripfinder enriches user knowledge with in-app informational content and videos about rip currents, helping users understand the dangers and how to avoid them. This educational component is vital for raising awareness and promoting safe behaviors at the beach. A standout feature of Ripfinder is its inclusion of citizen science. By inviting users to share data about identified rip currents, the app not only enhances scientific understanding but also fosters community engagement. This participatory approach leverages the collective efforts of users to contribute valuable data that can be used for further research and analysis, ultimately improving the overall understanding of rip current patterns and behaviors.

Ripfinder's integration of public safety, education, and scientific progress underscores its multifaceted approach to ensuring safer beach outings. By combining advanced technology with user engagement and educational resources, Ripfinder aims to create a comprehensive solution that addresses both immediate safety concerns and long-term scientific goals. The app exemplifies how modern technology can be harnessed to address real-world problems, making beaches safer and more enjoyable for everyone.

Chapter 7

RipScout: Realtime ML-Assisted Rip Current Detection and Automated Data Collection using UAS

This chapter presents RipScout, a system for real-time rip current detection and data collection using UAS or drones equipped with ML models. In contrast to RipSnap and RipFinder from the previous chapters, RipScout addresses more complex challenges such as deploying lightweight ML models suited for the limited computing resources of drones, ensuring stable data collection while in motion, and managing flight paths for optimal data coverage. Although the concept originated with RipSnap, designing an application for UAS involves significant optimization to meet constraints like real-time processing, limited power, and computational resources. UAS offer higher vantage

points, access to remote areas, versatile data collection angles, and real-time processing capabilities, enhancing the scope and quality of data collected. With RipScout, when a rip is detected along a flight path, the drone hovers in place and collects a video clip of a predefined length, followed by circling around the detected rip using pre-specified radii and heights to collect video samples from different vantage points and elevations. An important benefit is that the collection of rip current data can be performed by drone operators who are not familiar with rip currents, thereby reducing label noise. This integration of drones into the broader goals of this dissertation exemplifies the innovative use of advanced technologies to improve data collection efficiency and quality, aligning with the dissertation's emphasis on efficient data collection systems.

7.1 Introduction

This chapter introduces RipScout, an ML-assisted system designed for efficient field data collection of rip currents using drones. Rip currents are dangerous, fast-moving currents that can take people out to sea, leading to an estimated 100 drownings a year in the United States [86]. It is important to monitor beaches for the presence of these rip currents to improve beach safety. Data collected about rip currents are also important for validation studies of rip forecast models [65]. While traditional rip current data collection involves in-situ instrumentation such as GPS-equipped drifters and current meters [153, 170], recent work has demonstrated that rip currents can be detected from images and video [53, 55, 64, 174, 177, 193, 199, 200]. Since cameras are

more widely available than specialized instruments, these methods have the potential to greatly expand the amount of rip current data collected.

Uncrewed aerial vehicles (UAVs) such as drones have been widely used to perform visual data gathering tasks for environmental monitoring, search and rescue, and target tracking because they allow free-form flight maneuverability and thus versatile camera placement [19, 260]. Cameras on drones also have a distinct advantage of spotting rips from a higher elevation since some rips are difficult to spot from a lower elevation. This chapter investigates the question whether it is possible to create a system to detect and collect data about rip currents using drones that can perform as well as a domain expert e.g., lifeguard and/or as well as the leading rip detection algorithms today.

We first identify the technical challenges imposed by a mobile drone platform. These challenges include the ability to **detect rip currents in realtime** in order to support **field deployment**. The detection system must be **lightweight and run on limited computational resources** available on mobile platforms. Beaches are often remote with **no internet connectivity**, so sending images to a server for processing may not be possible. Given that battery capacity of consumer grade drones is typically in the 15-25 minute range per charge, data collection needs to be **efficient**. And to be of value, the rip detection must have **sufficient accuracy**.

We then designed a system that seeks to address these technical challenges. In our design, the mounted camera on the drone streams image data to a mobile phone. The mobile phone performs rip current detection and sends flight control signals

to the drone. When a rip current is detected, the drone is directed to take the desired data collection activity, such as recording 30 seconds of video from several different altitudes and/or video recording of the rip from different angles and elevations. Our proposed system architecture is practical and portable, leveraging the increasing availability of high-quality drones [176] and the ubiquity of smartphones. The system is described in System Architecture and Datasets Sections.

The system is then field tested and evaluated on how well it met its goal of accurate rip current detection and efficient data collection. To do so, we first narrowed down the choice for a base ML detection model (see Related Works Section). Those that required powerful Graphics Processing Units (GPUs) that are not available on mobile platforms were either removed from further consideration or modified to run on Central Processing Units (CPUs). There were three ML detection models that were evaluated and the best performing one selected for RipScout. Our tests show that the proposed system can perform as well as a domain expert in terms of accuracy and efficiency. As designed, the system is also agnostic to specific ML models as long as it can run within the resource constraints. Hence, the ML model employed for rip detection can be replaced with a better one in the future. The field tests, selection, and evaluations are described in the Field Testing and Evaluation Sections.

- The primary contribution of this chapter is a drone-based system architecture called RipScout that allows efficient rip current data detection and data collection in a resource-constrained environment.

- The rip current data that were collected during field experiments are valuable for further rip current research.

7.2 Related Works

7.2.1 Lightweight ML Models for Drones

We review recent literature to look for candidate ML models that are potential candidates for use in rip current detection. A series of VisDrone challenges [33, 63, 75, 287] have focused on improving the accuracy and speed of object detection and tracking. The deep learning models in those papers can be divided into one-stage and two-stage structures. Recent trends in the VisDrone-DET 2019-21 challenges [33,63,75] show that one-stage models prioritize prediction speed in frames per second (FPS), while two-stage models strive for the best per-frame accuracy but incur higher computational costs.

The two-stage region-based detectors, such as Faster R-CNN [205] (Region Convolutional Neural Network), typically include a region-proposal stage followed by a classifier. However, one major drawback of these detectors is their computational complexity, which makes them challenging to deploy on embedded platforms. For instance, Faster R-CNN has been successfully used for realtime object detection with drones by connecting to a remote GPU server [156]. However, this approach is not practical in locations without internet connectivity.

In contrast, one-stage techniques employ a single CNN to perform end-to-end object detection [167]. This single CNN architecture render the models lightweight and able to run quickly on mobile devices, making it an ideal choice for realtime object detection applications. The main trade-off for this speed is a decrease in accuracy compared to more complex models [109]. Sun et al. [235] demonstrated that single CNN architectures designed for mobile devices also require the least amount of memory compared to other CNNs.

As such, we focus our review on the latest one-stage ML models to determine suitable candidates for our system. We found SSD-MobileNetV2 [216] as one of the first reliable models that work well for mobile systems. However, the top choices from the most recent one-stage models optimized for mobile devices are EfficientDet [239], and You Only Look Once (YOLO) [120]. Detection Transformers (DETR) [34] was also briefly considered but was found to have a significantly slower inference speed and require a server with GPU. A comparative study by Mekhalfi et al. [175] demonstrated that EfficientDet is the more stable and offers good generalization capability for applications with aerial imagery. Even though the newest YOLOv8 (2023) improved over its predecessor in accuracy, EfficientDet still provides more stability while achieving similar or higher accuracy [243].

7.2.2 Field Tested Drone Applications with ML

Field testing is needed to ensure that results from experiments in controlled lab settings transfer and actually perform in real-world scenarios. In many drone applications, it is crucial to have a model that can process and predict data quickly, approaching the speed at which the drone transmits video. For example, the DJI Phantom Pro 4 V2.0 transmits images at 30 frames per second [61]. However, few research papers address the challenge of embedding models in real-world settings [37, 288]. For example, while frame accuracy of an ML model may appear very good when compared against known ground truths, deployment in real-world where other factors such as fog, haze, glare, vibrations, etc., come into play may be quite different.

Although some studies [182, 274] do present empirical studies on embedded UAVs to perform realtime visual object detection and tracking, they have not been validated through field tests, which are crucial for evaluating the performance of such systems in real-world environments.

In recent times, several projects have utilized ML models to process drone images. However, these projects vary in terms of where the processing takes place. For instance, Rohan et al. [212] presented a drone-based system that employed a lightweight SSD model on a desktop system equipped with a high-end GPU. Nevertheless, their system is not feasible for field deployment, and they only conducted tests indoors. On the other hand, Deng et al. [58] deployed a similar lightweight model on the Nvidia Jetson TX2 Board, a specialized embedded platform that can be mounted on a drone. While this

platform offers suitable performance, it is not as widely accessible as mobile phones, thus limiting its user base. Similarly, another project utilized the same GPU board for a UAV-based warning system with realtime object detection [244]. However, both of these projects did not conduct real-world field deployment and instead tested their system using existing drone footage.

Vajgl et al. [256] described a pattern-matching algorithm for drone control that runs on a mobile device and tested it to detect simple objects in a lab setting. None of these projects attempted to perform complex tasks such as realtime object detection for data collection in real-world settings. The differences in processing locations and the limitations in real-world deployment highlight the need for a more versatile and accessible approach to drone image processing and object detection.

In this chapter, we evaluated our proposed system with several field tests described in the Field Testing Section.

7.2.3 Rip Current Detection with ML

There has been a growing interest in remote detection of rip currents using images and/or video in recent years. We review and organize the literature by how suitable the approach might be for deployment on a mobile platform. In particular, we consider if images/videos from a stable platform are required and whether the ML model requires significant computing power i.e., GPU.

The problem of rip current detection has been addressed using various methods,

including approaches predating the emergence of deep learning techniques. Philip et al. [193] employed optical flow on video sequences to identify the predominant flow towards the sea, assisting human observers in rip current detection. Maryan et al. [174] utilized modified Haar cascade methods to detect rip currents from time-averaged images. de Silva et al. [53] were among the early adopters of deep learning methods for rip current detection, employing Faster R-CNN, a two-stage model that achieved high accuracy. They also proposed a frame aggregation technique that improved detection accuracy for fixed-position cameras, which is not applicable to moving cameras like those mounted on drones. [177] proposed a non-ML, flow-based method for highlighting and visualizing rip currents by showing the behavior of a set of virtual buoys (pathlines and timelines). Similarly, RipViz [55] analyzes 2D vector fields and uses ML to learn pathline behavior and automatically identify rip currents. This method highlights the **shape** of the rip region. Detection is based on water movement behavior rather than on the appearance of the water state. The methods above all require video from a stable platform and hence are not directly applicable for analyzing drone videos.

Rashid et al. [200] and Zhu et al. [286] introduced RipDet and YOLO-Rip, lightweight rip current detection models based on Tiny-YOLOv3 and YOLOv5s respectively. These are smaller models in the YOLO family and well-suited for running with limited computational resources. These models have since been superceded by YOLOv8. In fact, [64] utilized and compared various versions of YOLOv8 for rip

current segmentation. While YOLOv8 can theoretically run in realtime on both desktop and mobile system, Dumitriu et al. [64] did not discuss an implementation on a mobile system. Rampal et al [199] demonstrated that the mobile-optimized single-stage model SSD-MobileNetV2 can achieve comparable accuracy to Faster R-CNN. However, they used an Nvidia P100 GPU which has a maximum power draw of 250W and has a large form factor and therefore cannot be used in any smartphone. Further investigation revealed that it is possible to implement MobileNetV2 to run on a CPU with real-time results.

The models investigated by these latter sets of papers as well as the findings by Mekhalfi et al. [175] narrowed our field to three candidate models namely: SSD-MobileNetV2, YOLOv8, and EfficientDet D2. In the Evaluation Section, we present comparisons among these models.

7.3 System Architecture

Our objective is to design a drone-based system capable of realtime rip current detection and efficient data collection. The high-level architecture of our system is presented in Figure 7.1(left), which includes specific hardware components, as well as software components. This section provides a detailed discussion of the necessary hardware components required for our system, such as the drone and mobile device, and the critical software components, including the mobile application and ML model utilized for rip current detection.

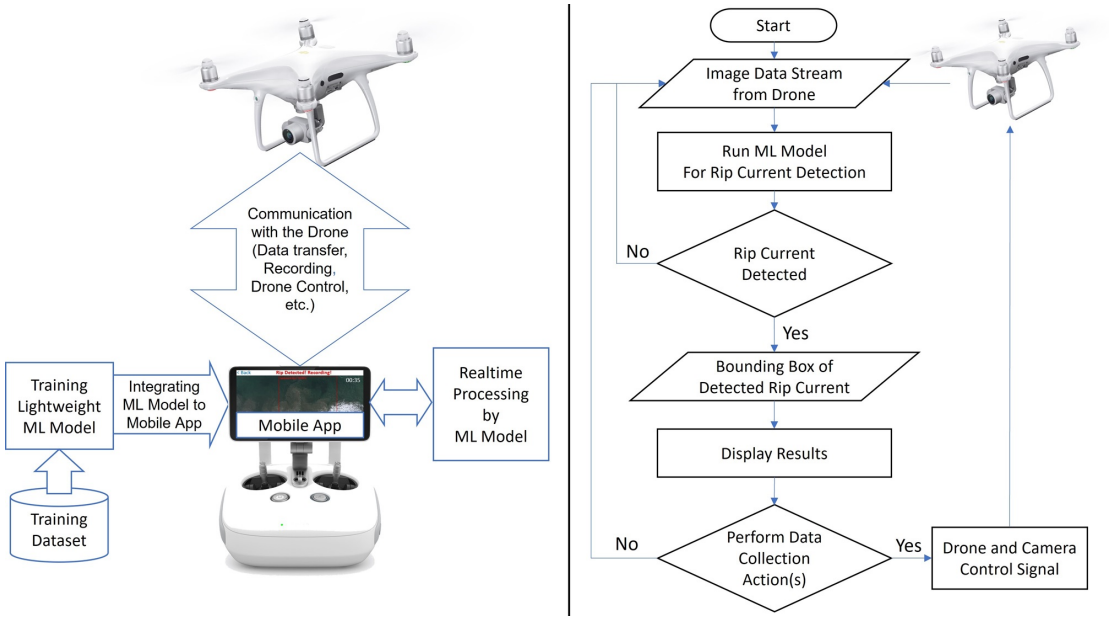


Figure 7.1: (Left) The high level architecture of the realtime ML-assisted data collection system using RipScout, (Right) Diagram of per frame processing by the mobile app.

7.3.1 Devices and Hardware

We have two main hardware components: a drone with an integrated camera and an associated physical controller, and a mobile device with a touchscreen display physically connected to the controller using a data cable.

7.3.1.1 Drone Hardware Selection

We used a DJI Phantom 4 Pro V2.0 quadcopter, a relatively low-cost professional drone. This model was selected because of our design goal to be **widely deployable**. DJI has the largest share of the drone market in the world and offers a wide range of models suitable for various purposes [61, 281]. The specific model we chose has many advanced features useful for conducting data collection missions, such as improved

electronic speed controllers, expanded flight autonomy, and obstacle avoidance using a multicamera and infrared sensing system.

It has an onboard gimbal camera featuring a 1-inch 20MP CMOS sensor and a mechanical shutter, eliminating rolling shutter distortion. The camera has a 70-degree field of view, making it a good choice for computer vision tasks.

The drone is controlled by a physical remote controller that receives a live video feed from the drone camera at a maximum resolution of 1920×1080 and 30 FPS [61]. We use this feed as input for our ML model, while the camera records 4k resolution videos to an onboard microSD card for later scientific analysis. While we showcase our work using the DJI Phantom 4 Pro V2.0, it is important to note that our system is built using the DJI Mobile SDK (Software Development Kit) [61], making it compatible with all DJI drones. Additionally, our code is open-source and can be adapted to work with any drone that provides similar functionalities to the DJI Mobile SDK. This makes our system highly adaptable and customizable.

7.3.1.2 Mobile Devices

The physical drone controller has the option of using a smartphone or a tablet to show the live camera feed and other system status. Either of these options would satisfy our need for a **portable** system. However, we also require **realtime** and **accurate** rip current detection. Thus, we opted to use a smartphone as it provides adequate processing power to run the ML models and process videos from drone camera in

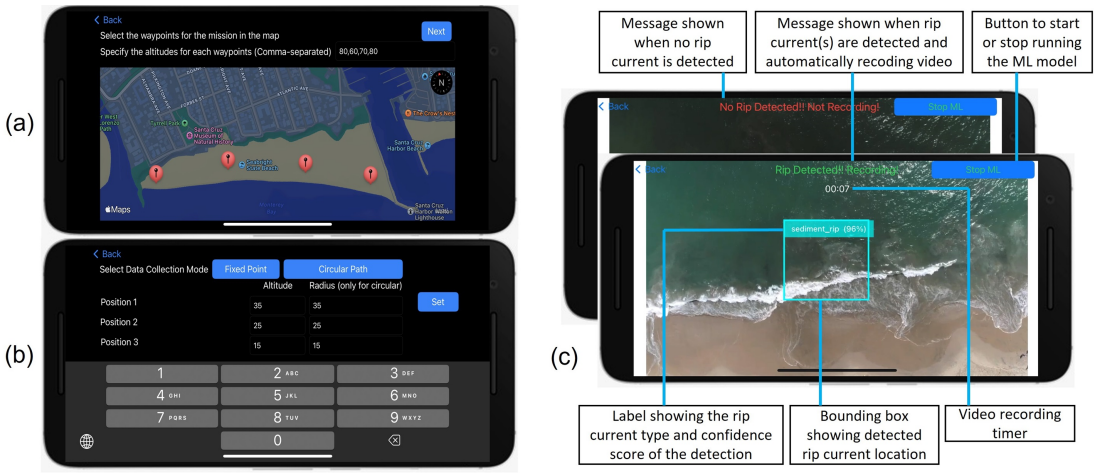


Figure 7.2: Graphical user interface for planning data collection mission. (a) The interactive map interface for selecting flight plans. (b) The interface to define data collection action plan. (c) Visualization of detected rip current from the live video feed.

realtime. We tested our system with an iPhone 12 Pro. While much less powerful than a desktop workstation or server, the A14 Bionic chip on the iPhone 12 Pro is relatively powerful for a mobile device.

One alternate design choice would have been to deploy ML rip current detection directly on the drone. However, one of the major constraints of using drones is the short flight time due to the limited battery power. For the drone we used, each battery lasts a maximum of 30 minutes [61] with 15-25 minutes being a more realistic figure. Delegating detection to a separate mobile device prevents the ML module from depleting battery power and reducing flight time.

7.3.2 Software Components

We implemented the software components of our system as a mobile application. This app includes the graphical user interface (GUI), ML model for rip current detection, and drone control and communication.

The app receives the live video feed from the drone through the DJI Mobile SDK. The video feed serves as input for the ML detection module, which produces detection output in the form of bounding boxes. Upon detecting rip currents, the application initiates specific drone control commands to capture relevant scientific data. Figure 7.1 (right) shows the flowchart of the software process.

7.3.2.1 The Drone App

The system functionality is presented to the user through a GUI. The GUI supports different aspects of a field data collection mission, including flight planning, providing feedback to the operator when a rip is detected, and specifying data collection actions to take when a rip is detected. There is also a button to enable or disable the background ML process when object detection is not necessary, such as during take-off, landing, or return to home. Figure 7.2 shows the GUI of the RipScout app.

The flight planning screen features an interactive map that allows the user to define the flight path by long-pressing on the map and dropping pins to mark waypoints. Additionally, the user can specify the altitude for each waypoint on this screen. Moving to the next screen, the user is presented with options to select the type of data collection

action to be performed when a rip current is detected. These actions may include recording videos of predefined lengths from one or more fixed altitudes, capturing footage from different camera angles along circular paths around the detected rip current, or a combination of these options for each detected rip current. Upon starting the mission on the subsequent screen, the user is provided with a live camera feed. Throughout the flight, if a rip current is detected, the locations of these rip currents are indicated by bounding boxes displayed on the live video feed. Furthermore, this screen also shows the status of the ongoing intelligent data collection activities.

Collecting data from various altitudes, angles, and viewpoints of a rip current through circular flights involves three distinct processes. First, the ML model running on the connected mobile device needs to detect the rip current. Second, once the location is identified, the mobile app sends the drone controller the circular flight path information with the detected rip current location at the center, along with user-defined parameters for radius and altitude. Third, as the camera operates independently with its singular gimbal axis, a separate set of camera control data is transmitted to the drone to specify the camera angle and drone heading, ensuring it captures video footage in the correct direction. Note that although apps such as DJI Go and other third-party apps offer features such as "follow me" or "circle around," these functionalities are primarily designed for common objects such as cars, bicycles, or individuals. Detection and tracking of rip currents are not supported.

7.3.2.2 ML for Rip Current Detection

The rip detector is a separate plug and play module where different ML models can be substituted. We compared three such models in the Evaluation. Video from the drone is transmitted as a one-dimensional array and converted to image frame buffers as input for the CNN of these ML models. Fig 7.3 shows a general architecture of these models. High level features extracted from the CNN are further processed to generate the detection results i.e., bounding boxes, class labels, and confidence scores. An overlay image is then generated using the detection results and superimposed over the input image to generate the output image.

7.4 Datasets

The datasets associated with this work can be classified into two distinct categories: (1) the initial dataset manually collected for training the rip current detection models, and (2) the data automatically collected by RipScout for subsequent scientific analysis and various other applications.

7.4.1 Training Data

There are several mechanisms that give rise to rip currents resulting in different types of rip currents [35]. For an ML model to detect rip currents, it needs to be trained with many different examples of their visual signature. The models we investigated

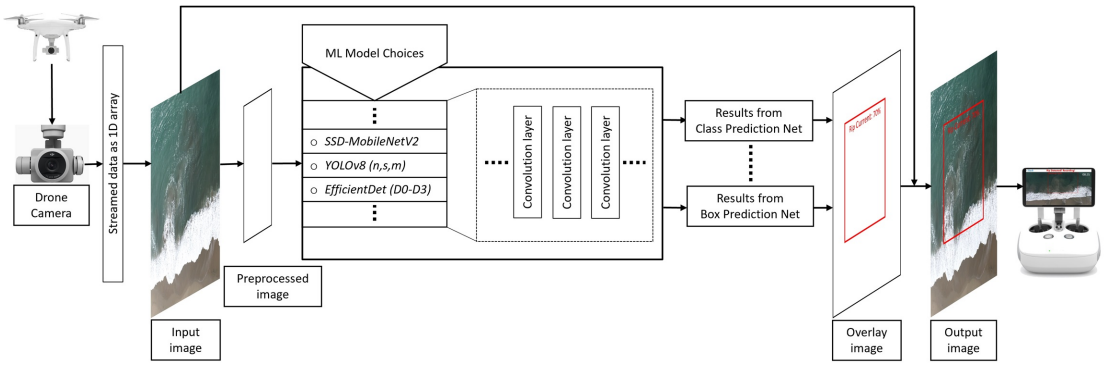


Figure 7.3: Pipeline showing how drone video is processed by (choice of) ML model to generate detections in realtime.

were designed to detect two types of rip currents. Distinction between rip current types is based on visual appearance, which in itself does not completely indicate the underlying morphological or hydrodynamic mechanisms causing a rip current. The first type considered in this chapter is visually characterized by a darker, calmer region of water flanked by brighter breaking waves [53]. We shall refer to this type of rip as **channel rips**. This type of rips are typically associated with bathymetry-controlled rip currents [35]. The second type is visually characterized by discolorations caused by sediment-laden water as they are carried away from shore, often forming a plume that extends beyond the breaking waves. We shall refer to this type of rip as **sediment rips**. Sediment rips are typically associated with transient or hydrodynamically controlled rip currents [35]. It is not the goal of this chapter to classify detected rips based on how a rip was formed but rather how the rip appears visually. It is also important to note that there are rips that have different visual characteristics aside from the two that are studied in this work. For example, presence of rips can be indicated by white foam or water

heading offshore. Taking spatial context into consideration, rips can also be indicated based on the wave direction and water texture next to natural and man-made structures. These other types of rips are not considered in this work. In terms of performance of ML detection, each type of rip is considered a different class. Usually, accuracy will drop slightly as more classes are considered. Hence, having more than one class allows us to study how well different ML models perform when there is one, or when there are two classes.

To ensure that our model could accurately detect both types of rip currents, we recognized the need for a separate dataset for sediment rips, in addition to the existing dataset of channel rip images from [53]. As shown in Figure 7.4, there are significant visual differences between these two types of rips. These differences are so pronounced that a model trained on one type of rip data would be unable to detect the other type.

However, obtaining data for sediment rips is a challenging task, and no existing datasets are available. Hence, we collected a new dataset of sediment rip images by manually flying our drone over the water surface along the shoreline and recording videos. This involved capturing data at different times of day and in various weather conditions, as the appearance of sediment plumes can vary depending on environmental factors. For creating the dataset, we extracted one frame per second from the drone video and manually labeled the frames that contained sediment rip currents. As the visual characteristics of a sediment rip involve water discoloration due to sediment plumes, we identified and labeled the frames that have this visual signature. In some

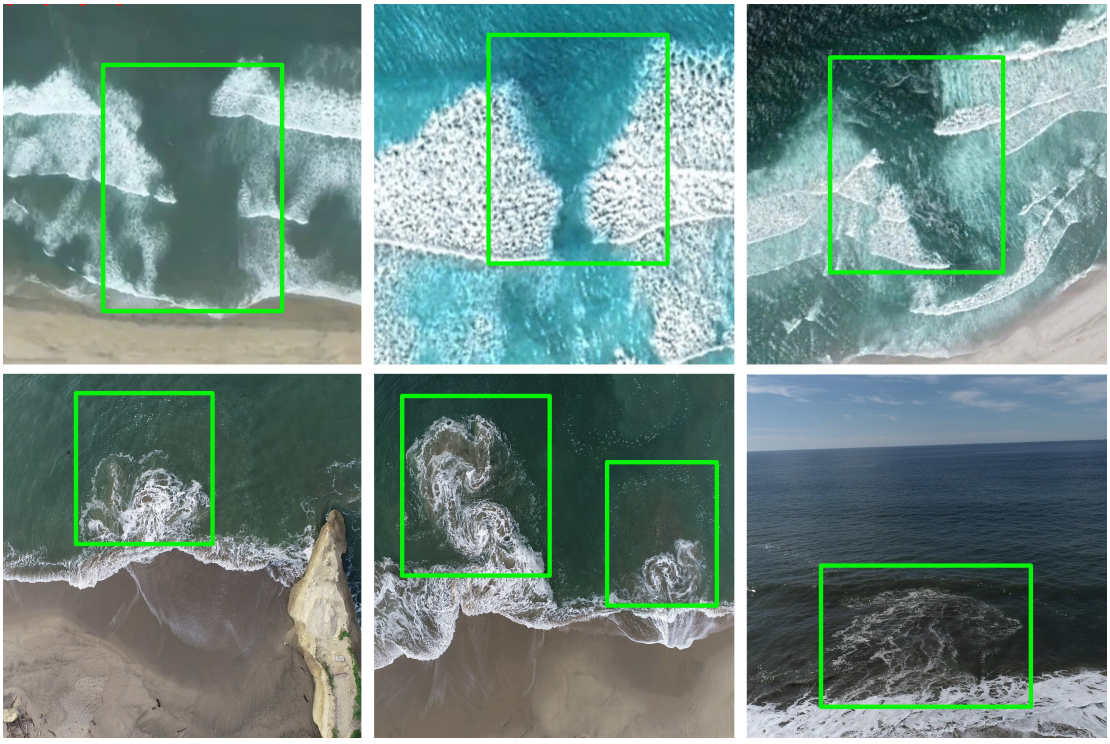


Figure 7.4: Some examples labeled images from the dataset with two types of rip currents: channel rips (top three images) and sediment rips (bottom three images). These images demonstrate the distinct visual characteristics of each type, highlighting the need for separate datasets to train an accurate model.

cases, a single frame contained multiple sediment rips, which we labeled accordingly (Figure 7.4 top three images).

We generated a new dataset of 2555 labeled images selectively extracted from 73 videos collected using the drone that captured the visual signature of sediment rip currents. This dataset was combined with the existing dataset of 1780 channel rip images from de Silva et al. [53] to train our model to detect both types of rip currents. The dataset was divided into an 80:20 ratio for training and testing, with 80% of the images allocated for training and 20% for testing. This split ratio is commonly used in ML [124]. Examples of the resulting datasets are displayed in Figure 7.4.

Figure 7.5: Sediment rips are more obvious from a higher elevation (left) than lower elevation (right).

7.4.2 Automated Data Collection

While channel rips can be identified by darker channels of calm water flanked by breaking waves even from ground level, sediment rips are much harder to identify from a lower elevation (see Figure 7.5). An ML model trained with the manually collected

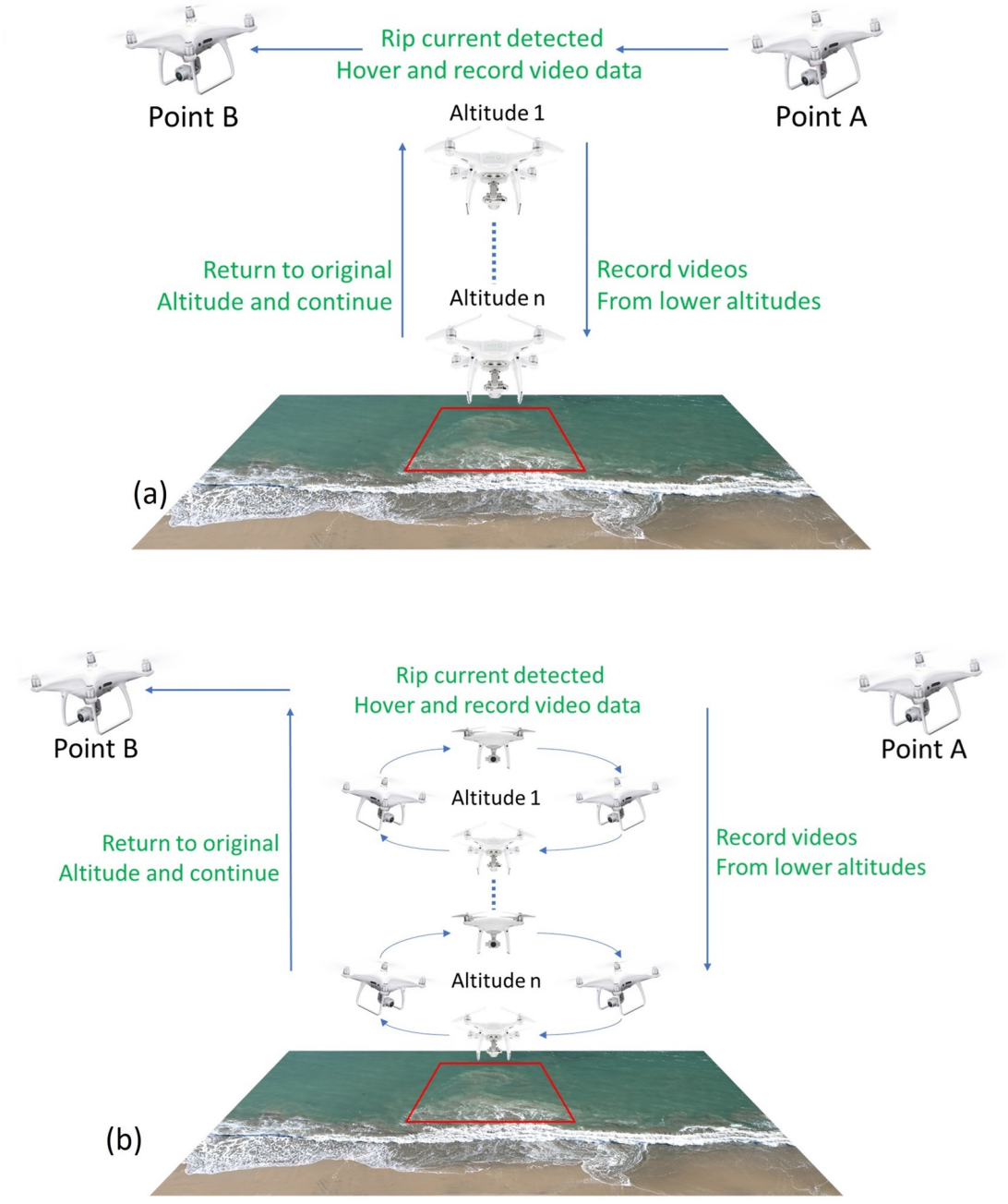


Figure 7.6: When RipScout detects a rip between two waypoints, it can be programmed to perform a combination of data collection actions such as (a) hover in place and record a video of pre-specified duration, or go to user specified height before hovering in place and recording the video, then resuming flight from its original height, (b) record a video from user specified radius and height with the detection location as the center.

sediment rip data described above was used with RipScout to automatically collect new data of sediment rips from different angles and elevations (see Figure 7.6). Such a dataset not only serves purposes for beach monitoring, lifeguard support, and validating rip forecast models, e.g. Dusek and Seim's model [65], but can also be utilized to train models capable of detecting rips observed from lower elevations. Models that are trained on datasets with multiple vantage points can be employed for rip current detection on lower elevation platforms such as web cameras and smartphones.

7.5 Field Testing

We collected data and conducted extensive field tests of our system on six public beaches in California. Our field tests were meant to validate the suitability of RipScout for automated data collection. Each field test consisted of multiple drone flights, all of which were performed by a licensed drone pilot certified by the Federal Aviation Administration (FAA). We obtained permits from federal and state authorities, including the California State Park System and the Monterey Bay Marine Sanctuary, to ensure compliance with all relevant regulations. Over a period of 18 months, from January 2022 to June 2023, we performed a total of 36 drone flight missions to collect training data and test our system in real-world conditions. All flights were conducted in strict adherence to FAA rules and guidance [73]. Our field test received approval from the Office of Research Compliance Administration at the University of California, Santa Cruz, as an exempt Human Ethics Study and was conducted in accordance with

the approved guidelines.

Figure 7.6 illustrates a typical data collection scenario when using the system in the field. The drone flies from Point A to Point B following a predefined flight plan. During the flight, the ML model runs on the mobile device connected to the controller and processes the live video feed from the drone camera. If a rip current is detected, the drone automatically stops and performs a data collection activity, such as recording a video clip for 30 seconds at several altitudes or flying in a circle to capture the rip from multiple angles. When at least one rip current is present, the app is instructed to send control information to the drone to capture a high-quality 4K video of a predefined length and save it on the onboard microSD card. The quality of this recorded video is much higher than the quality of the live feed used for realtime ML processing, as it is intended for further analysis, research, and archival purposes. The location of the rip current is shown on the app's preview screen using bounding boxes. Once the data capture is completed, the drone returns to its original flight plan. The drone pilot always has full control over the drone and can manually override such programmed behavior at any time.

For four of our field tests, we explicitly compared the performance of our ML-assisted system with the performance of drone pilots on an unaugmented drone. A total of ten drone pilots, who are not experts in rip current detection, participated in these comparison trials. The 10 participants were recruited by snowball sampling through our network in academia, and informed consent was obtained from them. The field

tests were performed at a state beach in California where sediment rip currents are prevalent to provide enough data samples for comparison. In each case, days and times were decided based on weather, tidal, and wave conditions in which rip currents were likely to be present. Before the field trips, participants were provided with tutorials on how to detect rip currents. The drone's flight path was preprogrammed to fly through a set of waypoints at a predefined altitude overlooking the shoreline. At the beginning of each field test, an initial flight pass was performed to collect an overview of the location for future review by rip current experts. The next round of flight followed the same path while running RipScout. Guided by the ML, every time a rip current was detected, the drone stopped and recorded a high-resolution video. In subsequent rounds, participants flew the drone manually through the same trajectory, and every time they visually determined the presence of a rip current, they manually triggered recording a high-resolution video. RipScout and participants were compared in terms of accuracy in locating rip currents, as well as the rate at which data samples could be collected.

7.6 Evaluation

We analyze two critical aspects of our system. First, we compare the accuracy, processing speed, and resource utilization of three ML models for rip current detection. Second, we compare the efficiency of our system for collecting rip current data via a series of field tests using drones with and without ML support.

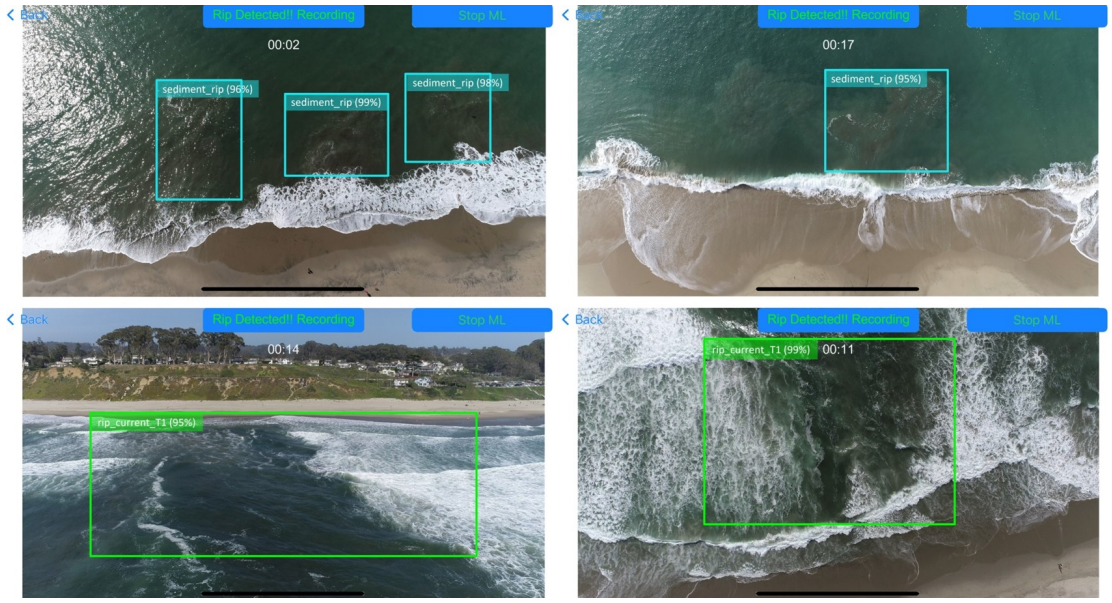


Figure 7.7: Here are some examples of realtime automatic detections of two types of rip currents using RipScout: sediment rips (top row) and channel rips (bottom row). As shown in the top-left image, RipScout can detect multiple rip currents in a single frame. The two images in the bottom row demonstrate that RipScout can detect rip currents from both side and top views.

7.6.1 ML Model Performance

Table 7.1: Comparison of detection accuracy, model size, and processing speed of three ML models when trained on a single class: either channel rip or sediment rip.

ML Model	**SSD-MobileNetV2**	**YOLOv8m**	**EfficientDet D2**
Accuracy (Channel rip)	82.1%	87.3%	94.8%
Accuracy (Sediment rip)	76.06%	92.6%	92.9%
Saved weight of ML model (Megabytes)	4.5	49.6	7.0
Training time (Channel rip)	10 hours	4 hours	5 hours
Training time (Sediment rip)	9 hours	3.5 hours	5 hours
Processing speed in frame per second	33	25	17

We analyze the detection accuracy of the three models when tasked with detecting one class of object only (either channel rips or sediment rips) in Table 7.1. We also analyzed the impact on detection accuracy when the three models are tasked with detecting rips from one of two classes in Table 7.2. For all comparisons, models were trained on a desktop with a GPU, and tested on CPU-based implementation running on an iPhone 12 Pro. Conversion of an implementation using a GPU to a CPU implementation to run on a smartphone involves conversion to a FlatBuffer format [94]. All the detection accuracy figures are based on our implementation of the three ML models. Training and testing were done on the same dataset described in the Datasets Section.

Table 7.2: Comparison of detection accuracy, model size, and processing speed of three ML models when trained to detect and distinguish between two classes: either channel rip or sediment rip.

ML Model	**SSD-MobileNetV2**	**YOLOv8m**	**EfficientDet D2**
Average Accuracy	78.45%	88.3%	93.1%
Accuracy (Channel rip)	80.83%	84.8%	93.6%
Accuracy (Sediment rip)	76.06%	91.8%	92.6%
Saved weight of ML model (Megabytes)	4.5	49.6	7.0
Training time	14 hours	5 hours	8 hours
Processing speed in frame per second	33	25	17

We were not able to exactly replicate the detection accuracy for SSD-MobileNetV2 as reported by Rampal et al. [199], likely due to the conversion to CPU implementation and differences in dataset (no code or data were shared). Likewise, the detection accuracy for Dumitriu et al. [64] are different as they used YOLOv8 for segmentation while we used it for detection. Additionally, a different accuracy metric was used. Among the five variations of YOLOv8, we selected YOLOv8m based on the benchmark performed by Dumitriu et al. [64]. There are eight variants of EfficientDet to select from, starting with EfficientDet D0 with expected input size 512×512 to EfficientDet D7 with expected input size 1536×1536. We selected EfficientDet D2 as it takes 768×768 as input, the closest to the 1280×720 resolution video transmitted from the drone while providing realtime processing speed.

Looking at Table 7.1, we see that EfficientDet D2 has the highest detection accuracy for detecting channel rips (when trained with channel rip dataset). EfficientDet D2 also narrowly edged out YOLOv8m in detecting sediment rips (when trained with sediment rip dataset). Note that model sizes are all reasonably small to run on most smartphone. EfficientDet D2 runs approximately about half as fast as SSD-MobileNetV2, but is still acceptable for realtime detection when every other frame from the drone is dropped.

As expected, all three methods incurred a slight drop in detection accuracy when a second class of objects was added (see Table 7.2). This can be seen when comparing the detection accuracy for channel rips amongst the three models in Tables 7.1 and 7.2. The same is true for sediment rips. The only exception is SSD-MobileNetV2 for sediment rips where accuracy remains unchanged. Since each of the three different models were presented with an equal number of test cases for channel and sediment rips, the average accuracy is the simple average of the detection accuracy for each type of rip. The top performer in terms of accuracy EfficientDet D2. Its model size and processing speed are within the requirements for a lightweight mobile app and realtime processing.

Training times are included for completeness. Training was performed using TensorFlow on a desktop machine equipped with an Intel Core(TM) i7 2.60 GHz microprocessor, 16 Gigabyte main memory, and an Nvidia RTX 2070 GPU with 8 Gigabyte video memory. After completing the training, we converted the models into the optimized FlatBuffer format for integration into both iOS and Android apps.

Figure 7.7 showcases the realtime detection of the two types of rip currents by

RipScout fitted with an EfficientDet D2 detector. Sediment rips are highlighted in cyan bounding boxes while channel rips are highlighted in green bounding boxes. RipScout can detect multiple rip currents in a single frame, as illustrated by the presence of multiple bounding boxes.

The tables present the accuracy of rip current detection on the test dataset and ground truth published by de Silva et al. [53], along with additional video clips containing sediment rips that we collected using the drone. Accuracy is calculated using the same method as de Silva et al. [53], where:

$$accuracy = \frac{correct\_labels}{total\_frames}$$

Frames were considered classified as correct if the detected bounding boxes had an Intersection over Union (IoU) score versus ground truth bounding boxes above 0.3. IoU is calculated as:

$$IoU = \frac{area\_of\_intersection}{area\_of\_union}$$

Apart from evaluating the accuracy of each model, we also assessed their memory storage requirements and average processing time per frame.

Our findings indicate that all three models are suitable for realtime processing and have memory requirements close to the average iOS size of 35MB.

Table 7.3: Field test comparison of rip current detection efficiency using drones with (RipScout) vs without (human only) the aid of ML.

Field Test	**Average Time to Capture Each Rip Current (Seconds)**		**Data Collection Speed Improved (Times faster)**
	Human User	**RipScout**	
1	118.50	29.60	4.00
2	148.17	34.00	4.36
3	102.71	32.00	3.21
4	179.60	42.00	4.28
Overall	137.24	34.40	3.99

7.6.2 Efficiency of RipScout

In this section, we compare the efficiency of data collection using RipScout fitted with EfficientDet D2 to traditional data collection involving human participants without ML assistance. Our comparison is based on several field tests, each involving multiple drone flights as described in the Field Testing Section. We only considered data from a field test if there were more than one rip current present at that time. Table 7.3 presents a summary and comparison of the average time required to capture each rip current by RipScout and human participants. Our results show that RipScout had a significantly lower average time of 34.40 seconds to capture a rip current compared to 137.24 seconds for human participants. Additionally, the overall flight time required by human participants was approximately four times greater than that of RipScout.

This finding indicates that the use of RipScout can provide more coverage and/or extended monitoring. This is noteworthy given that the flight time of each battery is approximately 15-25 minutes depending on weather conditions and operational use (e.g. whether recording or not).

While our field tests utilized the EfficientDet D2 model, one could also replace it with other models such as those presented in Table 7.1 and 7.2. We would expect similar efficiency gains of ML assisted detection and collection versus humans only, with the corresponding reduction in detection accuracy, or an improvement if a better lightweight model becomes available in the future.

In addition to comparing RipScout with non-expert human participants, we also evaluated our system and collected feedback from an experienced lifeguard. Lifeguards are much more experienced in detecting rip currents than our other participants. The detection performance of lifeguards averaged 34.2 seconds per rip, on par with that of RipScout. In terms of accuracy, RipScout was also able to detect all the rips that the lifeguard expert found. Although we only have one expert validation point at this time, the accuracy result was reassuring that RipScout performed satisfactorily.

7.6.3 Accuracy in the Field Tests

The accuracy metric presented in Tables 7.1 and 7.2 ignores false positives (rips that are not present but flagged as present) and false negatives (rips that are present but not detected). We looked for false positives and false negatives on the videos from the

field trip with the help of a rip current expert. In our analysis, we found that human participants had a 31% false positive rate, whereas RipScout has a 17% false positive rate. On the other hand, there were no general instances of false negatives for both human participants and RipScout. For the human participants, we believe the nature of the task (i.e., asking them to look for rip currents) led them to be more cautious and erred on flagging something as a rip when it is not. Hence, false negative rate was negligible. For RipScout, indeed there are frames where a rip was present but was not detected. However, at our sampling rate of 30fps, the same rip would always be detected in other frames, but necessarily all the frames. Hence, we marked the rip as being detected, leading to no false negative detection of the rip.

For the purpose of using RipScout for automated data collection, we considered the 17% false positive rate acceptable since we want to gather as much data as possible during each drone flight. The collected data can later be cleaned up using human experts (e.g. relabel the 17% false positive detections) or with a higher accuracy and more compute intensive two-stage ML model.

Moreover, we can further reduce the false positive and false negative rate by employing common ML model optimization techniques such as data augmentation, increasing the amount and quality of training data, and tuning the model's parameters. These techniques are widely used in the field of ML to improve a deep learning models' accuracy continuously [93].

7.7 Conclusion

In summary, RipScout is a system for rip current data collection, which integrates ML rip current detection into the flight control process. The ML system is carefully designed to work well under limited computational constraints, and we show that rip current detection accuracy of EfficientDet D2 is almost 5% better than its closest rival. Actual field tests show that this system is effective, allowing data to be collected almost four times faster than without RipScout, alleviating the need for domain experts to be present with the drone operator. Furthermore, with further validation, RipScout can be used to improve beach safety by providing rapid and wide coverage of beach monitoring for rips, helping to reduce fatigue of lifeguarding operations, especially in communities where they are understaffed.

RipScout highlights the locations of rips with bounding boxes. This not only helps users identify rip currents quickly, but it also serves as an effective tool for learning. We observed that after using the system, users improved their rip current detection skills even without the assistance of the ML model. This suggests that the system can serve as a training tool for beachgoers to learn how to spot rip currents and improve their overall beach safety awareness.

During our field tests, we observed that RipScout can readily detect rip currents in bright daylight where it is difficult for the human eyes to spot rip currents due to insufficient brightness and the small display of mobile devices. This helps make data collection less error prone. However, in poor lighting conditions including dense fog

or extreme glare in the video stream from the drone, the ML detection model will have difficulty finding rips. We plan to improve RipScout by augmenting our training data with additional data obtained in such conditions, as well as using generative AI to add these effects on our existing training data. Likewise, any ML detection model can only recognize rips that they were trained on. We plan to collect training data of rip currents with other types of visual signatures to further enhance RipScout.

While our focus has been on rip currents, we believe the proposed system architecture generalizes to other applications which require drone search over large areas to locate specific conditions and then collect image-based data. Examples might include biodiversity monitoring or search and rescue operations.

To encourage the use of this system for rip current data collection and other applications, the software code and all specifications are available as an open-source project at https://sites.google.com/ucsc.edu/ripscout/codes. To encourage research on rip currents using visual data, our datasets, including the new dataset of 2555 frames of sediment rip currents, are also available in the repository https://sites.google.com/ucsc.edu/ripscout/dataset.

Chapter 8

Automated Data Collection from Network Cameras using ML

8.1 Introduction

Webcams are prevalent in today's world, commonly used to monitor traffic, prevent crime, and observe public activities. Businesses like Surfline have capitalized on the extensive deployment of webcams, building services around real-time video feeds for surfers and beach-goers. Scientists can also harness the power of webcams for various research purposes. In coastal science, numerous coastal webcams are available that can be used to monitor changes in the shoreline, detect beach hazards, and study coastal erosion. This integration of webcams into scientific research ties directly into the broader context of data collection in my work. By utilizing existing webcam

infrastructure, valuable scientific data can be gathered efficiently and continuously. This approach demonstrates how readily available technologies can be repurposed to collect significant scientific data, much like how citizen scientists contribute through their observations and recordings. It highlights the potential for random coastal cams to provide useful data for scientific research, thus expanding the toolkit available for data collection and analysis in coastal science.

As introduced in the previous chapters, by leveraging network cameras installed along coastlines, it is possible to continuously monitor beach conditions and automatically identify rip currents in real time, thereby enhancing the efficiency and accuracy of rip current detection. This approach not only facilitates the collection of valuable data for further research and analysis but can also extend to other applications such as shoreline monitoring using segmentation methods and beach crowd level monitoring through people detection.

This work focuses on the development of a real-time rip current detection system using network cameras. The system integrates advanced computer vision techniques and ML models to analyze live video feeds and identify rip currents as they form. Additionally, the automated data collection capability of the system allows for the continuous accumulation of visual and environmental data, which can be used to improve the detection algorithms and enhance our understanding of rip current dynamics.

By implementing this real-time detection system, we aim to develop a robust and

Figure 8.1: The camera is deployed at the Walton Lighthouse near Seabright State Beach and Twin Lakes State Beach in Santa Cruz, CA, USA.

scalable solution that can be deployed across various coastal regions, significantly contributing to public safety and beach management efforts. This chapter details the technical specifications of an instance of such network camera installation and ML application deployment.

8.2 System Design and Implementation

The selected location for deploying the network camera is the top of the Walton Lighthouse in Santa Cruz, CA, USA. This location was chosen due to its elevated vantage point, which provides a clear view of large areas on two different beaches, Seabright State Beach and Twin Lakes State Beach (Figure 8.1), facilitating effective rip current monitoring, detection, and data collection. However, there are challenges associated with deploying the camera at this site, including the lack of power and

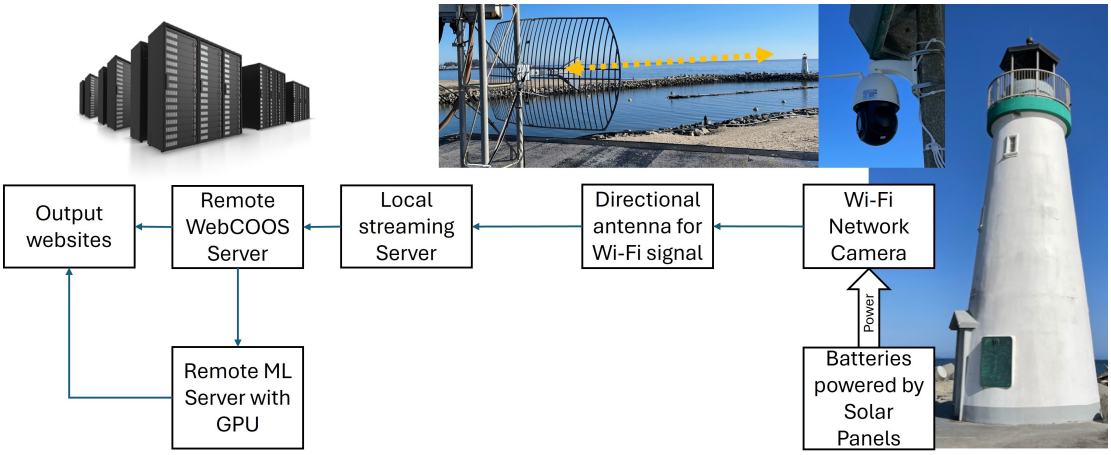


Figure 8.2: System Design and Implementation of the camera at Walton Lighthouse.

internet connectivity. The camera is part of WebCOOS, or the Webcam Coastal Observation System, a project implemented by SECOORA (Southeast Coastal Ocean Observing Regional Association) and funded by NOAA, aimed at developing a network of low-cost webcams for coastal observation to provide valuable data for scientific analysis, public safety, and resource management. The project focuses on standardizing data processing methodologies to make webcam data actionable for stakeholders and plans to expand these methodologies nationwide, enhancing the overall capability for coastal monitoring and management in the US.

The design of the system encompasses both hardware and software components (Figure 8.2). For the implementation of this system, we made various design choices based on the location and setup of the deployment, budget, logistics, and availability of equipment. In this section, we discuss the design choices specific to this instance, providing knowledge that can be useful for reproducibility and any future deployments

in different locations. A list of materials used in the system is provided in Table 8.1. The key components of the system are explained below.

Table 8.1: Materials and Specifications for the System.

Materials	**Specifications**
Generic Outdoor Camera	PTZ (Pan, Tilt, and optical Zoom) features Waterproof WiFi connectivity RTSP network protocol support
Solar panels	2 X 100W Monocrystalline Solar Panels MPPT (Maximum Power Point Tracker) controller with Bluetooth connectivity
Battery pack	LiFePO4 Battery 12V 100AH Lithium Battery Built-in 100A BMS (Battery Management System) Waterproof battery box
Parabolic Antenna Kit	Support for 2.4GHz WiFi signal 20-23 dBi of gain
WiFi router	WiFi router with port for external antenna connectivity
Mini PC for streaming	Dimensions: 4.9 x 4.4 x 1.6 inches CPU: AMD Ryzen 5 2.10 GHz Main memory: 16 GB
Cables and connectors	Various power cables and connectors as needed Various network cables and connectors as needed

Camera: A camera equipped with pan, tilt, and zoom (PTZ) capabilities collects input data as a video stream of sufficient quality for ML processing. Since the camera is deployed outdoors near the ocean, the selected model is designed to withstand adverse weather conditions such as rain and storms. The camera features both wired (RJ45 interface) and WiFi wireless network connectivity. WiFi connectivity was selected due to the lack of wired network connectivity in the lighthouse. In locations with wired internet, a wired network connection can be used without changing any other component of the overall system. The camera has a 5-megapixel image sensor and 30x optical zoom capabilities. The PTZ capability enables coverage of a large area for rip current data collection. To cover a large area, the camera is programmed to alternate between two viewpoints, covering Seabright State Beach and Twin Lakes State Beach for a predefined period. As an alternative to the WiFi wireless network camera, a wireless 4G/5G cellular PTZ camera can be used. Another more recent alternative is using a satellite internet service connected to the camera.

Power: As there is no electric power source available in the lighthouse, we installed a power system utilizing two 100-watt solar panels and a 100Ah battery pack. The choice of two 100-watt solar panels is based on the power consumption of the camera and the assumption that the system needs to operate continuously, even during periods of low sunlight. The camera typically draws about 20 watts during regular operation and up to 35 watts during physical panning or optical zooming. With an average daily runtime of 12 hours, the system requires approximately 300 watt-hours per day. The

100Ah battery provides 1,200 watt-hours of stored energy, which, coupled with the solar panels, ensures that the system can continue to operate for up to three days without significant sunlight. To save power, the camera is programmed to operate only during the daytime and turn off at night. A programmable WiFi switch is used for that purpose. For implementing a similar system at sites with readily available power, this component is not necessary.

Long-range Directional Antenna: As there is no wired internet connectivity in the lighthouse and no preexisting WiFi network coverage, a long-range directional parabolic antenna is used to provide a WiFi signal to the network camera, facilitating the required bandwidth for the real-time transmission of the video stream. The antenna is installed on a rooftop building in the most feasible location with wired network connectivity on the shore, approximately 1,000 feet away from the lighthouse. This component is not necessary for implementing a similar system at sites with readily available power.

Local Streaming Server: A low-powered mini PC serves as a local streaming server near the location. This server transmits the live video stream to the WebCOOS server. It is installed in the same building as the long-range directional antenna and connected to the same network. The server configuration includes an AMD Ryzen 5 2.10 GHz CPU and 16 GB of main memory. As this server is a low-powered, GPU-less, small form factor device, it is dedicated solely to streaming. No edge computing or ML processing

is performed here due to limited computational resources.

ML Processing Server: A remote computer with a GPU, capable of processing the video stream and running the ML algorithms in real time, receives the videos from the streaming server. One or more large ML models run on this server. The server specifications include an Intel Core i9 3.2 GHz CPU, 32 GB of main memory, and a Nvidia GeForce RTX 3080 GPU with 10 GB of memory, capable of processing each frame of the video stream with a large ML model in approximately 18 ms. This server performs tasks of automated data collection, data labeling, and reducing label noise.

ML models: Currently, two advanced object detection ML models—YOLOv8x [119] and RT-DETR [169]—are deployed within the system. These models are implemented using Python scripts and have been optimized for real-time performance. They are trained to detect both bathymetry rip currents and transient rip currents. The initial training dataset was compiled using the RipScout system described in the previous chapter and subsequently augmented with data collected directly via the network camera.

Website: A public-facing website has been developed to display the processed video feed, enhanced with visualizations derived from the ML outputs (Figure 8.3). These include bounding boxes that identify detected rip currents and segmentation masks that delineate the shoreline, offering an intuitive interface for stakeholders to monitor and analyze coastal conditions.

Figure 8.3: Automated rip current data collection and shoreline segmentation from wireless network camera.

8.3 Future Improvements

Future improvements could involve deploying low-powered, single-board computers with integrated GPUs, such as the NVIDIA Jetson Nano, at the camera site to process data locally using edge computing. This advancement would eliminate the need for a remote GPU server and enable on-site ML processing.

To enhance the overall quality of data collection and processing, future improvements should focus on methods that ensure the scientific value of the data. This underscores the importance of robust data processing techniques and methodological rigor in scientific data collection.

Chapter 9

Conclusion

Even though data is an essential component in scientific research and various applications, efficiently collecting high-quality data is challenging. In my dissertation, I explore strategies to improve the efficiency of the data collection process and enhance the quality of the gathered data from a novel perspective with a focus on visual data. This work is driven by my hypothesis that integrating citizen science with mobile technology and ML contributes to a significant advancement in the data collection process. Toward this goal, I have designed and implemented a few innovative platforms and tools, such as SmartCS, RipFinder, and RipScout, which empower the general public (including students) to participate in scientific data collection and analysis actively. I developed these tools to address the challenges of data collection efficiency and quality highlighted in the two key research questions (Chapter 1). By guiding and supporting the non-experts through the combined power of ML and

citizen science in the data collection process, the contribution of this work significantly enhances the overall data collection processes. Furthermore, the tools I created work towards maintaining data quality at the collection stage rather than relying on post-processing for corrections and filtering. This approach improves the effectiveness of data-dependent scientific investigations. It contributes to a better understanding and appreciation of science among the general public, advancing the state of the art in modern data collection processes.

9.1 Summary of Contributions

SmartCS Platform: The SmartCS platform is a user-friendly tool that enables the creation of ML-powered computer vision mobile apps for citizen science applications without requiring programming knowledge. By providing pre-built features and templates within a single framework, SmartCS facilitates rapid prototyping and deployment of mobile apps with ML guidance that can operate without internet connectivity. I validated the platform's effectiveness through a few user studies that demonstrated its usability and the quality of data collected by non-experts. This work highlights the potential of SmartCS to revolutionize citizen science by making advanced smartphone apps with ML capabilities accessible to a broader audience, enabling them to contribute with high-quality data.

Engaging High School Students in Research: Integrating ML within citizen science tools has proven effective for engaging high school students in meaningful research activities, as demonstrated in Chapter 5. Using a codeless platform like SmartCS, students can develop ML-powered mobile applications that contribute to real-world research projects. Chapter 5 investigates the educational benefits of this approach through the four research questions. The apps developed by the students illustrate my approach's positive and transformative impact on the students' learning experiences and their interest in STEM careers. As high school students were selected as a representative group of the general public, many lessons learned from this investigation also apply to them.

RipFinder - Real Time Rip Current Detection: The development of RipFinder, a smartphone app for real-time rip current detection, is an excellent example of using an ML-powered app for public safety issues. The main idea behind the app is to use multiple client-side and server-side ML models to detect the rip current with higher accuracy and collect data, even in remote locations without internet connectivity. The app's capability to detect rip currents in real time demonstrates the application of ML-powered mobile apps for impactful use cases such as public safety while contributing valuable data for scientific research.

RipScout - Real Time Data Collection using UAS: I present the design and implementation of RipScout, a real-time rip current detection and automated data

collection system using drones or UAS. The details of system architecture, the selection of ML models, and the various challenges faced during the deployment of this system are discussed based on some field test results and benchmarks. The RipScout system shows the potential to integrate ML with drones in practical, real-world applications such as environmental monitoring, providing valuable insights and data previously challenging to obtain. It also showcases the usefulness of ML guidance with UAS for collecting specialized field data by non-experts.

Automated Data Collection Using Network Cameras: I discuss implementing and deploying a system for real-time rip current detection and automated data collection using network cameras deployed in a remote location. The details of the design choices and technical aspects of installing and maintaining such a system, including the advancements achieved through the integration of ML, challenges posed by the remote environment, and the need for reliable data transmission and processing, are examined in Chapter 8. The goal is to investigate the methods and techniques that can be used to successfully deploy a system for applications similar to a rip current detection and automated data collection system from an engineering perspective.

9.2 Future Work

Based on the lesson learned from this work, several future directions and next steps can be taken, including expanding the scope and addressing the identified limitations

presented in the previous chapters.

Even though the SmartCS platform works well for the codeless creation of smartphone apps with ML, the features and customization options of the platform can be improved to make it better and allow the development of a broader range of citizen science applications. One such potential expansion of features can be enabling the support for integrating ML models for tasks such as semantic and instance segmentation and the continuous inclusion of newer ML models. Furthermore, providing additional data visualization and analysis tools will enable users to create sophisticated and effective apps.

The work investigating the impact of involving high school students in research through ML-power citizen science tools can progress further to learn valuable insights about the efficacy of this learning approach. Understanding how these experiences influence students' academic and career paths can help develop, improve, and expand such programs.

The combination of modern citizen science, ML, and mobile technology presents the opportunity to improve the data collection process to advance scientific research, enhance public engagement, and provide meaningful learning experiences. While the works based on this concept presented in this dissertation primarily focus on the rip current detection application, the same methodologies and systems can be translated and broadly applied to other areas such as environmental monitoring, biodiversity assessment, urban planning, autonomous vehicle research, and medical applications.

For instance, environmental monitoring can benefit from deploying citizen science applications similar to CoastSnap or SandSnap, allowing for large-scale data collection on coastal changes, erosion, pollution levels, etc. Biodiversity assessment can leverage applications like iNaturalist to document and monitor species distribution and abundance, contributing valuable data to research and conservation efforts. ML on mobile devices can be deployed for urban planning, understanding traffic patterns, and improving transportation systems. Autonomous vehicle research relies heavily on data to train ML models to work in various driving conditions, which can be enhanced through community-sourced data. Medical applications can employ these technologies for real-time health monitoring and diagnostics.

Based on the foundation of my work, we can look forward to a future where data collection is more effortless for the general public, enabling them to contribute to various research projects, thereby making scientific advancement a collaborative and inclusive endeavor accessible to all.

Appendix A

List of Related Publications

The following is a list of publications, either published or under peer review, contributed by the author and directly or indirectly related to this dissertation.

Publications directly related to this dissertation

The following publications are directly related to this dissertation. The text of this dissertation includes reprints of some of the materials listed here.

1. **Khan, Fahim Hasan**, Emily Lovell, Akila de Silva, Gregory Dusek, James Davis, and Alex Pang. "WIP: Citizen Science Tools with Machine Learning as a Pathway to Engage High School Students in Research." In 2024 IEEE Frontiers in Education Conference (FIE), pp. 1-5. IEEE, 2024.

2. **Khan, Fahim Hasan**, Akila de Silva, Gregory Dusek, James Davis, and

Alex Pang. “SmartCS: Enabling the Creation of Machine Learning–Powered Computer Vision Mobile Apps for Citizen Science Applications without Coding.” Citizen Science: Theory and Practice 9, no. 1 (2024).

3. **Khan, Fahim Hasan**, Akila de Silva, Gregory Dusek, James Davis, and Alex Pang. “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, pp. 89-94. 2021.

4. Jain, Nihar, and **Fahim Hasan Khan**. “Blood Cell Detection Using Deep Learning on Mobile Platforms.” In 2023 International Conference on Computational Science and Computational Intelligence (CSCI), pp. 1289-1293. IEEE, 2023.

5. Yeh, Chelsea, and **Fahim Hasan Khan**. “Citizen Science Mobile Apps with Machine Learning for Recyclable Objects.” In 2022 International Conference on Computational Science and Computational Intelligence (CSCI), pp. 1539-1542. IEEE, 2022.

6. **Khan, Fahim Hasan**, Akila de Silva, Ashleigh Palinkas, Gregory Dusek, James Davis, and Alex Pang. “RipFinder: Real time Rip Current Detection on Mobile Devices.” (Manuscript under review).

7. **Khan, Fahim Hasan**, Akila de Silva, Ashleigh Palinkas, Gregory Dusek, James Davis, and Alex Pang. “RipScout: Realtime ML-Assisted Rip Current Detection

and Automated Data Collection using UAVs." (Manuscript under review).

Publications indirectly related to this dissertation

Below is a list of publications that are not directly related to this dissertation; however, the knowledge gained from them was relevant and supportive of the research conducted.

1. de Silva, Akila, Mona Zhao, Donald Stewart, **Fahim Hasan Khan**, Gregory Dusek, James Davis, and Alex Pang. "RipViz: Finding Rip Currents by Learning Pathline Behavior." IEEE Transactions on Visualization and Computer Graphics (2023).

2. Luo, Jiahao, **Fahim Hasan Khan**, Issei Mori, Akila de Silva, Eric Sandoval Ruezga, Minghao Liu, Alex Pang, and James Davis. "How much does input data type impact final face model accuracy?." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 18985-18994. 2022.

3. Luo, Jiahao, **Fahim Khan**, Issei Mori, Akila de Silva, Eric Ruezga, and James Davis. "Face Models: How Good Does My Data Need To Be?." In 2021 IEEE International Conference on Image Processing (ICIP), pp. 3188-3192. IEEE, 2021.

4. **Khan, Fahim Hasan**, Akila de Silva, Jayanth Yetukuri, and Narges Norouzi. "Sequential Image Synthesis for Human Activity Video Generation." In Image Analysis and Recognition: 16th International Conference, ICIAR 2019, Waterloo, ON, Canada, August 27–29, 2019, Proceedings, Part II 16, pp. 129-133. Springer International Publishing, 2019.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Ayat Abourashed, Laura Doornekamp, Santi Escartin, Constantianus JM Koenraadt, Maarten Schrama, Marlies Wagener, Frederic Bartumeus, and Eric CM van Gorp. The potential role of school citizen science programs in infectious disease surveillance: A critical review. *International Journal of Environmental Research and Public Health*, 18(13):7019, 2021.

[3] Cemal Aker and Sinan Kalkan. Using deep networks for drone detection. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2017.

[4] SM Al-Salem, Paola Lettieri, and Jan Baeyens. Recycling and recovery routes of plastic solid waste (psw): A review. *Waste management*, 29(10):2625–2643, 2009.

[5] Nawal Soliman ALKolifi ALEnezi. A method of skin disease detection using image processing and machine learning. *Procedia Computer Science*, 163:85–92, 2019.

[6] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252, USA, 2019. PMLR.

[7] Viswanatha Reddy Allugunti. A machine learning model for skin disease classification using convolution neural network. *International Journal of Computing, Programming and Database Management*, 3(1):141–147, 2022.

[8] Oscar Alsing. Mobile object detection using tensorflow lite and transfer learning. Master's thesis, KTH, School of Electrical Engineering and Computer Science (EECS), 2018.

[9] EC Amazon. Amazon web services. *Available in: http://aws. amazon. com/es/ec2/(November 2012)*, page 39, 2015.

[10] Apple Inc. Apple iPhone 12 Pro - technical specifications.

[11] appsgeyser.com. Appsgeyser: Free app maker | create an app without code. https://appsgeyser.com/, 2022. (Accessed on 09/15/2022).

[12] JL Araújo, C Morais, and JC Paiva. Student participation in a coastal water quality citizen science project and its contribution to the conceptual and procedural learning of chemistry. *Chemistry Education Research and Practice*, 23(1):100–112, 2022.

[13] José Luís Araújo, Carla Morais, and João Carlos Paiva. Students' attitudes towards science: The contribution of a citizen science project for monitoring coastal water quality and (micro) plastics. *Journal of Baltic Science Education*, 20(6):881–893, 2021.

[14] Maria Aristeidou and Christothea Herodotou. Online citizen science: A systematic review of effects on learning and scientific literacy. *Citizen Science: Theory and Practice*, 5(1):1–12, 2020.

[15] William G Axinn and Lisa D Pearce. *Mixed method data collection strategies*. Cambridge University Press, 2006.

[16] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M Paixao, Filipe Mutz, et al. Self-driving cars: A survey. *Expert Systems with Applications*, 165:113816, 2021.

[17] Florence T Balagtas-Fernandez and Heinrich Hussmann. Model-driven development of mobile applications. In *2008 23rd IEEE/ACM International Conference on Automated Software Engineering*, pages 509–512, USA, 2008. IEEE.

[18] Samuel T Barber. 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):85–111, 2018.

[19] Mesay Belete Bejiga, Abdallah Zeggada, and Farid Melgani. Convolutional neural networks for near real-time object detection from UAV imagery in avalanche search and rescue operations. In *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 693–696. IEEE, 2016.

[20] Ekaba Bisong and Ekaba Bisong. Google colaboratory. *Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners*, pages 59–64, 2019.

[21] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

[22] Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of computational science*, 2(1):1–8, 2011.

[23] Rick Bonney, Tina B Phillips, Heidi L Ballard, and Jody W Enck. Can citizen science enhance public understanding of science? *Public understanding of science*, 25(1):2–16, 2016.

[24] Alex Bowyer, Chris Lintott, Greg Hines, Campbell Allen, and Ed Paget. Panoptes, a project building tool for citizen science. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing (HCOMP'15). AAAI, San Diego, CA, USA*, pages 1–2, 2015.

[25] Danah M Boyd and Nicole B Ellison. Social network sites: Definition, history, and scholarship. *Journal of computer-mediated Communication*, 13(1):210–230, 2007.

[26] Christian Brannstrom, Heather Lee Brown, Chris Houser, Sarah Trimble, and Anna Santos. "you can't see them from sitting here": Evaluating beach user understanding of a rip current warning sign. *Applied Geography*, 56:61–70, 2015.

[27] Robin F Brown, Bryan E Wright, Matthew J Tennis, and Steven Jeffries. California sea lion (zalophus californianus) monitoring in the lower columbia river, 1997–2018. *Northwestern Naturalist*, 101(2):92–103, 2020.

[28] Widodo Budiharto, Alexander AS Gunawan, Jarot S Suroso, Andry Chowanda, Aurello Patrik, and Gaudi Utama. Fast object detection for quadcopter drone

using deep learning. In *2018 3rd International Conference on Computer and Communication Systems (ICCCS)*, pages 192–195. IEEE, 2018.

[29] Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on psychological science*, 6(1):3–5, 2011.

[30] Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. Amazon's mechanical turk: A new source of inexpensive, yet high-quality data? *Perspectives on Psychological Science*, 6:3–5, 2016.

[31] buildfire.com. App builder | industry leading app maker for ios & android mobile apps. `https://buildfire.com/`, 2022. (Accessed on 09/15/2022).

[32] A Cole Burton, Eric Neilson, Dario Moreira, Andrew Ladle, Robin Steenweg, Jason T Fisher, Erin Bayne, and Stan Boutin. Wildlife camera trapping: a review and recommendations for linking surveys to ecological processes. *Journal of applied ecology*, 52(3):675–685, 2015.

[33] Yaru Cao, Zhijian He, Lujia Wang, Wenguan Wang, Yixuan Yuan, Dingwen Zhang, Jinglin Zhang, Pengfei Zhu, Luc Van Gool, Junwei Han, et al. VisDrone-DET2021: The vision meets drone object detection challenge results. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2847–2854, 2021.

[34] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[35] B Castelle, T Scott, RW Brander, and RJ McCarroll. Rip current types, circulation and hazard. *Earth-Science Reviews*, 163:1–21, 2016.

[36] B. Castelle, T. Scott, R.W. Brander, and R.J. McCarroll. Rip current types, circulation and hazard. *Earth-Science Reviews*, 163:1–21, 2016.

[37] Changrui Chen, Yu Zhang, Qingxuan Lv, Shuo Wei, Xiaorui Wang, Xin Sun, and Junyu Dong. RRNet: A hybrid detector for object detection in drone-captured images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 0–0, Oct 2019.

[38] Hsinchun Chen, Roger HL Chiang, and Veda C Storey. Business intelligence and analytics: From big data to big impact. *MIS quarterly*, pages 1165–1188, 2012.

[39] Jonathan H Chen and Steven M Asch. Machine learning and prediction in medicine—beyond the peak of inflated expectations. *The New England journal of medicine*, 376(26):2507, 2017.

[40] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional

nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[41] Min Chen, Shiwen Mao, and Yunhao Liu. Big data: A survey. *Mobile networks and applications*, 19:171–209, 2014.

[42] Tiffany Yu-Han Chen, Lenin Ravindranath, Shuo Deng, Paramvir Bahl, and Hari Balakrishnan. Glimpse: Continuous, real-time object recognition on mobile devices. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, SenSys '15, page 155–168, New York, NY, USA, 2015. Association for Computing Machinery.

[43] B Louise Chilvers and Stefan Meyer. Conservation needs for the endangered new zealand sea lion, phocarctos hookeri. *Aquatic Conservation: Marine and Freshwater Ecosystems*, 27(4):846–855, 2017.

[44] Yu-Chen Chiu, Chi-Yi Tsai, Mind-Da Ruan, Guan-Yu Shen, and Tsu-Tian Lee. Mobilenet-SSDv2: An improved object detection model for embedded systems. In *2020 International Conference on System Science and Engineering (ICSSE)*, pages 1–5, 2020.

[45] Floriana Ciaglia, Francesco Saverio Zuppichini, Paul Guerrie, Mark McQuade, and Jacob Solawetz. Roboflow 100: A rich, multi-domain object detection benchmark. *arXiv preprint arXiv:2211.13523*, 2022.

[46] citizenscience.org. Platforms for hosting participatory science projects - citizen science association. https://citizenscience.org/platforms-for-hosting-participatory-science-projects/, 2023. (Accessed on 10/17/2023).

[47] citsci.org. Citizen science association: Building the field of citizen science. https://citsci.org/, 2022. Accessed: 2022-10-16.

[48] Robert N. Colwell. *Manual of Remote Sensing*. American Society of Photogrammetry, 1983.

[49] Lewis M. Cowardin, Virginia Carter, Francis C. Golet, and Edward T. LaRoe. Classification of wetlands and deepwater habitats of the united states. Technical report, U.S. Fish and Wildlife Service, 1979.

[50] cpu monkey.com. Apple A14 bionic - benchmark, test und technical specifications.

[51] John W Creswell and J David Creswell. *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications, 2017.

[52] cybertracker.org. Cybertracker: The most efficient way of field data collection. https://cybertracker.org/, 2022. Accessed: 2022-10-16.

[53] Akila de Silva, Issei Mori, Gregory Dusek, James Davis, and Alex Pang.

Automated rip current detection with region based convolutional neural networks. *Coastal Engineering*, 166:103859, 2021.

[54] Akila de Silva, Mona Zhao, Donald Stewart, Fahim Hasan, Gregory Dusek, James Davis, and Alex Pang. Ripviz: Finding rip currents by learning pathline behavior. *IEEE Transactions on Visualization and Computer Graphics*, 2023.

[55] Akila de Silva, Mona Zhao, Donald Stewart, Fahim Hasan, Gregory Dusek, James Davis, and Alex Pang. RipViz: Finding rip currents by learning pathline behavior. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–13, 2023.

[56] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[57] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[58] Jianing Deng, Zhiguo Shi, and Cheng Zhuo. Energy-efficient real-time UAV object detection on embedded platforms. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(10):3123–3127, 2020.

[59] Jane Disney, Duncan Bailey, Anna Farrell, and Ashley Taylor. Next generation citizen science using anecdata. org. *Maine Policy Review*, 26(2):70–79, 2017.

[60] Jane Disney, Duncan Bailey, Anna Farrell, Ashley Taylor, and Bridie McGreavy. Anecdata. org: An online citizen science platform for building climate resilient communities. In *OCEANS 2018 MTS/IEEE Charleston*, pages 1–4, USA, 2018. IEEE.

[61] DJI. Phantom 4 pro v2.0 - specifications - dji. `https://www.dji.com/phantom-4-pro-v2/specs`, 2023. (Accessed on 09/14/2023).

[62] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

[63] Dawei Du, Pengfei Zhu, Longyin Wen, Xiao Bian, Haibin Lin, Qinghua Hu, Tao Peng, Jiayu Zheng, Xinyao Wang, Yue Zhang, et al. VisDrone-DET2019: The vision meets drone object detection in image challenge results. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[64] Andrei Dumitriu, Florin Tatui, Florin Miron, Radu Tudor Ionescu, and Radu Timofte. Rip current segmentation: A novel benchmark and YOLOv8 baseline results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1261–1271, June 2023.

[65] G Dusek and H Seim. A Probabilistic Rip Current Forecast Model. *Journal of Coastal Research*, 29(4):909 – 925, 2013.

[66] Andres Echeverria, Idoia Ariz, Judit Moreno, Javier Peralta, and Esther M Gonzalez. Learning plant biodiversity in nature: The use of the citizen–science platform inaturalist as a collaborative tool in secondary education. *Sustainability*, 13(2):735, 2021.

[67] Melissa V Eitzel, Jessica L Cappadonna, Chris Santos-Lang, Ruth Ellen Duerr, Arika Virapongse, Sarah Elizabeth West, Christopher Kyba, Anne Bowser, Caren Beth Cooper, Andrea Sforzi, et al. Citizen science terminology matters: Exploring key terms. *Citizen Science: Theory and Practice*, 2(1):1–20, 2017.

[68] Sarah Elwood, Michael F Goodchild, and Daniel Z Sui. Researching volunteered geographic information: Spatial data, geographic research, and new social practice. *Annals of the association of American geographers*, 102(3):571–590, 2012.

[69] epicollect.net. Epicollect5: Free and easy-to-use mobile data collection. `https://five.epicollect.net/`, 2023. Accessed: 2022-10-16.

[70] Martin J Eppler. *Managing information quality: Increasing the value of information in knowledge-intensive products and processes*. Springer Science & Business Media, 2006.

[71] Andre Esteva, Alexandre Robicquet, Bharath Ramsundar, Volodymyr Kuleshov, Mark DePristo, Katherine Chou, Claire Cui, Greg Corrado, Sebastian Thrun, and Jeff Dean. A guide to deep learning in healthcare. *Nature medicine*, 25(1):24–29, 2019.

[72] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[73] FAA. FAA unmanned aircraft systems (UAS), Mar 2022.

[74] Tom P Fairchild, Mike S Fowler, Sabine Pahl, and John N Griffin. Multiple dimensions of biodiversity drive human interest in tide pool communities. *Scientific reports*, 8(1):1–11, 2018.

[75] Heng Fan, Dawei Du, Longyin Wen, Pengfei Zhu, Qinghua Hu, Haibin Ling, Mubarak Shah, Junwen Pan, Arne Schumann, Bin Dong, et al. VisDrone-MOT2020: The vision meets drone multiple object tracking challenge results. In *European Conference on Computer Vision*, pages 713–727. Springer, 2020.

[76] Fastlane-community. Fastlane: The easiest way to automate beta deployments and releases for your ios and android apps, 2023. Accessed: insert date.

[77] fathomnet.org. Fathomnet. `https://fathomnet.org/fathomnet`, 2022. (Accessed on 09/15/2022).

[78] Xin Feng, Youni Jiang, Xuejiao Yang, Ming Du, and Xin Li. Computer vision algorithms and hardware implementations: A survey. *Integration*, 69:309–320, 2019.

[79] Craig W Fisher and Bruce R Kingma. Criticality of data quality as exemplified in two disasters. *Information & Management*, 39(2):109–116, 2001.

[80] Luciano Floridi and Massimo Chiriatti. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30:681–694, 2020.

[81] Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869, 2013.

[82] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[83] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37, 2014.

[84] Amir Gandomi and Murtaza Haider. Beyond the hype: Big data concepts, methods, and analytics. *International journal of information management*, 35(2):137–144, 2015.

[85] Andrew Garbett, Rob Comber, Edward Jenkins, and Patrick Olivier. App

movement: A platform for community commissioning of mobile applications. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 26–37, 2016.

[86] Victor A. Gensini and Walker S. Ashley. An examination of rip current fatalities in the united states. *Natural Hazards*, 54(1):159–175, Jul 2010.

[87] Mohammadali Gharaat, Mohammadreza Sharbaf, Bahman Zamani, and Abdelwahab Hamou-Lhadj. Alba: a model-driven framework for the automatic generation of android location-based apps. *Automated Software Engineering*, 28(1):1–45, 2021.

[88] Fabio Giglietto, Luca Rossi, and Davide Bennato. The open laboratory: Limits and possibilities of using facebook, twitter, and youtube as a research data source. *Journal of technology in human services*, 30(3-4):145–159, 2012.

[89] Tilmann Gneiting and Matthias Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1(1):125–151, 2014.

[90] Hervé Goëau, Pierre Bonnet, Alexis Joly, Vera Bakić, Julien Barbe, Itheri Yahiaoui, Souheil Selmi, Jennifer Carré, Daniel Barthélémy, Nozha Boujemaa, et al. Pl@ ntnet mobile app. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 423–424, 2013.

[91] Gregory R. Goldsmith. The field guide, rebooted. *Science*, 349(6248):594–594, 2015.

[92] Michael F Goodchild. Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69:211–221, 2007.

[93] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[94] Google. Flatbuffers: Flatbuffers. `https://flatbuffers.dev/`. (Accessed on 07/08/2023).

[95] Google. *Firebase Authentication Documentation*. Google, 2023.

[96] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[97] Mordechai Muki Haklay, Daniel Dörler, Florian Heigl, Marina Manzoni, Susanne Hecker, Katrin Vohland, et al. What is citizen science? the challenges of definition. *The science of citizen science*, 13, 2021.

[98] Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE intelligent systems*, 24(2):8–12, 2009.

[99] Courtney H Hann, Lei Lani Stelle, Andrew Szabo, and Leigh G Torres. Obstacles and opportunities of using a mobile app for marine mammal research. *ISPRS International Journal of Geo-Information*, 7(5):169, 2018.

[100] Mitchell Harley, Michael Kinsela, Elena Sánchez Sánchez-García, and Kilian Vos. Coastsnap: Crowd-sourced shoreline change mapping using smartphones. In *AGU Fall Meeting Abstracts*, volume 2018, pages EP52D–26, USA, 2018. SAO/NASA Astrophysics Data System.

[101] Jane K Hart and Kirk Martinez. Environmental sensor networks: A revolution in the earth system science? *Earth-Science Reviews*, 78(3-4):177–191, 2006.

[102] Christian Heipke. Crowdsourcing geospatial data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(6):550–557, 2010.

[103] Tim Highfield and Tama Leaver. Instagrammatics and digital methods: Studying visual social media, from selfies and gifs to memes and emoji. *Communication research and practice*, 2(1):47–62, 2016.

[104] Martin Hilbert and Priscila López. The world's technological capacity to store, communicate, and compute information. *science*, 332(6025):60–65, 2011.

[105] Suzanne E Hiller and Anastasia Kitsantas. The effect of a horseshoe crab citizen science program on middle school student science performance and stem career motivation. *School Science and Mathematics*, 114(6):302–311, 2014.

[106] Colleen Hitchcock, Heather Vance-Chalcraft, and Maria Aristeidou. Citizen science in higher education. *Citizen science: Theory and practice*, 6(1), 2021.

[107] Leif Howard, Charles B van Rees, Zoe Dahlquist, Gordon Luikart, and Brian K

Hand. A review of invasive species reporting apps for citizen science and opportunities for innovation. *NeoBiota*, 71:165–188, 2022.

[108] Yuheng Hu, Lydia Manikonda, and Subbarao Kambhampati. What we instagram: A first analysis of instagram photo content and user types. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pages 595–598, 2014.

[109] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 0–0, July 2017.

[110] Peter Hubwieser, Michal Armoni, and Michail N Giannakos. How to implement rigorous computer science education in k-12 schools? some answers and many questions. *ACM Transactions on Computing Education (TOCE)*, 15(2):1–12, 2015.

[111] Philipp Hummer and Christine Niedermeyer. Don't walk alone: Synergy effects for citizen science created through adaptive platform design in spotteron. In *Austrian Citizen Science Conference 2018*, page 66, 2018.

[112] Ngu Phuc Huy and Do Vanthanh. Evaluation of mobile app paradigms.

In *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia*, pages 25–30, USA, 2012. ACM.

[113] Ayaz Hyder and Andrew A May. Translational data analytics in exposure science and environmental health: A citizen science approach with high school students. *Environmental Health*, 19:1–12, 2020.

[114] images.cv. Images.cv: Your machine learning and data science community. https://images.cv/, 2023. Accessed: 2022-10-16.

[115] ispotnature.org. ispot nature: Your place to share nature. https://www.ispotnature.org/, 2022. Accessed: 2022-10-16.

[116] Nihar Jain and Fahim Hasan Khan. Blood cell detection using deep learning on mobile platforms. In *2023 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1289–1293. IEEE, 2023.

[117] Joel Janai, Fatma Güney, Aseem Behl, Andreas Geiger, et al. Computer vision for autonomous vehicles: Problems, datasets and state of the art. *Foundations and Trends® in Computer Graphics and Vision*, 12(1–3):1–308, 2020.

[118] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Yolo by ultralytics. ORCID: https://orcid.org/0000-0001-5950-6979, https://orcid.org/0000-0002-7603-6750, https://orcid.org/0000-0003-3783-7069.

[119] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLOv8, 2023.

[120] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, Yonghye Kwon, Kalen Michael, Jiacong Fang, Zeng Yifu, Colin Wong, Diego Montes, et al. ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation. *Zenodo*, 2022.

[121] Burke Johnson and F Turner. Data collection strategies. *Handbook of mixed methods in social and behavioural research. Thousand Oaks: Sage*, pages 297–315, 2003.

[122] Mona Erfani Joorabchi, Ali Mesbah, and Philippe Kruchten. Real challenges in mobile app development. In *2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, pages 15–24, USA, 2013. IEEE.

[123] Michael I Jordan and Tom M Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[124] V. Roshan Joseph. Optimal ratio for data splitting. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 15(4):531–538, 2022.

[125] kaggle.com. Kaggle: Your machine learning and data science community. `https://www.kaggle.com/`, 2022. Accessed: 2022-10-16.

[126] Yue Kang, Hang Yin, and Christian Berger. Test your self-driving algorithm: An

overview of publicly available driving datasets and virtual testing environments. *IEEE Transactions on Intelligent Vehicles*, 4(2):171–185, 2019.

[127] Magnus Høholt Kaspersen, Karl-Emil Kjær Bilstrup, Maarten Van Mechelen, Arthur Hjort, Niels Olof Bouvin, and Marianne Graves Petersen. High school students exploring machine learning and its societal implications: Opportunities and challenges. *International Journal of Child-Computer Interaction*, 34:100539, 2022.

[128] Avita Katal, Mohammad Wazid, and Rayan H Goudar. Big data: issues, challenges, tools and good practices. In *2013 Sixth international conference on contemporary computing (IC3)*, pages 404–409. IEEE, 2013.

[129] Kakani Katija, Eric Orenstein, Brian Schlining, Lonny Lundsten, Kevin Barnard, Giovanna Sainz, Oceane Boulais, Megan Cromwell, Erin Butler, Benjamin Woodward, et al. Fathomnet: A global image database for enabling artificial intelligence in the ocean. *Scientific reports*, 12(1):15914, 2022.

[130] Shinpei Kato, Eijiro Takeuchi, Yoshio Ishiguro, Yoshiki Ninomiya, Kazuya Takeda, and Tsuyoshi Hamada. An open approach to autonomous vehicles. *IEEE Micro*, 35(6):60–68, 2015.

[131] Gloria Ashiya Katuka, Yvonika Auguste, Yukyeong Song, Xiaoyi Tian, Amit Kumar, Mehmet Celepkolu, Kristy Elizabeth Boyer, Joanne Barrett, Maya Israel, and Tom McKlin. A summer camp experience to engage middle school learners

in ai through conversational app development. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, pages 813–819, 2023.

[132] Doron Katz. *Continuous delivery for mobile with fastlane: automating mobile application development and deployment for iOS and Android*. Packt Publishing Ltd, 2018.

[133] Gurmeher Kaur, Kris Jordan, and Jasleen Kaur. Using foundational cs1 curricula for middle school & early high school computer programming education. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, pages 827–833, 2023.

[134] Julia Kelemen-Finan, Martin Scheuch, and Silvia Winter. Contributions from citizen science to science education: an examination of a biodiversity citizen science project with schools in central europe. *International Journal of Science Education*, 40(17):2078–2098, 2018.

[135] Ruth Kermish-Allen, Karen Peterman, and Christine Bevc. The utility of citizen science projects in k-5 schools: measures of community engagement and student impacts. *Cultural Studies of Science Education*, 14(3):627–641, 2019.

[136] Fahim Hasan Khan, Akila de Silva, Gregory Dusek, James Davis, and Alex Pang. 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*, pages 89–94, 2021.

[137] Fahim Hasan Khan, Akila de Silva, Gregory Dusek, James Davis, and Alex Pang. 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*, CSCW '21, page 89–94, New York, NY, USA, 2021. Association for Computing Machinery.

[138] Fahim Hasan Khan, Akila de Silva, Gregory Dusek, James Davis, and Alex Pang. 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), 2024.

[139] Fahim Hasan Khan, Emily Lovell, Akila de Silva, Gregory Dusek, James Davis, and Alex Pang. Wip: Citizen science tools with machine learning as a pathway to engage high school students in research. In *2024 IEEE Frontiers in Education Conference (FIE)*, pages 1–5. IEEE, 2024.

[140] Tae Kyun Kim. T test as a parametric statistic. *Korean journal of anesthesiology*, 68(6):540–546, 2015.

[141] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen

Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.

[142] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[143] Rob Kitchin. Big data, new epistemologies and paradigm shifts. *Big data & society*, 1(1):2053951714528481, 2014.

[144] Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. The future of crowd work. In *Proceedings of the 2013 conference on Computer supported cooperative work*, pages 1301–1318, 2013.

[145] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

[146] Sotiris B Kotsiantis, Ioannis Zaharakis, P Pintelas, et al. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1):3–24, 2007.

[147] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[148] Christopher Kullenberg and Dick Kasperowski. What is citizen science?–a scientometric meta-analysis. *PloS one*, 11(1):e0147152, 2016.

[149] Neeraj Kumar, Peter N Belhumeur, Arijit Biswas, David W Jacobs, W John Kress, Ida C Lopez, and João VB Soares. Leafsnap: A computer vision system for automatic plant species identification. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part II 12*, pages 502–516. Springer, 2012.

[150] Doug Laney et al. 3d data management: Controlling data volume, velocity and variety. *META group research note*, 6(70):1, 2001.

[151] A Lapresta-Fernández and LF Capitán-Vallvey. Environmental monitoring using a conventional photographic digital camera for multianalyte disposable optical sensors. *Analytica chimica acta*, 706(2):328–337, 2011.

[152] Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser. *Research methods in human-computer interaction*. Morgan Kaufmann, 2017.

[153] Stephen B Leatherman. Rip current measurements at three south florida beaches. *Journal of Coastal Research*, 33(5):1228–1234, 2017.

[154] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[155] Jangwon Lee, Jingya Wang, David Crandall, Selma Šabanović, and Geoffrey Fox. Real-time, cloud-based object detection for unmanned aerial vehicles. In *2017 First IEEE International Conference on Robotic Computing (IRC)*, pages 36–43, USA, 2017. IEEE.

[156] Jangwon Lee, Jingya Wang, David Crandall, Selma Šabanović, and Geoffrey Fox. Real-time, cloud-based object detection for unmanned aerial vehicles. In *2017 First IEEE International Conference on Robotic Computing (IRC)*, pages 36–43, 2017.

[157] Kyung Mog Lee. Design of a smart phone application controlling agricultural watering system with a drone. In *Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering*, 2018.

[158] Rob Lemmens, Vyron Antoniou, Philipp Hummer, and Chryssy Potsiou. Citizen science in the digital world of apps. *The Science of Citizen Science*, 461, 2021.

[159] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *2011 IEEE intelligent vehicles symposium (IV)*, pages 163–168. IEEE, 2011.

[160] Yiqun Li, Aiyuan Guo, and Ching Ling Chin. A platform for mobile augmented reality app creation without programming. In *SIGGRAPH Asia 2015 Mobile Graphics and Interactive Applications*, pages 1–1. ACM, USA, 2015.

[161] Yiting Li, Qingsong Fan, Haisong Huang, Zhenggong Han, and Qiang Gu. A modified yolov8 detection network for uav aerial image recognition. *Drones*, 7(5):304, 2023.

[162] Thomas M. Lillesand and Ralph W. Kiefer. *Remote Sensing and Image Interpretation*. John Wiley & Sons, 1979.

[163] Tjen-Sien Lim, Wei-Yin Loh, and Yu-Shan Shih. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine learning*, 40(3):203–228, 2000.

[164] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[165] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

[166] Hai-Ying Liu, Daniel Dörler, Florian Heigl, and Sonja Grossberndt. Citizen science platforms. In *The Science of Citizen Science*, pages 439–459. Springer, Cham, Switzerland, 2021.

[167] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[168] lobe.ai. Machine learning made easy, 2022.

[169] Wenyu Lv, Shangliang Xu, Yian Zhao, Guanzhong Wang, Jinman Wei, Cheng Cui, Yuning Du, Qingqing Dang, and Yi Liu. DETRs beat YOLOs on real-time object detection, 2023.

[170] Jamie MacMahan, Ad Reniers, Jenna Brown, Rob Brander, Ed Thornton, Tim Stanton, Jeff Brown, and Wendy Carey. An introduction to rip currents based on field observations. *Journal of Coastal Research*, 27(4):iii–vi, 2011.

[171] Tia C Madkins, Jakita O Thomas, Jessica Solyom, Joanna Goode, Frieda McAlear, and S Grover. Learner-centered and culturally relevant pedagogy. *Computer science in K-12: An A-to-Z handbook on teaching programming*, pages 125–129, 2020.

[172] Stuart Madnick and Hongwei Zhu. Improving data quality through effective use of data semantics. *Data & Knowledge Engineering*, 59(2):460–475, 2006.

[173] Gayathri Manikutty, Sreejith Sasidharan, and Bhavani Rao. Driving innovation through project based learning: A pre-university steam for social good initiative. In *2022 IEEE Frontiers in Education Conference (FIE)*, pages 1–8. IEEE, 2022.

[174] Corey Maryan, Md Tamjidul Hoque, Christopher Michael, Elias Ioup, and Mahdi Abdelguerfi. Machine learning applications in detecting rip channels from images. *Applied Soft Computing*, 78:84–93, 2019.

[175] Mohamed Lamine Mekhalfi, Carlo Nicolò, Yakoub Bazi, Mohamad Mahmoud Al Rahhal, Norah A. Alsharif, and Eslam Al Maghayreh. Contrasting YOLOv5, Transformer, and EfficientDet detectors for crop circle detection in desert. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2022.

[176] Rico Merkert and James Bushell. Managing the drone revolution: A systematic literature review into the current use of airborne drones and future strategic directions for their effective control. *Journal of Air Transport Management*, 89:101929, 2020.

[177] Issei Mori, Akila de Silva, Gregory Dusek, James Davis, and Alex Pang. Flow-based rip current detection and visualization. *IEEE Access*, 10:6483–6495, 2022.

[178] Laurence Moroney and Laurence Moroney. Using authentication in firebase. *The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform*, pages 25–50, 2017.

[179] Luigi Mucerino, Luca Carpi, Chiara F Schiaffino, Enzo Pranzini, Eleonora Sessa, and Marco Ferrari. Rip current hazard assessment on a sandy beach in liguria, nw mediterranean. *Natural hazards*, 105:137–156, 2021.

[180] Greg Newman, Andrea Wiggins, Alycia Crall, Eric Graham, Sarah Newman, and Kevin Crowston. The future of citizen science: emerging technologies and shifting paradigms. *Frontiers in Ecology and the Environment*, 10(6):298–304, 2012.

[181] Francesco Nex and Fabio Remondino. Uav for 3d mapping applications: a review. *Applied geomatics*, 6:1–15, 2014.

[182] Paraskevi Nousi, Ioannis Mademlis, Iason Karakostas, Anastasios Tefas, and Ioannis Pitas. Embedded UAV real-time visual object detection and tracking. In *2019 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 708–713. IEEE, 2019.

[183] Jill Nugent. inaturalist: citizen science for 21st-century naturalists. *Science Scope*, 41(7):12–15, 2018.

[184] Sten Odenwald. Smartphone sensors for citizen science applications: Radioactivity and magnetism. *Citizen Science: Theory and Practice*, 4(1), 2019.

[185] Boris Otto. Quality and value of the data resource in large enterprises. *Information Systems Management*, 32(3):234–251, 2015.

[186] Rafael Padilla, Sergio L Netto, and Eduardo AB Da Silva. A survey on performance metrics for object-detection algorithms. In *2020 international conference on systems, signals and image processing (IWSSIP)*, pages 237–242. IEEE, 2020.

[187] Rafael Padilla, Wesley L. Passos, Thadeu L. B. Dias, Sergio L. Netto, and Eduardo A. B. da Silva. A comparative analysis of object detection metrics with a companion open-source toolkit. *Electronics*, 10(3), 2021.

[188] Kathryn Paige, Robert Hattam, and Christopher B Daniels. Two models for implementing citizen science projects in middle school. *The Journal of Educational Enquiry*, 14(2), 2015.

[189] Gabriele Paolacci, Jesse Chandler, and Panagiotis G Ipeirotis. Running experiments on amazon mechanical turk. *Judgment and Decision making*, 5(5):411–419, 2010.

[190] Elise Paradis, Bridget O'Brien, Laura Nimmon, Glen Bandiera, and Maria Athina Martimianakis. Design: Selection of data collection methods. *Journal of graduate medical education*, 8(2):263–264, 2016.

[191] Shyamal Patel, Hyung Park, Paolo Bonato, Leighton Chan, and Mary Rodgers. A review of wearable sensors and systems with application in rehabilitation. *Journal of neuroengineering and rehabilitation*, 9:1–17, 2012.

[192] Michael Quinn Patton. *Qualitative research & evaluation methods: Integrating theory and practice*. Sage publications, 2014.

[193] Shweta Philip and Alex Pang. Detecting and visualizing rip current using optical flow. In *EuroVis (Short Papers)*, pages 19–23, 2016.

[194] Patricia Pulliam Phillips and Cathy A Stawarski. *Data collection: Planning for and collecting all types of data*. John Wiley & Sons, 2008.

[195] Leo L Pipino, Yang W Lee, and Richard Y Wang. Data quality assessment. *Communications of the ACM*, 45(4):211–218, 2002.

[196] plantnet.org. Plantnet: The plant identification app. `https://plantnet.org/`, 2022. Accessed: 2022-10-16.

[197] Zheng Qin, Zeming Li, Zhaoning Zhang, Yiping Bao, Gang Yu, Yuxing Peng, and Jian Sun. Thundernet: Towards real-time generic object detection on mobile devices. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[198] Hana Ra, Benjamin L. Richards, Audrey Rollo, Dianna Miller-Greene, and Jeremy Taylor. Keeping track of hawaii's bottomfish populations with the help of citizen scientists. *Fisheries*, n/a(n/a), 2022.

[199] Neelesh Rampal, Tom Shand, Adam Wooler, and Christo Rautenbach.

Interpretable deep learning applied to rip current detection and localization. *Remote Sensing*, 14(23), 2022.

[200] Ashraf Haroon Rashid, Imran Razzak, Muhammad Tanveer, and Antonio Robles-Kelly. RipDet: A fast and lightweight deep neural network for rip currents detection. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE, 2021.

[201] Jainesh Rathod, Vishal Waghmode, Aniruddh Sodha, and Praseniit Bhavathankar. Diagnosis of skin diseases using convolutional neural networks. In *2018 second international conference on electronics, communication and aerospace technology (ICECA)*, pages 1048–1051, USA, 2018. IEEE.

[202] Thomas C Redman. The impact of poor data quality on the typical enterprise. *Communications of the ACM*, 41(2):79–82, 1998.

[203] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[204] Zuzana Reitermanova et al. Data splitting. In *WDS*, volume 10, pages 31–36, Prague, Czechia, 2010. Matfyzpress Prague.

[205] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[206] Arry Retnowati, Muh Aris Marfai, and JT Sri Sumantyo. Rip currents signatures zone detection on alos palsar image at parangtritis beach, indonesia. *Indonesian Journal of Geography*, 43(2):12–27, 2012.

[207] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019.

[208] roboflow.com. Roboflow: Give your software the power to see objects in images and video. `https://roboflow.com/`, 2023. Accessed: 2022-10-16.

[209] Joseph Roche, Laura Bell, Cecília Galvão, Yaela N Golumbic, Laure Kloetzer, Nieke Knoben, Mari Laakso, Julia Lorke, Greg Mannion, Luciano Massetti, et al. Citizen science, education, and learning: Challenges and opportunities. *Frontiers in Sociology*, 5:613814, 2020.

[210] Nicole M Rodriguez, Alisa Arce, Alice Kawaguchi, Jenna Hua, Bonnie Broderick, Sandra J Winter, and Abby C King. Enhancing safe routes to school programs through community-engaged citizen science: two pilot investigations in lower density areas of santa clara county, california, usa. *BMC public health*, 19:1–11, 2019.

[211] Yuji Roh, Geon Heo, and Steven Euijong Whang. A survey on data collection

for machine learning: a big data-ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1328–1347, 2019.

[212] Ali Rohan, Mohammed Rabah, and Sung-Ho Kim. Convolutional neural network-based real-time object detection and tracking for parrot AR drone 2. *IEEE Access*, 7:69575–69584, 2019.

[213] Holly Rosser and Andrea Wiggins. Tutorial designs and task types in zooniverse. In *Companion of the 2018 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '18, page 177–180, New York, NY, USA, 2018. Association for Computing Machinery.

[214] Algeir P Sampaio, Paulo CMA Farias, and Roberto A Bittencourt. A case study of using machine learning in k-12 education. In *2023 IEEE Frontiers in Education Conference (FIE)*, pages 1–8. IEEE, 2023.

[215] SA Sanchez, HJ Romero, and AD Morales. A review: Comparison of performance metrics of pretrained models for object detection using the tensorflow framework. In *IOP Conference Series: Materials Science and Engineering*, volume 844, page 012024, Bristol, United Kingdom, 2020. IOP Publishing.

[216] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In

2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4510–4520, Los Alamitos, CA, USA, 2018. IEEE Computer Society.

[217] Monica Scannapieco, Antonino Virgillito, Carlo Marchetti, Massimo Mecella, and Roberto Baldoni. The daquincis architecture: a platform for exchanging and improving data quality in cooperative information systems. *Information systems*, 29(7):551–582, 2004.

[218] Jennifer Schneiderhan-Opel and Franz X Bogner. How fascination for biology is associated with students' learning in a biodiversity citizen science project. *Studies in Educational Evaluation*, 66:100892, 2020.

[219] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning*, pages 4528–4537, USA, 2018. PMLR.

[220] David Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison. Hidden technical debt in machine learning systems. *Advances in neural information processing systems*, 28, 2015.

[221] SECOORA. Webcams for coastal observations and operational support.

[222] Harsh R Shah and Luis R Martinez. Current approaches in implementing

citizen science in the classroom. *Journal of microbiology & biology education*, 17(1):17–22, 2016.

[223] Saleh Shahinfar, Paul Meek, and Greg Falzon. "how many images do i need?" understanding how sample size per class affects deep learning model performance metrics for balanced designs in autonomous wildlife monitoring. *Ecological Informatics*, 57:101085, 2020.

[224] T Shanthi, RS Sabeenian, and R Anand. Automatic diagnosis of skin diseases using convolution neural network. *Microprocessors and Microsystems*, 76:103074, 2020.

[225] Viktor B Shapovalov, Yevhenii B Shapovalov, Zhanna I Bilyk, Anna P Megalinska, and Ivan O Muzyka. The google lens analyzing quality: an analysis of the possibility to use in the educational process. *environment*, 2:3, 2019.

[226] Aliaksandra Shutsko. User-generated short video content in social media. a case study of tiktok. In *Social Computing and Social Media. Participation, User Experience, Consumer Experience, and Applications of Social Computing: 12th International Conference, SCSM 2020, Held as Part of the 22nd HCI International Conference, HCII 2020, Copenhagen, Denmark, July 19–24, 2020, Proceedings, Part II 22*, pages 108–125. Springer, 2020.

[227] Nahian Siddique, Sidike Paheding, Colin P Elkin, and Vijay Devabhaktuni. U-

net and its variants for medical image segmentation: A review of theory and applications. *Ieee Access*, 9:82031–82057, 2021.

[228] Robert Simpson, Kevin R Page, and David De Roure. Zooniverse: observing the world's largest citizen science platform. In *Proceedings of the 23rd international conference on world wide web*, pages 1049–1054, USA, 2014. ACM.

[229] Kylie Soanes, Kate Cranney, Marie C Dade, Amy M Edwards, Ravindra Palavalli-Nettimi, and Tim S Doherty. How to work with children and animals: A guide for school-based citizen science in wildlife research. *Austral Ecology*, 45(1):3–14, 2020.

[230] Caterina Solé, Digna Couso, and María Isabel Hernández. Citizen science in schools: A systematic literature review. *International Journal of Science Education, Part B*, pages 1–17, 2023.

[231] Parvathaneni Naga Srinivasu, Jalluri Gnana SivaSai, Muhammad Fazal Ijaz, Akash Kumar Bhoi, Wonjoon Kim, and James Jin Kang. Classification of skin disease using deep learning neural networks with mobilenet v2 and lstm. *Sensors*, 21(8):2852, 2021.

[232] Lars St, Svante Wold, et al. Analysis of variance (anova). *Chemometrics and intelligent laboratory systems*, 6(4):259–272, 1989.

[233] Zachary R Steelman, Bryan I Hammer, and Moez Limayem. Data collection in the digital age. *MIS quarterly*, 38(2):355–378, 2014.

[234] Diane M Strong, Yang W Lee, and Richard Y Wang. Data quality in context. *Communications of the ACM*, 40(5):103–110, 1997.

[235] Chenfan Sun, Wei Zhan, Jinhiu She, and Yangyang Zhang. Object detection from the video taken by drone via convolutional neural networks. *Mathematical Problems in Engineering*, 2020, 2020.

[236] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022.

[237] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114, USA, 2019. PMLR.

[238] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019.

[239] Mingxing Tan, Ruoming Pang, and Quoc V Le. EfficientDet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020.

[240] Suriyon Tansuriyavong, Hideto Koja, Motoki Kyan, and Takashi Anezaki. The development of wildlife tracking system using mobile phone communication

network and drone. In *2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, volume 3, pages 351–354. IEEE, 2018.

[241] Omer Tene and Jules Polonetsky. Big data for all: Privacy and user control in the age of analytics. *Nw. J. Tech. & Intell. Prop.*, 11:239, 2012.

[242] Tensorflow. Tensorflow 2 detection model zoo, May 2021.

[243] Juan Terven and Diana Cordova-Esparza. A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond. *arXiv preprint arXiv:2304.00501*, 2023.

[244] Nils Tijtgat, Wiebe Van Ranst, Toon Goedeme, Bruno Volckaert, and Filip De Turck. Embedded real-time object detection for a UAV warning system. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 2110–2118, 2017.

[245] Roger F. Tomlinson. Current and potential uses of geographical information systems: The north american experience. *International Journal of Geographical Information Systems*, 1(3):203–218, 1987.

[246] Yongxin Tong, Zimu Zhou, Yuxiang Zeng, Lei Chen, and Cyrus Shahabi. Spatial crowdsourcing: a survey. *The VLDB Journal*, 29:217–250, 2020.

[247] Eric J Topol. High-performance medicine: the convergence of human and artificial intelligence. *Nature medicine*, 25(1):44–56, 2019.

[248] Jennifer Tsan, David Weintrop, Donna Eatinger, and Diana Franklin. Learner ideas and interests expressed in open-ended projects in a middle school computer science curriculum. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, pages 820–826, 2023.

[249] Compton J. Tucker, John R. Townshend, and T. E. Goff. African land-cover classification using satellite data. *Science*, 227(4685):369–375, 1985.

[250] Andranik Tumasjan, Timm Sprenger, Philipp Sandner, and Isabell Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. In *Proceedings of the international AAAI conference on web and social media*, volume 4, pages 178–185, 2010.

[251] David P Turner, Warren B Cohen, Robert E Kennedy, Karin S Fassnacht, and John M Briggs. Relationships between leaf area index and landsat tm spectral vegetation indices across three temperate zone sites. *Remote sensing of environment*, 70(1):52–68, 1999.

[252] Tzutalin. Labelimg is a graphical image annotation tool and label object bounding boxes in images, 2015.

[253] Tzutalin. A curated list of awesome data labeling tools, 2019.

[254] Ultralytics. Yolov8: A new state-of-the-art computer vision model, 2023. Accessed: 2023-07-15.

[255] Eric Umuhoza and Marco Brambilla. Model driven development approaches for mobile applications: A survey. In *International Conference on Mobile Web and Information Systems*, pages 93–107, USA, 2016. Springer.

[256] Marek Vajgl, Petr Hurtik, and Petra Števuliáková. Drone real-time control based on pattern matching. In *2017 Joint 17th World Congress of International Fuzzy Systems Association and 9th International Conference on Soft Computing and Intelligent Systems (IFSA-SCIS)*, pages 1–6. IEEE, 2017.

[257] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018.

[258] Maarten Van Mechelen, Rachel Charlotte Smith, Marie-Monique Schaper, Mariana Tamashiro, Karl-Emil Bilstrup, Mille Lunding, Marianne Graves Petersen, and Ole Sejer Iversen. Emerging technologies in k–12 education: A future hci research agenda. *ACM Transactions on Computer-Human Interaction*, 30(3):1–40, 2023.

[259] Cor N Verdouw, J Wolfert, AJM Beulens, and Agathe Rialland. Virtualization of food supply chains with the internet of things. *Journal of Food Engineering*, 176:128–136, 2016.

[260] Bas Vergouw, Huub Nagel, Geert Bondt, and Bart Custers. Drone

technology: Types, payloads, applications, frequency spectrum issues and future developments. In *The future of drone use*, pages 21–45. Springer, 2016.

[261] Katrin Vohland, Anne Land-Zandstra, Luigi Ceccaroni, Rob Lemmens, Josep Perelló, Marisa Ponti, Roeland Samson, and Katherin Wagenknecht. *The science of citizen science*. Springer Nature, 2021.

[262] Justice T Walker, Amanda Barany, Alex Acquah, Sayed Mohsin Reza, Alan Barrera, Karen Del Rio Guzman, and Michael A Johnson. Coding like a data miner: A sandbox approach to computing-based data science for high school student learning. In *2023 IEEE Frontiers in Education Conference (FIE)*, pages 1–5. IEEE, 2023.

[263] Chen Wang, Aibek Musaev, Pezhman Sheinidashtegol, and Travis Atkison. Towards detection of abnormal vehicle behavior using traffic cameras. In *Big Data–BigData 2019: 8th International Congress, Held as Part of the Services Conference Federation, SCF 2019, San Diego, CA, USA, June 25–30, 2019, Proceedings 8*, pages 125–136. Springer, 2019.

[264] Qi Wang, Yue Ma, Kun Zhao, and Yingjie Tian. A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, 9(2):187–212, 2022.

[265] Richard Y Wang and Diane M Strong. Beyond accuracy: What data quality means to data consumers. *Journal of management information systems*, 12(4):5–33, 1996.

[266] Zirui Wang, Zihang Dai, Barnabas Poczos, and Jaime Carbonell. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[267] Susan C Weller and A Kimball Romney. *Systematic data collection*, volume 10. Sage publications, 1988.

[268] Brandon C Welsh and David P Farrington. Public area cctv and crime prevention: an updated systematic review and meta-analysis. *Justice quarterly*, 26(4):716–745, 2009.

[269] Steven Euijong Whang and Jae-Gil Lee. Data collection and quality challenges for deep learning. *Proceedings of the VLDB Endowment*, 13(12):3429–3432, 2020.

[270] Andrea Wiggins and Kevin Crowston. From conservation to crowdsourcing: A typology of citizen science. In *2011 44th Hawaii international conference on system sciences*, pages 1–10. IEEE, 2011.

[271] wildme.org. Home | wild me. `https://www.wildme.org`, 2022. (Accessed on 09/15/2022).

[272] Marco Willi, Ross T Pitman, Anabelle W Cardoso, Christina Locke, Alexandra Swanson, Amy Boyer, Marten Veldthuis, and Lucy Fortson. Identifying animal species in camera trap images using deep learning and citizen science. *Methods in Ecology and Evolution*, 10(1):80–91, 2019.

[273] Sarah A Wood, Patrick W Robinson, Daniel P Costa, and Roxanne S Beltran. Accuracy and precision of citizen scientist animal counts from drone imagery. *PloS one*, 16(2):e0244040, 2021.

[274] Xin Wu, Wei Li, Danfeng Hong, Ran Tao, and Qian Du. Deep learning for unmanned aerial vehicle-based object detection and tracking: A survey. *IEEE Geoscience and Remote Sensing Magazine*, 10(1):91–124, 2021.

[275] www.andromo.com. Andromo - mobile app builder for android and ios. `https://www.andromo.com/`, 2022. (Accessed on 09/15/2022).

[276] www.appypie.com. No code app development & workflow automation platform. `https://www.appypie.com/`, 2022. (Accessed on 09/15/2022).

[277] Xiaoling Xia, Cui Xu, and Bing Nan. Inception-v3 for flower classification. In *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*, pages 783–787, USA, 2017. IEEE.

[278] Keiichi Yasumoto, Hirozumi Yamaguchi, and Hiroshi Shigeno. Survey of real-time processing technologies of iot data streams. *Journal of Information Processing*, 24(2):195–202, 2016.

[279] Chelsea Yeh and Fahim Hasan Khan. Citizen science mobile apps with machine learning for recyclable objects. In *2022 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1539–1542. IEEE, 2022.

[280] Chelsea Yeh and Fahim Hasan Khan. Citizen science mobile apps with machine learning for recyclable objects. In *2022 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1539–1542, 2022.

[281] Maryam Yousef, Farkhund Iqbal, and Mohammed Hussain. Drone forensics: A detailed analysis of emerging DJI models. In *2020 11th International Conference on Information and Communication Systems (ICICS)*, pages 066–071. IEEE, 2020.

[282] Tal Z Zarsky. Incompatible: The gdpr in the age of big data. *Seton Hall L. Rev.*, 47:995, 2016.

[283] Yao Zhang, Wanru Huang, Xunan Liu, Chi Zhang, Guodong Xu, and Bin Wang. Rip current hazard at coastal recreational beaches in china. *Ocean & Coastal Management*, 210:105734, 2021.

[284] Yu Zhang, Yun Wang, Haidong Zhang, Bin Zhu, Siming Chen, and Dongmei Zhang. Onelabeler: A flexible system for building data labeling tools. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery.

[285] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection

with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[286] Daoheng Zhu, Rui Qi, Pengpeng Hu, Qianxin Su, Xue Qin, and Zhiqiang Li. YOLO-Rip: A modified lightweight network for rip currents detection. *Frontiers in Marine Science*, 9, 2022.

[287] Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Haibin Ling, Qinghua Hu, Qinqin Nie, Hao Cheng, Chenfeng Liu, Xiaoyu Liu, et al. VisDrone-DET2018: The vision meets drone object detection in image challenge results. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[288] Xingkui Zhu, Shuchang Lyu, Xu Wang, and Qi Zhao. TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2778–2788, 2021.