APPROVAL SHEET

Title of Dissertation:	HUMAN-MACHINE INTELLIGENCE: A DESIGN PARADIGM
Name of Candidate:	Mahbubur Rahman Doctor of Philosophy, 2020

Dissertation and Abstract Approved: N.Banerjee

Dr. Nilanjan Banerjee
Associate Professor
Department of Computer Science and
Electrical Engineering

Date Approved: 01/31/2020

ABSTRACT

Title of Document: HUMAN-MACHINE INTELLIGENCE: A DESIGN PARADIGM

Mahbubur Rahman, Doctor of Philosophy, 2020

Directed By: Dr. Nilanjan Banerjee, Associate Professor
Department of Computer Science & Electrical Engineering

In this age of artificial intelligence, we are witnessing the power of human-machine collaboration in transforming the way we live, work, and solve different problems. Humans and machines can complement each other in resolving intractable and sophisticated issues that are hard or impossible for computers alone. The collaboration achieved great results addressing the problems of digitizing books, detecting star clusters, and transcribing audio and video, etc. Researchers investigated these problems in isolation. There is no clear guideline about why and when human intelligence can be useful and, if so, what design pattern to follow. Integrating humans will add human knowledge, which can help to solve complex, open-ended, and uncertain problems. However, this will also bring the human limitations of less automation, less precision, and biased opinion. Analyzing the tread-off of integrating humans is necessary before designing a collaborative system.

In this dissertation, we have addressed the issues described above and propose a collaborative system design paradigm. Analyzing the general architecture of such a system, we found that human intelligence can help at three different functional positions - data preprocessing, feature extraction, and decision making. In all these functional areas, humans can help to improve the performance of a system. We also provide the conditions that will help a system to get rid of humans in the long run. We have developed four different

systems that represent all four conditions mentioned above and provide detailed guidelines. We provided detailed steps of integrating humans in decision making, feature extraction, and preprocessing through our weed identification system, resource localization, and group conversation analysis system, respectively. We also explained the conditions and steps to reduce human contribution in the long run through the object detection system.

In each system, we showed - a) why humans over computer intelligence are necessary? b) what are the steps to integrate human knowledge to overcome the difficulties? c) what are the trade-off on integrating humans instead of machines?

HUMAN-MACHINE INTELLIGENCE: A DESIGN PARADIGM

by

Mahbubur Rahman

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2020

Dedication

To my beloved parents, who envisioned my dreams and used every endeavor possible to create a smooth path to achieve those dreams.

ACKNOWLEDGMENTS

First and foremost, I would like to acknowledge my advisor and committee chair, Dr. Nilanjan Banerjee. He has indeed been a mentor to me. This dissertation would not have been possible without his continuous mentoring, guidance, understanding, and support.

I would also like to thank Dr. Ryan Robucci for his time and support. His guidance to pay attention to details will continue to benefit me in future endeavors.

I would like to express gratitude to my committee members Dr. Ting Zhu, Dr. Charles Nicholas, and Dr. Nirmalya Roy, for their time and support.

I am very grateful to my family, who provides continuous support and strength throughout the whole journey.

I am also thankful to everyone who has helped me throughout the journey.

This research is financially supported by National Science Foundation (NSF) grants Grants 1308723, 1305099, 1314024, 1314024, and by grants Grants from the Arkansas Corn and Grain Sorghum Board (ACGSB) and Arkansas Soybean Promotion Board (ASPB), respectively. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or the ACGSB and ASPB.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

We are living in an exciting time in the history of technology where artificial intelligence (AI) (Marr 1977; Russell & Norvig 2016) is taking over every aspect of our life. Human beings are the most intelligent species. They learn a lot of different things very quickly. A machine, on the other hand, is precise at work if it is designed correctly and trained to do a specific task. Thus, machine-human augmentation to replicate human intelligence and beyond, through computers, brought a breakthrough in technology and automation.

In recent years, machine learning (ML) is emerging as the most significant technique. ML is very efficient at optimizing if there is a clear goal set up. It has been instrumental in accumulating data from different sources, enriching it, and classifying at a scale and speed. It can also generate insight from the training dataset. However, it fails if there is no clear goal set, or the data is not significant enough to train a good ML model. The lack of data can be leveraged using human intelligence because humans are more efficient at making judgments even with a small set of data. For these reasons, human-machine intelligence augmentation to enable a better decision-making system is a sweet spot.

Integrating human intelligence brings a new level of intelligence to human-machine collaborative systems, as well as some flaws that humans possess. Humans have extraordinary cognitive intelligence, but their abilities vary in quality from person to person, and they often introduce errors. Compared to humans, machine intelligence is excellent at large-scale storage, high computation, and high precision. Researchers tried to optimize human-machine intelligent systems integrating humans as less as possible to avoid the side-effects of adding human intelligence. The evolutionary result comes through ML, which becomes the leading technology to solve problems in every aspect of human life. From natural language processing, video analysis to self-driving cars, ML is becoming one of the successful techniques. However, ML requires a significant amount of structured and labeled data that we get from humans. Moreover, addressing all problems with ML is impossible as not all domains have sufficient data to train an ML model. The partially trained model with limited data creates multiple competitive outcomes from a set of classes and could not disambiguate them precisely. ML also requires high storage and high processing power, which is not available in embedded, mobile, and sensor systems. However, humans have a unique quality of analyzing things with a limited number of instances, which can leverage the lack of data, big storage and high processing power.

Many researchers proposed human-machine augmented systems to solve problems in different domains. However, none offers a comprehensive analysis of human-machine collaboration, where and how human intelligence can be useful with limited intervention by humans and what is the trade-off of integrating humans. In this dissertation, we present a comprehensive analysis of human-machine collaborative systems and propose various techniques to answer these questions. It also finds out answer for- *a) When should we consider to integrate humans?, and b) Can we reduce human dependency eventually?*

1.1 Human-machine Integration

Humans and machines can play a complementary role in the human-machine intelligence system. However, it is very challenging to integrate both of their better skills. Humans are very good at making a decision even with limited information and in varying environmental conditions with ease, which is very difficult for a machine. Incorporating human intelligence also adds human weaknesses to the system as well. It increases system latency, decreases precision, and is prone to subjective decisions. The process of integrating human intelligence into machine algorithms is very challenging and increases the sophistication of any system.

The typical way of incorporating human intelligence is to find out what information the system requires from humans, how to get that information, how to make sure about its quality. The target is always to use as less information as possible from humans. The devices with high processing power and energy and with a significant amount of structured data ensure the minimal level of human intervention in human-machine collaborative systems. Data scarcity or less sophisticated devices require more help from human intelligence. Furthermore, if the number of data instances increases overtime of a domain, the dependency on humans may decrease in the future with data-driven design.

This research attempted to discover the scope and requirements of integrating human intelligence at each functional position of an application. We researched and developed a few human-machine intelligent systems through which we show *a) what information from humans will be enough for the specific function of an application? b) How can that information be obtained from humans? c) What would be the effect of these integrations?*

1.2 Trade-off of Integrating Humans

Human-machine augmented systems introduce attributes related to humans in addition to software and hardware attributes. Sophisticated designs of any fields embody trade-offs embedded by designers (Kazman *et al.* 1998). The software and hardware attributes, i.e., latency, accuracy, and throughput, is impacted by the design choice that the designer selects over a set of possible design components. When a designer adds human intelligence as part of the design component, new attributes related to humans, i.e., error, fatigue, high response time, and complexity, are also added to the architecture. The more a system accepts human input, the more the system performance and complexity align with human behavior. Every architect should consider the trade-off of selecting each of the components. Accepting more information from the human crowd over the machine will add amazing cognitive skills of humans but will also be affected by human flaws and limitations. This research provides an empirical analysis of the trade-off of integrating humans with machines in different domains.

1.3 Dissertation Organization

The rest of the research is organized as follows:

Chapter 2 on page 6 is derived from state of the art research articles to discuss some of the backgrounds and motivations behind this research. It explains where we can add humans contribution. It also provides different design architectures integrating humans at different functional levels. Chapter 3 on page 22 is derived from our work on the weed identification system where a hierarchical human crowd contributes to decision space and helps identify weeds in the crop fields with high accuracy even with a limited number of data. Chapter 4 on page 49 is derived from our work on speech dynamics analysis from a group conversation, where humans contribute at the data preprocessing stage. Chapter 5 on page 68 is derived from another of our works where human intelligence helps to extract features. The chapter also shows the case that the enrichment of data does not contribute to reducing human contribution in the long run. Chapter 6 on page 88 is derived from our work on obstacle detection on the sidewalk using low powered and less complicated devices. Chapter 7 on page 109 explains some empirical research findings and how they affect a system at different performance benchmarks, and finally, Chapter 8 on page 121 describes some of the limitations of this research and shows the future direction of this domain.

Chapter 2

HUMAN-MACHINE COLLABORATIVE SYSTEM

In the early 90s, human-machine collaboration became an emerging technology. Researchers at that time defined collaboration as a process in which two or more agents work together to achieve a shared goal (Terveen 1995). Human-computer collaboration is the process of collaboration involving at least one computational agents and one human (Horvitz 1999). In a collaborative system, one of the agents can initiate action, access information, and send responses. This type of system is often called a mixed-initiative system (Cook & Thomas 2005). It has been studied in vast areas, including AI problem-solving (Ferguson, Allen, & others 1998) and knowledge discovery (Valdés-Pérez 1999).

2.1 Importance of Human-machine Collaboration

The recent success of AI, especially ML, is drastically changing the relationship between humans and their surrounding environment (McCarthy & Hayes 1981; Holland & others 1992). The new brand of AI is resolving problems of high uncertainty, complexity, and vulnerability of every field of scientific research, engineering, and social activity (Eakin & Luers 2006; Ledford 2015; Martin 2012). The success in AI is so disruptive that it is even challenging humans with its creativity and effectiveness (Monostori & Prohaszka 1993; Stone *et al.* 2016). In recent years, the deep learning version of ML development increased rapidly with the increase of data collection, storage, and processing power (Demuth *et al.* 2014; Sun, Wang, & Tang 2014). The AI boom touched almost all the fields, especially wearable devices (Son *et al.* 2014), big data analysis (Youseff, Butrico, & Da Silva 2008), cloud computing (Iafrate 2018), and robotics (Lee 2013). The success may lead to think that every problem can be solved using data, storage, and processing power. However, the state of art AI still depends on human intelligence, and there are several reasons behind that.

First, the state of the art AI has several limitations compared to humans. AI can solve problems based on experience and knowledge. However, AI cannot be considered simply the combination of models and algorithms from AI methods. Some of the AI systems succeeded to challenge humans with the vast computing power in some specific fields like DeepBlue (Campbell, Hoane Jr, & Hsu 2002) in chess, AlphaGo in boardgame (Mnih *et al.* 2015) and Watson in question answering (Rachlin 2012). However, these are the results of high training and computing power, and it does not have creativity and own thought process. We cannot consider these as general AI as there is a clear gap of self-learning ability between AI and these systems (Simon 2019; Newell, Simon, & others 1972; Selfridge 1958). With the widespread success of AI, the intelligent machine has become an integral part of our day to day life and will be more prevalent in our future society. However, there is a slight difference between machine intelligence and human intelligence. Humans face many problems that are very sophisticated, nondeterministic, uncertain, and open-ended. The intelligence behind any intelligent system is developed based on human intelligence as the human is the ultimate judge of those attributes (Zheng *et al.* 2017).

Second, embedded, and sensor system does not have high processing power and storage. In the era of IoT, devices works in a group to support humans in carrying out their daily activities in natural way with ease using the information and intelligence that is hidden in the IoT network connected sensor and embedded systems (Madakam *et al.* 2015). More and more poeple are joining the network and by 2020 more than 7.6 billion people will be directly connected to get intelligence provided by these smaller devices (Ismail 2019). But, these smaller devices do not possess high storage and processing power.

Third, data is not the solution for everything. Ultimate AI success depends on the data. However, only data does not guarantee human intervention free intelligent system. There are a vast number of problems that are being solved by AI but still need human intelligence to verify. The very known problems are Alexa, Google voice (Hoy 2018), search engine, industrial risk control (Mantere *et al.* 2012; De Rocquigny, Devictor, & Tarantola 2008), medical diagnosis (Kononenko 2001; Szolovits, Patil, & Schwartz 1988), and many more. Nowadays, the Internet is so rich in information that we can get any data by searching and processing from different sources. There is a temptation that the availability of a vast amount of data will be able to solve every problem using machine learning. However, the collected data from the internet does not always possesses quality required for a specific domain. We present two case studies where we collected data from internet, preprocessed and trained state of the art machine learning model.

2.1.1 Case Studies 1: Medical Activity Detection from Video

Problem Statement: In a neonatal care center, children with health issues are admitted to the hospital for proper caring and treatment. These children require extra care during regular health checkups, feeding, diaper changing, give medication, change of oxygen, and to supply blood. The routine is strict as these children are in a vulnerable situation. Hence, it is crucial to monitor rigorously that the doctors and nurses are not missing any step. With so many children, keeping written logs is not enough. There is a chance of human error as many children are at the same place. An automated process of logging every activity performed for a child will be a great way to monitor the process. The monitoring will help to find the anomaly on the treatment process as well as will help to avoid those steps in the future.

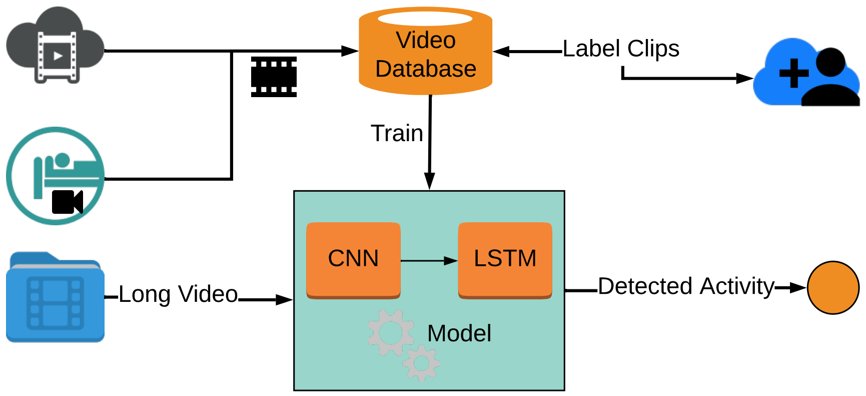


FIG. 2.1: System architecture of medical activity detection from videos. Video of different activities is collected from children hospital as well as from the cloud. A group of experts annotated actions in a video stream. A deep learning model is trained with a set of annotated videos. For a long recorded video, the model detects the temporal location and activity type from the video.

Data: Collecting data of neonatal is very hard as the parents are very concerned about the privacy of their children. In the national childcare center, we placed a camera in the child's bed frame focusing on the children and recorded high-quality videos. We recorded twelve children's treatment procedures with their parent's consent. We recorded twenty-four hours to seventy-two hours of video for each child. The recorded video contains all the activities performed by the nurses and doctors. We annotated all those videos and labeled segments for different actions present at a specific segment. We labeled five frequent activities from those videos, which are - diaper change, injection, feeding, blood test, and surgery. From all the recorded video, we created a database of activity that contains twenty-five instances of each activity. However, training deep neural network model with twenty-five instances of each class is not enough. Online could be a great source of information. There are lots of video instances of similar activity available on YouTube and different childcare sites. We searched for those videos to increase the dataset. Though there are lots of videos available online with related activities, the camera position, quality, and caring process are different. With our best effort, we were able to increase video instances from twenty-five to forty only of each class.

Training and Evaluating an ML Model: We designed a deep learning neural network-based activity detection model as shown in Figure 2.1. We adopted the deep neural network proposed by Nieto al et. (Montes, Salvador, & Giró i Nieto 2016) and trained the model with our dataset.

Results and Observation: The trained model is not sufficient to detect video activity from a long video. The quality of videos recorded in the hospital was at low resolutions, and a fixed point camera was hiding the actual activities. The internet instances have a better view but have different camera positions. Also, the number of instances that represent were limited, comparing the requirements of the state of the art activity detection model.

2.1.2 Case Studies 2: Automated Speech Recognition for Children

Problem Statement: Automated Speech Recognition (ASR) is a famous and influential AI technique that recognizes human speech and can exchange words interactively. Research on automated speech recognition gains much attention nowadays. Though the research on adult speech recognition is in advance phase, little research has been done on developing ASR for children. If we have an ASR for children, it will be easier to monitor the children's language development. It can also help to detect red flags of autism and to measure step by step language development.

Data: The essential part of any automated speech recognition is the collection of speech from different types of people to capture all possible environment, scope, and accent. The voices are segmented into small parts, transcribed, and consolidated to create a corpus. The corpus can be used to train a statistical or machine learning model to learn the phonemes. The trained model with decoder forms an ASR which can recognize human speech in real-time or offline. There are vast collections of speech data that Google, Amazon, or Facebook has gathered. Lots of corpora available online for a different language to develop ASR for that language. However, there is no corpus with significant speech collection of children. The major challenge of collecting children's voices is privacy and lack of commercial interests. Moreover, children are not active online. Children's voice is different from the adults as their vocal cord frequency is different from an adult. Most of the research done on children's speech uses adult corpus and transforms the adult frequency to children. However, the accuracy of those models falls far bellow than expected, since it also does not reflect children's accent, language model, and frequency.

Some language development researchers collected children's voices to monitor the development progress. They published those voices online. We collected a few of the corpora, merged with CSLU kid's corpus to create a significant corpus. However, the corpus does not reflect the necessary criteria for designing an ASR. If there are enough instances of the same sentences or words pronounced by many children, it will help to train a sound ASR system.

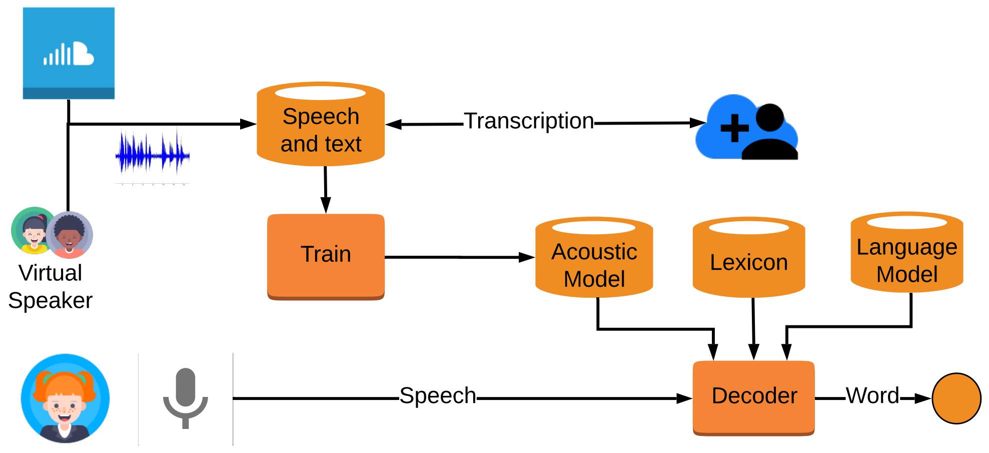


FIG. 2.2: Children voices are collected from different research labs. An extensive set of audio is also recorded using Acapella speech tools. Humans transcribed the audio. A GMM-based acoustic model was trained. An automated speech recognition(ASR) for kinds was built with an acoustic model, lexicon, and language model. The trained model accepts speech and converts it to the corresponding text.

Speech synthesis tool **Acapella** (Acapella Accessed 2019 08 23) uses real children's voices to generate synthesized speech for children who are not able to talk. Furthermore, many speakers participated in producing core phoneme. This feature makes it a good candidate for recording the speech of children and build a corpus. The difference of children corpus from adult one is that the language model and word collection are not the same for adults and children. To resolve the issue, we selected a list of children's books and used the tools to read those books using different child's voices. The child book collections ensure that the vocabulary covers the children's level of vocabulary. We built a corpus consolidating recorded audio and their transcript. The corpus constitutes of one hundred and twenty hours of audio from fifteen speakers, male and female, combined with different age groups (3-8).

Training and Evaluation: We used Kaldi ASR (Povey *et al.* 2011) tool to train a statistical GMM based acoustic model. The trained acoustic model was combined with a lexicon and language model (see the Figure 2.2) to build an ASR for kids. The ASR can take a segment of a child's voice and translate it into a corresponding text.

Results and Observation: We tested and validated our model using random, synthesized, and real audio. The word error rate (WER) for synthesized sound was 12%, but for real children voices, the WER was 81%. It is obvious from this model that even with a large amount of data, we cannot guarantee accuracy if it does not represent the exact scenario.

The data that we use in the case of video activity detection is not useful. Also, the data available on the internet does not represent the exact camera position that is required to detect activities. In the case of child ASR, we found that with unrelated data, the ML model failed to achieve acceptable accuracy. The accuracy of an ML model is not all about data. The reasons explained above brings the interest of using human-computer integration, resolving problems which computer cannot solve alone. Human intelligence can leverage some of the factors behind those reasons, which include lack of data, sophisticated computation, ever-changing dynamics of issues, and application acceptance criteria. Human intelligence can be the leverage of all of these limitations and vice versa.

2.2 Methods of Integrating Human-machine

The above discussion shows that human-machine integration can be a great technique to solve many complex problems. However, this introduces an extra level of design complexity. The foundation of a human-computer collaborative system is to take both of their better skills. The designers require to optimize the combination to extract both of their better, avoiding the adverse effects. Researchers have tried to find a systematic approach to integrating individual or group intelligence and machine intelligence to perform a specific task. The target of this approach is to be better leveraged by both contributors. However, the system should be more automated than manual. Fitts (Fitts 1951) attempted to investigate the characteristics of various tasks better performed by humans than machines and vice versa. This list was regarded for a long time as the guideline of a collaborative system. However, the advancement of technology brought both of these actors more closer than before. Humans and machines have their distinct skills; at the same time, they can help each other. Jordan (Jordan 1963) later articulated that human and machine can play a complementary role, rather than antithetical.

Researchers proposed several techniques to solve different problems integrating humans and machines. The functional allocation between human and machine is envisioned as an iterative process rather than a definitive listing (Price 1985). A specific human-machine collaborative system can be optimized by considering many parameters. The iterative feedback from the computer can even improve the performance of human skills. Self-assessment is better than random response, but with external feedback, the individual yields better performance (Kulkarni, Can, & Hartmann 2011; Dow & Klemmer 2011; Douglas & Kirkpatrick 1999). Relevant feedback in image processing has already proved to be successful (Zhou & Huang 2003). Researchers also tried to apply human intelligence at different stages of a collaborative system. In a sensor-based system, the intelligence can

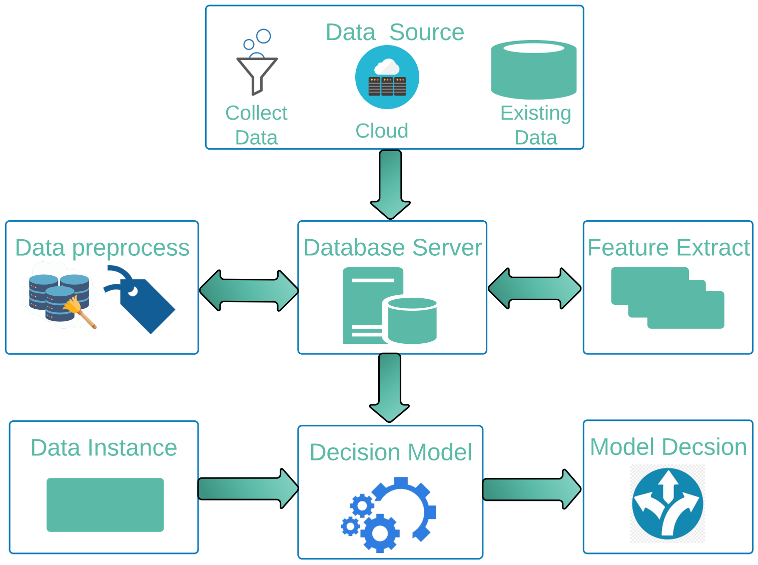


FIG. 2.3: In a data-driven system, humans can contribute at different functional positions-a) data preprocessing, b) feature extraction and c) decision making

be integrated into feature space as well as at algorithmic or functional level (Guo *et al.* 2015). In the classification system, human intelligence can be integrated into the data collection and labeling stage. Some applications may require more granular level intelligence, such as for feature extraction, selection, and decision making.

We analyzed a standard collaborative system (see Figure 2.3). A generic collaborative system architecture consists of data collection, preprocessing, and labeling. The labeled data is analyzed to engineer data features that are related to the problem of interest and are extracted by computer or humans. A set of features related to the actual problem are selected for training a machine model. Once trained, the model serves to provide the actual response to a query. There are three distinct functional phases where humans can contribute-preprocessing phase, feature extraction phase, and decision phase. Also, the final goal of a collaborative system is to make it more automated by reducing human contributions. What humans can contribute to all of these cases are discussed below briefly.

2.2.1 Human Intelligence in Preprocessing Phase

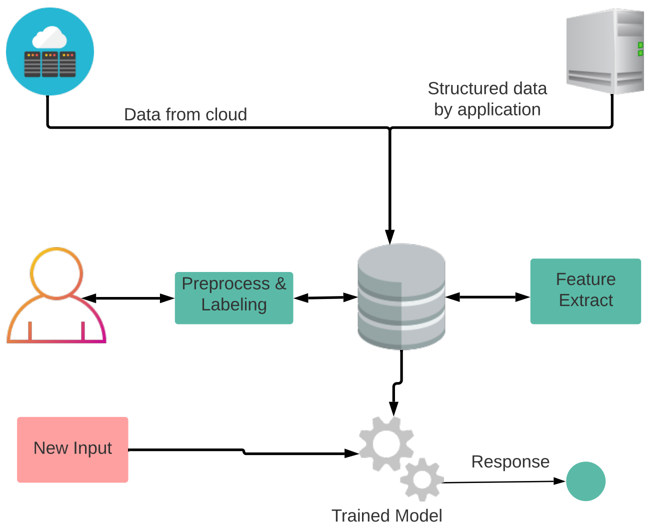


FIG. 2.4: Humans contribute to the data acquisition phase. Online available data are not complete, dirty, full of uncertainty, and unlabeled. Humans functionality is mainly in data acquisition and labeling phase. Individuals categorizes data into different classes and removes irrelevant data. The preprocessed data is later used by machine application to extract features, train model, and to provide services.

Data collection and labeling are the main bottlenecks of a human-machine collaborative system. Data storage and processing power have become widely available at a lower cost, which makes the ML technique more suitable for a wide variety of domains. However, supervised learning requires to have an adequate amount of labeled data, which are very hard to get for a specific problem. Deep learning, unlike traditional ML, extracts features from data automatically (Roh, Heo, & Whang 2018). However, deep learning requires to have a massive amount of categorized data. The data collection and acquisition process is different for different domains, so are their labeling techniques. The collaborative systems that have a significant amount of data can apply supervised ML techniques but require human intervention in data preprocessing, labeling, and categorization (see Figure 2.4). Traditional ML modeling uses human intelligence offline to annotate datasets. The preprocessing and labeling of data are different for different types of applications. For image classification, object detection, and scene understanding, individuals' knowledge helps to categorize images, label objects in an image, and detect the context of a scheme in the picture, respectively. In video activity detection, human intelligence helps to label a section of the video clip, identify any particular object movement, etc. In natural language research, transcribing a long audio clip is a vital step. In general business data, human knowledge helps to categorize data to a particular group or class.

2.2.2 Human Intelligence in Feature Phase

Data-driven application has become more popular because of its immense power of solving computationally hard problems. However, some problem domains do not have enough data to design machine only system. The number of problems that have data scarcity is significant. Different types of applications focus on different features of a dataset. The feature that has the most significant impact on the outcomes of that model may not be easy to extract from the instances. With a limited number of examples, getting insight from the dataset is challenging as the deep learning feature extraction is not possible. The obscurity and fuzziness of detecting and calculating features come from the complex nature of data instances. The very situation arises where human creativity and evolutionary experience are necessary to decide rather than computational equations. Individual cognitive skills are required to distinguish a few features, mostly in images, scene understanding, video actions, linguistic meaning, audio, etc. The application that does not have a significant amount of data can integrate human intelligence in extracting essential features that are not possible by computer alone.

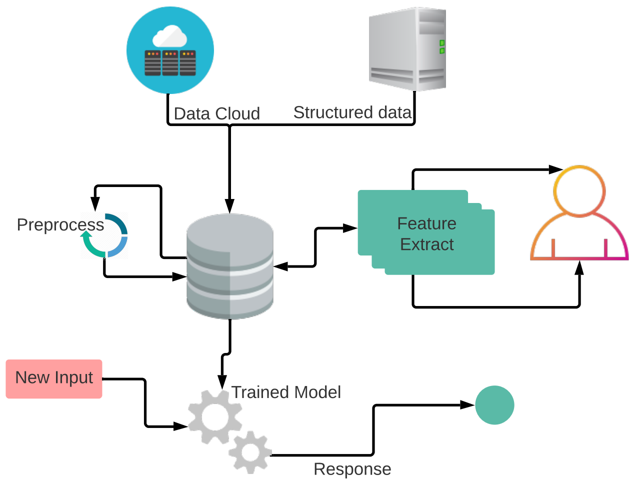


FIG. 2.5: Humans contribution in feature space. A set of data collected or acquired requires to detect and annotate features by humans. The data is used by the model to train or to build rules. The model is used to provide services for new input instances.

However, integrating human intelligence increases latency. A system with human intelligence can not respond in real-time as humans require a significant amount of time for that. However, If we want to design a real-time system using human intelligence in extracting features, the system can extract features offline to build a knowledge base from extracted features. The knowledge database will ensure the system of availability of features in real-time. A skeleton architecture of this type is depicted in Figure 2.5. This low processing and storage requirement makes the process suitable for embedded and sensor systems.

2.2.3 Human Intelligence for Temporary Solution

The idea of a human-machine collaborative system often provides solutions that prohibit total automation. However, the long term goal of the system is to increase data

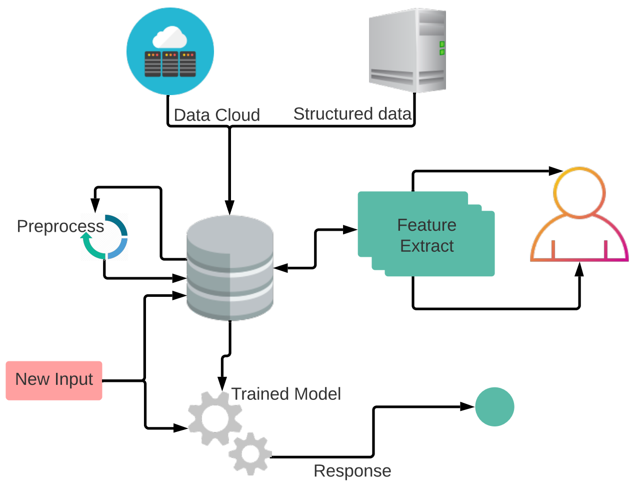


FIG. 2.6: Human contribution in feature space. A set of data collected or acquired requires humans to annotate features. The data is used by the model to train or to build rules. The model is used to provide services for new input instances. But, the new instance has the same features as of the instances used to train the system. The new input is added to the existing database incrementally and increases the dataset size.

and make the model more independent of human intervention and move towards more automation. In some cases, every instance represents a different set of features. Adding a new situation does not add repeated instances to the dataset. However, for most of the ML problems, a further example contains the same set of parameters that of the previous all cases. When a partially trained human-machine collaborative system serves a new query, it will add another instance to the training dataset. This property ensures that, over time, the database will grow. If humans are selected only because of the scarcity of data, the enriched dataset will be able to reduce human integration eventually. A skeleton architecture of this type of human-machine system is provided in Figure 2.6.

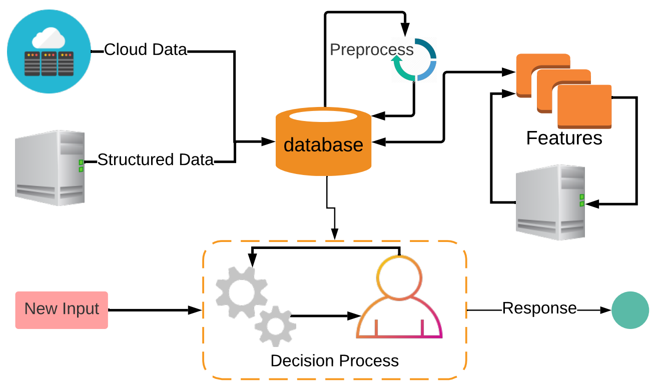


FIG. 2.7: Humans contribution in decision space. A well-organized dataset already exists, and features are already extracted. However, a decision-making module cannot work precisely without the help of humans.

2.2.4 Human Intelligence in Decision Phase

The researcher mainly focuses on automatic machine applications where a significant benefit comes from the astronomical number of data sets. A well-trained model with quality data can make the decision more accurate. However, in many sectors, we are confronted with a small collection of data samples or rare events. With this limited data set, the training process is not rigorous. Some of the problems are more intractable, even with a considerable amount of data. It requires human cognitive skills. For example, in health informatics and medical science, the decision depends not only on the data but also on domain expertise. Biomedical data sets are full of uncertainty and incompleteness, and some problems are hard enough to make it fully automated. The complexity of any intricate problem may provide probabilistic decisions, especially if the source of information is non-expert crowds or machine algorithms. Consequently, integrating domain experts can sometimes become indispensable, and integrating experts would greatly help in the knowledge discovery process. Machine, with its enormous power of computation, can help humans make a better decision (see Figure 2.7), helping to consolidate information within the human intellectual ability. The method also helps the machine to achieve the potential of making an accurate decision even with limited data.

The techniques for integrating human intelligence is different for different phase and conditions. We proposed various ideas to varying situations by developing four separate applications to provide comprehensive coverage of each category discussed above. We discussed those projects in detail in the following chapters.

Chapter 3

HUMAN INTELLIGENCE IN DECISION SPACE

In this chapter, we discuss a human-machine collaborative system that helps to identify weeds in crop fields. The system does not have enough data to train a machine learning model. The system takes human intelligence in the decision-making phase to overcome the multiple competitive outcomes that arise from an underfitting trained model.

3.1 Problem Statement

Weeds compete with crops for light, water, and nutrients. If left uncontrolled, crop yields are adversely impacted. Methods of weed control and management are constantly being updated, and more efficient practices emerge as a result of research, but farmers can only benefit from these advances if they have access to the most current information. The present extension approach to disseminate information on weed identification and their control practices for weed infestations relies on traditional methods such as extension publications, county-level meetings, and one-on-one consultations.

The above methods can be slow and unreliable for spread of relevant information; additionally, the scope of such methods can be too broad, forcing a farmer in need of specific advice to search through extraneous information or wait through unrelated meeting lectures in order to receive his desired information. In a social media driven world, farmers need latest information right there in the field. To satisfy this demand, extension agencies (Maunder & others 1972) are required to have access to trained manpower, both in the new technology and subject matter expertise. Economic downturn in the US economy has resulted in budget cuts in several land-grant universities thus, leading to a reduction in trained manpower. Given this reality, it is time to start thinking of innovative approaches for keeping extension and outreach services viable and useful by developing tools that could potentially be customized by extension agencies for use outside a state or a region. In this research, we use weed management as a test case to propose a novel identification and control system that could be used to augment existing programs. A typical weed identification call from a producer results in field visit by a county extension agent (Maunder & others 1972) [1]. The agent either completes the identification or refers to a specialist who in turn provides the necessary information. Unfortunately this manual process incurs high latency and increases the burden on extension service agents.

Smartphones have penetrated the rural population both in developing and developed countries. For instance, low-end Android phones that provide basic cellular data plans, a camera, and location information are common devices for farmers and extension agents. In a recent initiative in Arkansas, for example, the extension agents were provided with data plan-enabled iPads. We leverage this observation to design a system that can provide low latency, high accuracy, and low cost control practice dissemination to farmers when their crops are weed infested. At the core of our technique is a weed identification system that leverages the concept of crowdsourcing (van Etten 2011;

[1]Agricultural extension is a general term used for applying research in the field of agriculture through farmer education. The experts who facilitate agricultural extension are called extension agents.

Lowry & Fienen 2013) —using a human network to solve a computationally hard problem— and functions as a weed management information distribution system.

The initiator of the crowdsourcing effort would be the extension services of land-grant universities and the crowd consists of non-experts provided by services such as Amazon Mechanical Turk [2], and experts employed by land-grant universities. Such a system would benefit the farmers of the state who are in need of expert advice.

The design, implementation, and evaluation of our weed management architecture presents three novel research contributions. Our first contribution is a system that combines image analysis (machine intelligence) and crowdsourcing (human intelligence) to accurately infer the type of weed infecting a farmer's field. The technique uses a probabilistic decision engine to provide high accuracy, low latency and low cost inferencing. Our second contribution is the use of two levels of crowdsourcing, non-experts from services such as Amazon Mechanical Turk (Ipeirotis 2010a) and experts like extension agents. We propose another probabilistic technique to determine when the system is confident of its inference, and whether the second layer of crowdsourcing from experts is required. This helps minimize the use of experts, and can consequently reduce the cost of inferencing.

Our third contribution is an end-to-end system that includes smartphone applications for farmers and experts, and backend services that house the image processing and crowdsourcing logic. The present article evaluates the end to end system's latency, accuracy, and energy consumption characteristics.

[2]Amazon Mechanical Turk is a web service provided by Amazon that provides access to thousands of workers (Turkers) that can be used to solve tasks.

3.2 Design Goals

The goal for our weed identification system is to provide a low cost, low latency, accurate, and highly usable system to automate the weed detection and mitigation problem. We strictly adhere to the following design goals while implementing our system:

- Reduce the weed identification burden for extension agents and experts and provide high accuracy weed identification.
- Easy to use end-user applications: The adoption of the weed identification system is predicated on creating intuitive smartphone interfaces for both the farmer and the expert. To this end, we have designed our application interface using feedback from extension agents, farmers, and crop boards.

3.3 Methods and Architecture

The research presented here uses a combination of hierarchical crowdsourcing and image analysis to provide timely and accurate feedback to farmers requesting information on plausible weed infestation in their crop fields. The end-user, for example a farmer, is assumed to own a camera, and GPS-enabled smartphone. When the farmer determines that his crop is infested by weeds, he takes a picture using the smartphone camera. Our designed client application uses the picture, automatically geo-tags it with GPS coordinates, inserts any comments that the farmer may have, and uploads it to our backend database using a custom designed web-service. The web service first performs an image analysis that compares the picture of the weed with a database of weeds seen in the region in the past. The image analysis ranks the images in the database based on the overlap with the weed. The image processing acts as a low latency mechanism to determine which weed has infested the farmer's field, and also acts as a filter to narrow down a subset of plausible weeds that might be similar to the weed infecting the crop field. The web-service then uses two levels of crowdsourcing to identify the weed. First, it uses a set of non-experts, from a portal like Amazon Mechanical Turk to extract crude but low cost feedback on what the weed might be. We use a probabilistic decision engine that uses a majority vote on the results provided by the Turkers to determine if the weed is correctly identified. If there is no clear majority, the system consults the experts (extension agents). This helps minimize the use of the extension agents and reduces the overall cost and latency associated with the weed identification process. The decision engine is detailed in section 3.4.3. Once the system identifies the weed, it uses a database of control practices to provide accurate and up-to-date suggestions on how to mitigate the weed infestation.

Our weed identification system architecture is illustrated in Figure 3.1.

The system uses a combination of hierarchical crowdsourcing based on Amazon Mechanical Turk and experts, augmented with automated image processing, to accurately identify weeds and provide control practice recommendations to farmers. Interacting with this backend are the client-side mobile applications. We next describe our smartphone applications and the backend image processing logic.

3.3.1 Smartphone Applications

The system is comprised of two smartphone applications. The first smartphone application resides on a farmer's smartphone (see Figure 3.5). The farmer can use the application to take geo-tagged images of weeds in his field. These images, annotated with text and audio comments, are transferred to our backend server over a cellular or Wi-Fi connection. The second smartphone application resides on the expert's device. If our backend logic cannot reliably infer the weed species using image processing and the Amazon

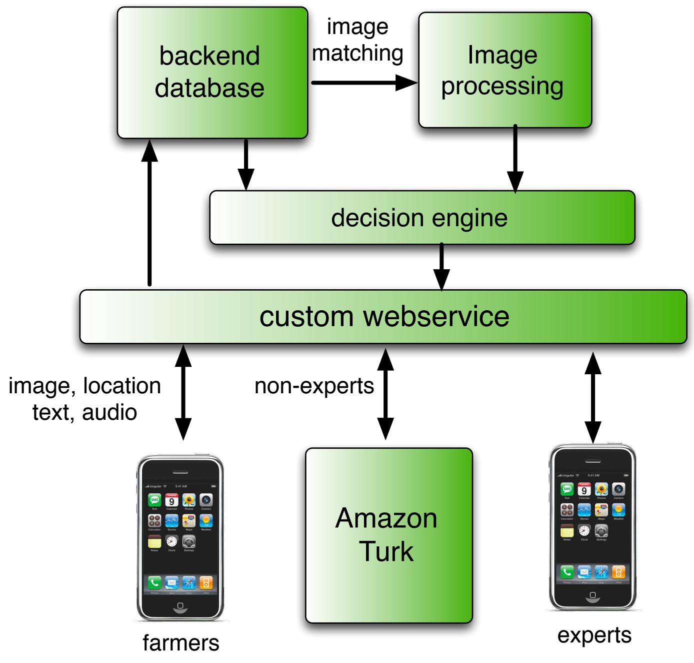


FIG. 3.1: System Architecture for our crowdsourcing-based weed management system. The system uses smart phone applications, automated image processing, and hierarchical crowdsourcing for weed identification and low latency dissemination of control practices to farmers.

Mechanical Turk crowd, it notifies a set of select experts, soliciting input on the type of weed. The smartphone applications (see section 3.5 for screenshots) provide features allowing the users to keep track of the requests submitted by the farmers, responses from the system and experts, and weed control practices. The app also features a weed database that the users can manually browse and compare weed images against.

3.3.2 Backend Image Processing Logic

The geo-tagged image and comments, sent by the farmer to the backend server is submitted to an automated image processing engine for identifying the weed. While image processing can uniquely identify images if reliable image features are available, it fails to correctly identify images in several cases. There are several factors that can affect the accuracy of an image processing algorithm. For example, if an image of the same weed is taken at different growth stages from different angles, at different orientations, or under different lighting conditions, the image could be mis-identified. Moreover, since farmers are not expert photographers, and off-the-shelf cell phone cameras are used to take the images, it is imperative to assume that the image quality may be suboptimal. Additionally, because our image processing engine compares the query image with a database of weed images, it is impossible to identify unknown invasive weed species or a new breed of weeds (Committee & others 2006). While the image processing engine may not be able to identify the weed images in several cases, it can, however, narrow down the search to a smaller subset of candidate images. Narrowing down the image search to a few candidate images is especially important because our system depends on a human crowd to identify the correct weed image. An Amazon Mechanical Turker, for instance, is unlikely to browse through hundreds of images in our weed database to identify the correct weed image. There might be scenarios when the image processing algorithm is unable to find close matches to the original weed image. For example, if the candidate weed image belongs to a new species, it will not be present in the weed database. In such a scenario, the image processing algorithm still outputs a set of images that are the closest match to the candidate image. These images, however, would be discarded by Amazon Mechanical Turkers as the weed image, and consequently, the experts would have to provide their input on the new weed species.

Our image processing engine, therefore, takes a preset number of images as input (n), and outputs the top n images that are a close match to the weed image provided by the farmer. We set $n = 5$ for our implementation. There are two reasons why we chose this $n = 5$. First, the lower bound on the number of images that needs to be displayed to the turker is 3 since the turkers rank the top three images. Second, we do not want to display a large number of images to the turkers. Prior research (Yan, Kumar, & Ganesan 2010) has shown that tasks that require turkers to browse through a large set of options before they can respond to the query produce poorer results than shorter and succinct tasks since many turkers use mobile devices to respond to queries. Our image processing algorithm uses a concept of multi-feature fusion to extract the best candidate set of five images that match the original weed image. The multi-feature fusion algorithm is illustrated in Figure 3.2. The algorithm works in two phases.

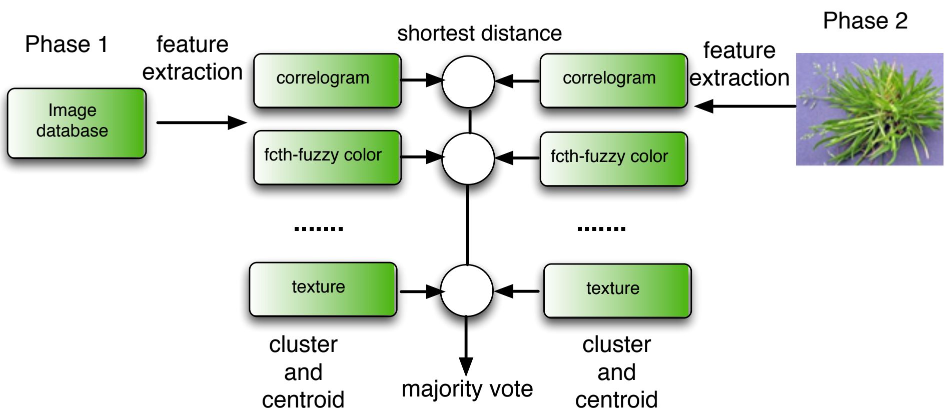


FIG. 3.2: The figure illustrates the two phases of the image processing algorithm. During Phase 1, the features are extracted, clusters are created for each feature, a centroid for each cluster is calculated, and the centroids are indexed in a database. During Phase 2, the same features and cluster centroids are calculated for the query image, and the distance calculated with the centroids in the indexed database. The top five images are calculated for each figure, and a majority vote is used to find the five images that closely match the query image.

- **Phase 1:** Phase 1 is an offline training phase. During this phase , our algorithm extracts multiple features from the database of images. Specifically, our system uses the following features: sift-scale invariant features, cedd color edge features, pixel correlograms, fcth-fuzzy color, and texture histogram (Lux & Chatzichristofis 2008).

We have chosen these features as they encapsulate a wide range of characteristics of weed images, and individually can capture leaf shapes, background color, plant texture, and edges characterized by plant stems. We use an open source lucene index-based image processing library called LIRe to build our feature extraction databases. The feature values for the images are clustered and the centroid of the cluster is stored as a byte payload in a lucene-based index (Lux & Chatzichristofis 2008). For instance, if the color histogram is a feature extracted by our system, three clusters are created for the Red (R), Green (G), and Blue (B) values for each image, and the centroid of each cluster is stored in the index database.

- **Phase 2:** During phase 2 of the algorithm (online phase) the query image from a farmer's smartphone application is processed and the same set of features as Phase 1 are extracted. The feature values are clustered and the center is calculated, like in Phase 1. For each feature f, the top five images $I_f = \{I_1, ..., I_5\}$ are selected with shortest distance between the center of cluster for feature f and the center of the same feature for the query image. To determine the five images closest to the query image, the system calculates a **rank**. The rank R_i for image i is equal to the number of sets I_f that it appears in. The image processing algorithm choose the images with the top rank. The image processing algorithm also outputs a probability P_I that the image is the same as the query image. P_I is the normalized rank of the image. Hence, if the ranks of the five images are $R_1, ..., R_i, ..., R_5$, $P_I = \frac{R_i}{\sum_{j=1}^{j=5} R_j}$. These five images are used as input to our novel hierarchical crowdsourcing algorithm that combines Amazon Mechanical Turk and experts to determine the weed that closely matches the query image, or registers a new weed species.

3.4 Hierarchical Crowdsourcing for Weed Identification

Humans can perform complex tasks using their vision and thinking capabilities, a process that remains difficult or sometimes impossible for computers to replicate (Shahaf & Amir 2007). Vision and image identification is one such application where identifying a scene or an object is a simple task for a human but is sometimes infeasible for computer vision algorithms. For instance, consider the weed identification problem. Two images of the same weed taken from different camera orientations, in different lighting conditions, or using different camera hardware may bear little semblance. An image processing algorithm that compares features extracted from raw pixels, therefore, may not be able to infer the similarities between the two images. An untrained human eye, on the other hand, can look at two images of the same weed taken in different lighting conditions and can easily infer that the images belong to the same weed species. The problem of weed identification, however, is exacerbated by weed images taken at different stages of weed growth. Consider Figure 3.3 that shows the same weed at different growth stages. To an untrained human eye, the two images might look different, however, a trained weed specialist or an agricultural extension agent can easily infer that the images belong to the same weed. Feedback from experts is however expensive given that the extension agents are already overburdened. There is a need, therefore, to build systems that augment image processing with untrained and trained human workers but use trained experts judiciously. To this end, our system uses a hierarchical approach to crowdsourcing described below.

3.4.1 Low Cost Non-expert Crowdsourcing

Amazon Mechanical Turk (Barowy & Curtsinger 2012) (AMT) is an Internet marketplace for human crowdsourcing. This marketplace enables a computer system, known as Requesters, to coordinate the use of human intelligence and machine intelli-

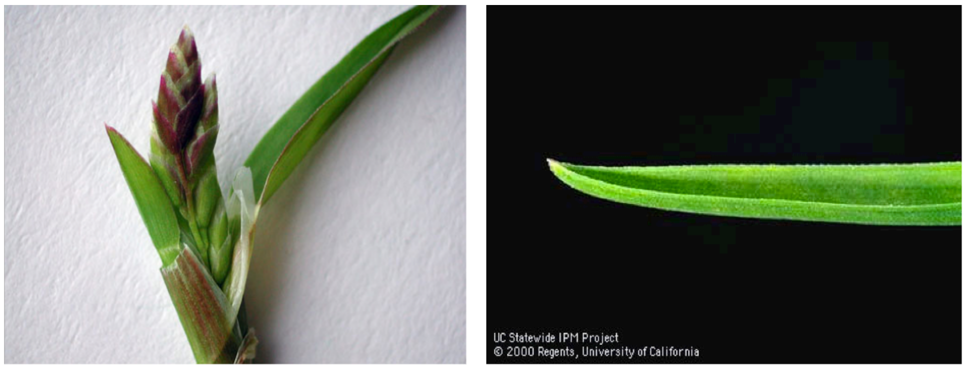


FIG. 3.3: The figure illustrates the same weed using images taken at different angle at different growth stages. It is difficult for a non-expert eye to determine that the images belong to the same weed.

gence to solve a problem that would be difficult or impossible to perform using only machine intelligence. The Requesters can publish tasks known, as HITs (Human Intelligence Tasks) (Ipeirotis 2010b), such as choosing the best images, writing product descriptions, run online surveys or identify performers on music CDs. Workers are called Providers in Mechanical Turk's Terms of Service, or, more colloquially, Turkers (Barowy & Curtsinger 2012). These workers can browse existing tasks for which they are eligible and complete them for a monetary payment set by the Requester. The monetary payment amount is preset by the Requestor and can vary depending on the task, but is always greater than \$ 0.005. However, the recommended minimum payment is 1c per task. The turk web services also provide tools to keep track of the workers and their performance. A Requestor can choose workers for the task based on past performance (Barowy & Curtsinger 2012). The time of publishing HITs, the time the HIT is accepted by a turker and when a response was received can be tracked using the Mechanical Turk API. In our system, we construct a request using our query image, and the five candidate images that are outputs from the image processing algorithm, and publish the request to the AMT webservice using a Java-based application interface that communicates with the Amazon Web Services. The detailed rules associated with the requests are also included to provide the workers with a better understanding of the questions before answering them.

3.4.2 Expert Crowd

The ***expert*** human crowd comprises of individuals who have in-depth knowledge about the weeds. We have recruited these agents from the State of Arkansas. In the hierarchical identification system, experts play the role of a top hierarchical object. In many cases, both the image processing algorithm and the non-expert turkers may fail to identify the query weed. Our probabilistic decision engine (described below) automatically infers if the system comprising of the non-expert and image processing algorithm is sufficiently confident that the weed identified is the correct one. If that is not the case, the system solicits input from the experts in our database using the expert version of our smartphone application.

3.4.3 Probabilistic Decision Engine

Our weed identification system works with two constraints: (1) a constraint on the *acceptable latency* that the end-user (farmer) can tolerate; and (2) a constraint on the *number of dollars* that the system can invest in crowdsourcing results from a public web-service like AMT and the time offered by extension agents. Throughout our system design, we assume that our system would be maintained by a government entity like the state, or private groups like crop commodity boards. In our system configuration, the image processing subsystem has been designed to have a low latency signature and zero dollar cost. However, the subsystem can be potentially inaccurate. For content based image retrieval systems, the average retrieval rate is less than 50% (Müller *et al.* 2001) if queries are applied to random images from a database. The expert crowd is assumed to have a substantial dollar cost with it when compared to using an non-expert crowd like AMT. As future work we plan to accurately model the cost of using experts to respond to image queries. The cost of using AMT is configurable and the manager of the system can determine the amount of money spent per request. The goal of our decision engine, therefore, is to calculate when the system is confident enough that a given weed has been correctly identified while satisfying the latency and cost constraints.

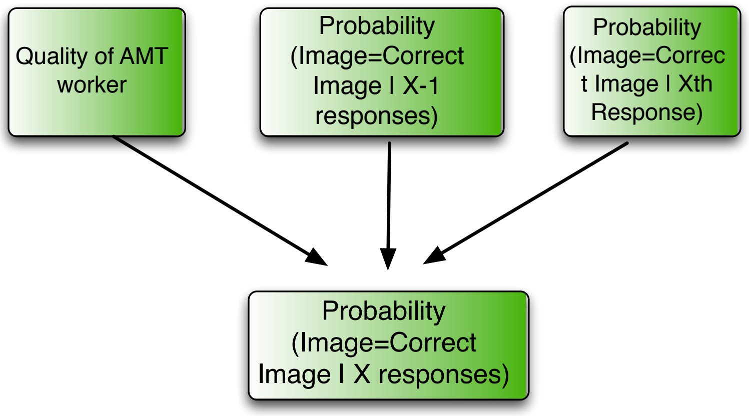

Quality of AMT worker
Probability (Image=Correct Image I X-1 responses)
Probability (Image=Correct Image I Xth Response)
Probability (Image=Correct Image I X responses)

FIG. 3.4: The figure illustrates the probabilistic decision engine that determines the probability that image I is the correct match for the query image after X responses. The probability is a product of the quality of the AMT worker, Probability that the image I is the correct match given $X - 1$ responses, and the probability that the image I is the correct match given the X^{th} response.

Our probabilistic algorithm, motivated by greedy Q-Learning (Watkins & Dayan 1992) is illustrated in Algorithm 1 (also illustrated in Figure 3.4). The algorithm takes as input the n images $\{I_1, ..., I_n\}$ that is the output of the image processing algorithm, a constraint on the dollar amount (D), and a constraint on the latency (T). The latency corresponds to the actual latency constraint set by the system or the farmer, after subtracting the estimated constant latency of using extension agents and the image processing subsystem. The dollar D is used to determine the number of HITs that are published to AMT. For our present implementation, we use a combination of 1c HITs and 5c HITs. The proportion of higher cost HITs can be tuned to match the latency constraint—we plan to pursue this problem as future work. Our probabilistic algorithm is executed every time a response is received from an Amazon Turker.

Table 3.1: Symbols used in Probabilistic Decision Engine

Symbols	Description
#R	Number of responses
I_i	i^{th} candidate image in the question
$Q(w)$	Quality of worker (scale of 0 to 1)
T	Required accuracy threshold
$P_{curr}(I_i)$	Probability of i^{th} image being correct in the current response
X	Current response number

```
**for** every response $R$ from AMT
**Calculate** $P(I_i|\#R = X) = Q(W) \cdot P(I_i|\#R = X - 1) \cdot P_{curr}(I_i)$
**IF** $P(I_i|\#R = X) > T$ **BREAK**
**end for**
```

Algorithm 1: Probabilistic Decision Engine (C, T)

To understand how the algorithm works, lets assume that our system receives the X_i^{th} response from Turk worker W_i, and has already received $X_1, ..., X_{i-1}$ responses from turkers. The system tracks the quality of every Turker W_i using a function $Q(W_i)$, where $Q(W_i)$ is the fraction of correct responses W_i has historically provided. To determine if a turker has provided correct responses in the past, we use a manual approach where an expert extension agent (co-author of our paper in our case) uses a web interface to verify whether the returned weed image was the correct one. To keep track of Turker performance, the system increments the worker's positive performance by one if the response is correct, otherwise the system increments the negative performance of the Turker by one. The quality of the Turker, $Q(W_i)$ is calculated as the positive performance divided by the number of times the Turker responded. Therefore, $Q(W_i) = 1$ represents a high quality Turker who has always provided the correct answer. Using $Q(W_i)$, the system calculates a probability $P(I_i|\#R = X)$, for every image I_i. $P(I_i|\#R = X)$ represents the probability that I_i is the correct weed image given the number of responses $\#R$ is X. The probability is calculated using Equation 3.1.

$$P(I_i|\#R = X) = Q(W) \cdot P(I_i|\#R = X - 1) \cdot P_{curr}(I_i) \tag{3.1}$$

where $P(I_i|\#R = X - 1)$ is the probability that I_i is the correct image after receiving $X - 1$ responses, and P_{curr} is the probability that I_i is the correct image given the X^{th} response. P_{curr} is calculated using a greedy theory technique proposed by Sheng et al. (Sheng, Provost, & Ipeirotis 2008), and is explained through the following example. The Amazon turker can rank up to three images out of five images that are provided through the HIT. However, the turker also has the option to rank fewer than three images. For instance, the Turker can rank one image as the exact match, or three images as Rank 1, 2, and 3. For the first case, we assign exponentially reducing probability to the images — 0.5 probability to rank 1 image, and distribute the rest of 0.5 to the four images (0.12 to each image). For the second case, the rank 1 image is assigned 0.5, the rank 2 image is assigned 0.25, and the rank 3 image is assigned 0.125, and the rest is divided between the two remaining images. Using this method of probability assignment, no image is assigned a 0 probability. Using this assignment, $P(I_i|\#R = X)$ is forced to be a positive value. The initial value of these probabilities correspond to the output of the image processing subsystem. Finally,

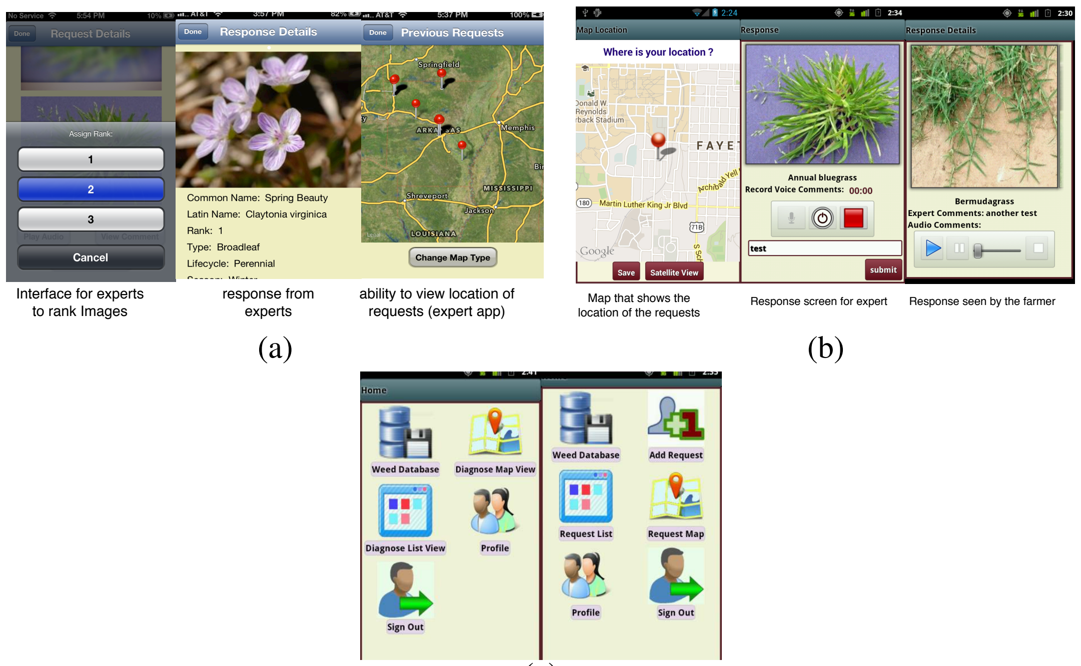


FIG. 3.5: (a) The Figure illustrates the screenshots from the iOS smartphone application. (b) The Figure illustrates the screenshots from the Android smartphone application. (c) Android application interface for the farmer and expert end of the application.

the probabilities are normalized such that $\sum_{\forall i} P(I_i) = 1$. If the probability for a particular image is greater than a threshold (set to 0.8 in our system, but is configurable), the system assumes that the correct image has been identified. If after the latency constraint (T) is exceeded, none of the images have a probability of greater than 0.8, the system solicits feedback from the extension agents.

3.5 System Implementation

Our smartphone application is available on iOS and Android. Figure 3.5 illustrates the screenshots of our smartphone applications. We also support a web-based application. The applications are written in Objective C and Java respectively for the two mobile platforms, while the web-based client is written using a PHP frontend. The image processing algorithm is implemented in Java. The weed image data and data requests made by the farmers is stored in a backend mysql database. Overall, we have written 8200 lines of iOS code, 19,665 lines of Android code, 10,000 of image processing and mechanical turk code and 3000 lines of backend webservice and processing code. We have implemented several optimizations including background download of images and caching of data on the smartphone to improve the performance of the system. We profile the memory and energy signature of the smartphone applications and latency characteristics of our backend in section 3.6.

3.6 System Evaluation

The goal of our hierarchical crowdsourcing system is to provide accurate and low latency weed identification at minimal cost. A complementary design goal of the system, therefore, is to reduce the burden on county extension agents and other weed experts in the identification process. Hence, our evaluation focuses on the following key questions.

- What is the weed identification accuracy of our image processing algorithm and the AMT crowdsourcing subsystem?
- What is the latency of weed identification for our hierarchical system?
- What are the trade-offs between the cost of publishing AMT HITs and the latency of turker responses?

We also present micro-benchmarks on the energy and memory consumption overheads of our implemented smartphone applications.

3.6.1 Experimental setup

Here, we discuss the the experimental setup for evaluating our end-to-end system. We break down the system into three components—image capture and upload using the smartphone applications, backend image analysis, and offload of images to Amazon Mechanical Turk—and rigorously evaluate each component. While we do not directly evaluate the accuracy and cost of using extension agents in the identification process, based on feedback from domain experts, we assume that both the cost and accuracy of extension agents' identifying weed images is high. To evaluate the AMT component, we created 70 instances of each requests. For each request, the cost of each HIT was either $1c$ per HIT or $5c$ per HIT. The HITs are comprised of two components—the candidate image and the top five images that the image processing algorithm selected from our apriori collected weed image database. On average, we received 66 responses. For our evaluation, we use three metrics—accuracy of identifying a weed image, the average cost of identifying a weed image using AMT, and the latency of identifying a weed image.

3.6.2 System Accuracy

In our first set of experiments, we evaluate the accuracy of the weed identification process using our system. Specifically, we examine the accuracy of our image processing algorithm and crowdsourcing subsystem. For our experiments, we used a set of candidate weed images chosen from our database of weed images and injected artificial gaussian noise to distort the images. The Gaussian noise added had a mean of 0.1 and variance of 0.2. This noise was added to the images since we donot have real time images of weeds captured by farmers. Therefore, every weed has two sets of images—weed database images and candidate images (with noise added). We selected 3-8 images of each weed based on their physical features, i.e physical color, shape of weeds and leaf structure, and availability

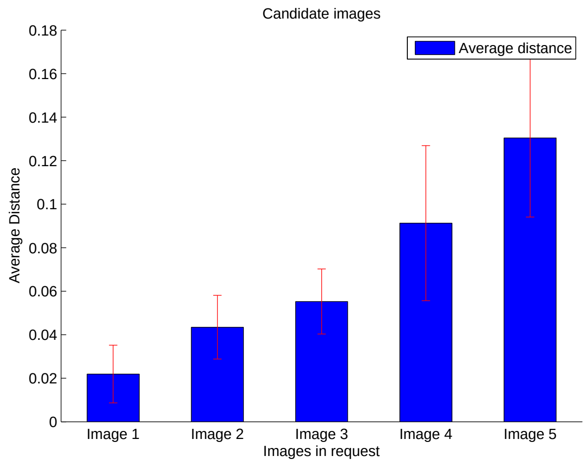
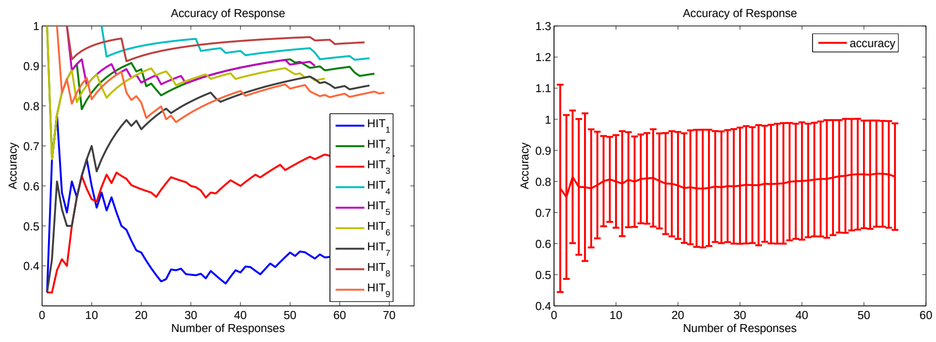


FIG. 3.6: The figure shows the distances calculated by the image processing algorithm for the top five candidates.

of images at different growth stages. Images were taken at different angles, lens exposure, and ambient lighting conditions. For our experiments, we were able to use a total of 70 weed types and approximately 5 images per weed type (the number of images per weed type varied from 3 to 7). Our dataset for this experiment included a total of of 300 images. For each candidate image, we execute our image processing algorithm described in section 3.3.2. The image processing algorithm identifies the top five images and ranks them by the average distance from the candidate image. Figure 3.6 plots the distance measured for the top five candidates across 9 distinct runs of the algorithm. From the experiment, we find that the image processing algorithm identifies the correct image in 89% of the cases in the top five choices. However, there is a considerable amount of uncertainty associated with the image matching algorithm identifying the closest image to the candidate image. As shown in the figure, the difference in the distance between the top candidate image and second best match is small. Hence, it is likely that the algorithm will get confused between two images and will not output the actual image as a top candidate. Since our image pro-

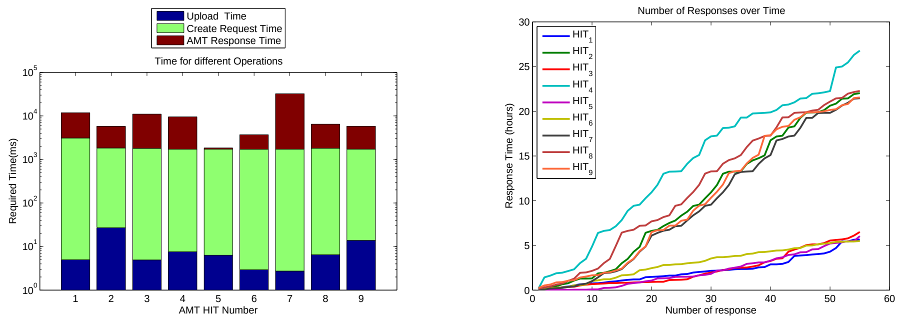


(a) (b)

FIG. 3.7: (a) The figure illustrates the cumulative probability calculated using our probabilistic algorithm for nine independent HITs published to Amazon Mechanical Turk. (b) The figure shows the average of the probabilities for the nine HITs. The error bars are calculated using standard deviation. The figure shows that 0.8 is a reasonable threshold for our application and the system requires at least 10 responses before making a reliable inference.

cessing subsystem calculates different set of features that are used for image analysis in several competing algorithms, their performance would be similar to the image processing subsystem in our system. Image processing, therefore, is insufficient for identifying weed images—the human crowd is required to solve this fairly complicated weed identification problem.

We next evaluate the accuracy of using the non-expert Amazon Turk crowd to identify the candidate image. Our HITs published to AMT allows each turker to rank up to three out of five images that are an output from the image processing algorithm. We use greedy theory to assign the probability, P_{curr} that weed image I_i is the best candidate image after the the X_i^{th} response. Based on this probability, we calculate $P(I_i|\#R = X)$, the probability that image I_i is the correct image after receiving X responses. We plot this probability, $P(I_i|\#R = X)$, for nine independent HITs published to AMT for the correct candidate image. Figure 3.7 (a) plots the above probability as a function of the number of responses for each of the nine independent HITs. Each line corresponds to the probability calculated by the system for the *correct* weed image. Figure 3.7 (b) plots the average probability (with error bars) calculated by our system for the correct image as a function of the number of responses for the nine HITs. From the figures, we can draw two conclusions. First, for most HITs, the probability of the correct image being selected improves slowly after ten AMT responses. Moreover, fewer than ten responses are insufficient since the transients are too noisy to make reliable inferences. Hence, the system should wait for at least ten responses before making a reliable inference. We find from the figure that as the number of responses increases the accuracy stabilizes. However, there is still some fluctuations as late responses are received. However, these late responses do not make the average accuracy fall below a threshold. Secondly, the figure shows that 0.8 is a reasonable threshold to determine when the system should decide that it is confident that the right weed image has been identified. From Figure 3.7 (a), we can also see that there are at least two out of nine HITs where the probability of selecting the correct weed image is below 0.5. There are several instances, therefore, when the untrained AMT crowd is unable to identify the correct image, and it is important to use extension agents. Since extension agents compare the candidate image with the entire weed database, they can perform accurate weed detection even if the error is in the image processing subsystem. In more than 80% of the HITs, it is possible to use the low cost AMT crowd service to get fairly accurate results without any intervention from extension agents. The 80% is calculated by finding the HITs were the accuracy is above 0.8 (illustrated in Figure 3.7 (b)). This can substantially reduce the cost of detecting weed infestations.



(a) (b)

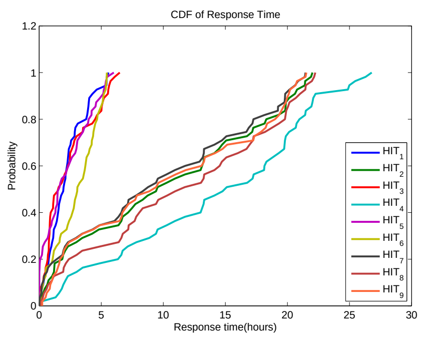
FIG. 3.8: (a) Time taken by different components of the system. The highest overhead corresponds to the AMT response time (b) The latency incurred by the Amazon Mechanical Turk responses are magnified in the figure. The figure shows the latency in getting responses as a function of the number of responses. The latency increases nearly linearly with the number of responses.

3.6.3 System Latency

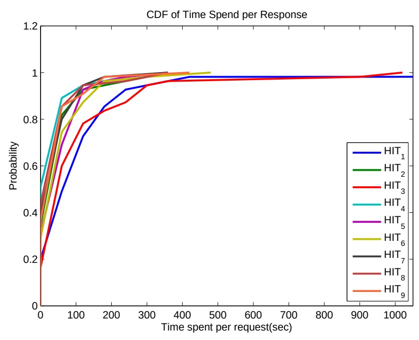
In our next set of experiments, we evaluate the latency of identifying the correct weed image. We measure the latency of capturing and uploading images to our backend server, analyzing the images using our image processing algorithm, and using Amazon Turk to identifyimages using an untrained human crowd. Figure 3.8 (a) presents a stacked bar graph that plots the amount of time taken by different components of the system. We use a threshold of 0.8 to determine when the correct weed image has been identified. The intuition behind the threshold is explained in the previous section. In the figure, we also plot the HITs that did not provide the right answer (HIT 1 and HIT 3) after 40 responses. For the other HITs a threshold of 0.8 was reached in fewer than 35 instance. For each HIT, we perform 20 iterations for the same experiment and report the average. The y-axis is plotted in log-scale. The "Upload time" corresponds to the time taken to upload the images to our backend server and is a function of the backhaul connection bandwidth to the Internet (over Wi-Fi). The "Create Request time" is the amount of time taken to publish a request to Amazon Mechanical Turk, and the “AMT Response Time” is the cumulative time of receiving sufficient responses from turkers that the system is confident that a candidate image has been identified. For a simple survey that requires less than 10 mins of turker engagement, which is similar to our posted task to mechanical turk, the normal response rate per hour is 5.6 if the turker is paid 2 cents per request. Our system in most cases can come up with a decision using an average of 20 responses per question while paying 2.7 cents per response. On an average, it takes less than 3 hours (minimum of 1 minute and a maximum of 25 hours) for us to correctly identify the weed image which is similar to the four hour time frame seen by other researchers in unrelated application domains (Buhrmester, Kwang, & Gosling 2011). We log the starting time and end time of our image processing system for each query and calculate the average time taken by the subsystem to output the top five images. We found that the average time taken by the image processing algorithm to determine the top five images is 1.9 seconds, which is minuscule compared to the latency associated with the other components of the system, hence the image processing numbers do not appear on the graph. The maximum latency is incurred in receiving responses from the AMT workers. We magnify this incurred time for each HIT in Figure 3.8 (b) that plots the amount of time taken by the system as a function of the number of responses for each HIT. From the figure, we find that the amount of time taken is proportional to the number of responses and increases nearly linearly for the first fifty responses. It is therefore possible to design a framework that takes a latency constraint as input and determines the number of responses that the system should accumulate before it must make an inference.

3.6.4 Cost Vs Latency Trade-offs

An important independent variable in our evaluation is the amount of money spent on each HIT by the system. Specifically, we study the effect of the amount of money



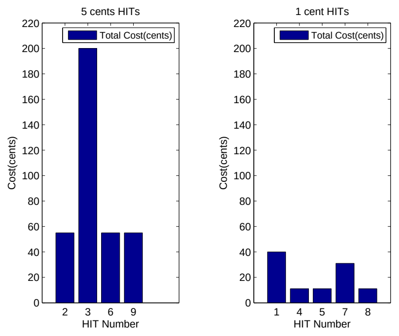
(a)



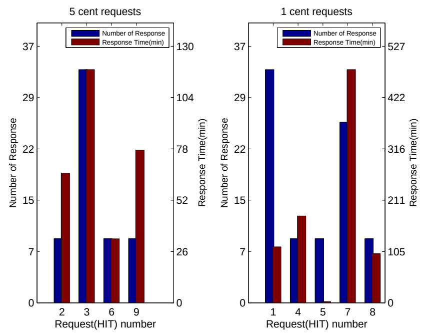
(b)

FIG. 3.9: (a) Distribution of the response time of turk responses (b) The figure shows the distribution of the time spent by turkers on a HIT.

spent per HIT on the time to solicit responses from turkers and the amount of time spent by turkers on generating responses. This is an important variable in our system since it helps explore the tradeoffs between the cost of publishing HITs and the latency associated with inferring the correct answer. We explore two cost rates for publishing HITs—$5c$ per HIT and $1c$ per HIT. Figure 3.9 (a) plots a cumulative distribution function of the amount of time taken for a response, and Figure 3.9 (b) illustrates the cumulative distribution function of the amount of time spent by a turker on a HIT. The lines that correspond to the $5c$ and $1c$ HITs are annotated in the graphs. From the graph we find that the median response time for $5c$ HITs is one-third of the $1c$ HITs. However, the amount of time spent by turkers on these HITs is equivalent (as shows in Figure 3.9(b)). To help understand this observation, we present Figures 3.10 (a) and (b). The figures illustrate the total amount of time taken per HIT, the number of responses, and the cost of publishing the HITs. It is clear that the amount of money spent on the HITs is higher for the $5c$ case while the total number of responses received are identical—this is because a minimal number of responses are required to make an accurate inference. However, these results also show that turkers



(a)



(b)

FIG. 3.10: (a) Cost in cents per HIT for $1c$ and $5c$ requests. (b) Average response time and number of responses for each HIT.

prioritize HITs based on the amount of money they earn per HIT. Hence, a HIT with a larger price tag will be attended by the turker sooner leading to lower overall latency which in turn leads to a higher overall cost. As future work, we plan to utilize this insight to design an optimization framework that can determine the optimal price tag for HITs such that a latency constraint can be met.

3.6.5 Micro-benchmarks

Finally, we evaluate the overhead of our system with respect to the energy consumed by our smartphone applications and the amount of memory utilized by the phone applications, both for the expert and farmer versions. Specifically, in our experiment we log the battery depletion as a function of time on the Android and iOS platform as we use our crowdsourcing application. Figure 3.11 illustrates the energy consumed when creating a request at the farmer end and uploading that request. We performed the experiment across twenty different requests. From the figure, we find that the average energy consumed per request is around 20 J which is equal to 1.7 mAh. Hence, a single request on a 1500 mAh

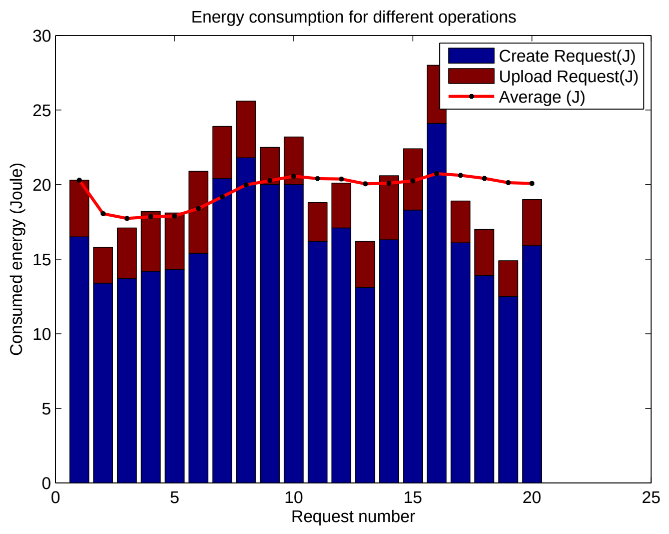


FIG. 3.11: Energy consumed in performing different operations at the smartphone application. The unit of energy is Joules that is equal to the energy transferred (or work done) when applying a force of one newton through a distance of one meter.

battery (common battery size for iPhone and Android phones) can reduce the battery lifetime by 0.1%. Since a farmer is unlikely to upload more than 2-5 requests per day, the energy overhead of our system is low. In terms of memory usage, the Android application uses between 54 MB and 105 MB, and the iOS application consumes between 5 to 20 MB of RAM. Most of the memory overhead is due to the caching that the application performs with images in the database to minimize latency.

3.7 Conclusion

In this chapter, we present a hierarchical system that uses smartphone image capture applications, a backend image processing algorithm, and two levels of crowdsourcing to identify weed images. The system provides low latency, low cost, and accurate identification of invasive weed infestations. Such an automated system can help reduce the loses caused by the delay in identifying weeds, and hence, lead to quick remedial control practices applied to contain weed infestations. The two crowdsourcing levels consist of a non-expert inexpensive AMT crowd and a set of expert county extension agents. We propose a probabilistic decision engine that determines, in an unsupervised way, whether inputs from extension agents are required for weed identification or whether the low cost amazon turk crowd is sufficient to identify the weed image. We evaluate our end-to-end system using real weed images and show that the system can provide accurate weed identification within 3 hours while minimally using the extension agents. Moreover, the system can provide highly accurate weed identification. As future work, we plan to use the insights gained from the economics underlying the crowdsourcing approach to design a cost model-driven optimization framework that takes latency and cost constraints and determines the best possible candidate image. Additionally, we plan to leverage additional contextual data associated with the images to improve the machine intelligence (image processing) component of the system.

Chapter 4

HUMANS IN THE PREPROCESSING PHASE

We have designed and developed an end to end system of detecting human conversation from a group of people in workspace. In this system, the database is significantly big, but the computer algorithm is not able to label speech segments from the group conversation in unsupervised training. Humans can help machines annotate the data so that machine algorithms can handle the rest of the steps. We discuss the details steps of how to integrate humans in the data processing phase.

4.1 Problem Statement

Language is the light of a human mind, and speech is a fundamental part of any language. It is also the best way of social interaction, interpersonal and group conversation (Maynard & Peräkylä 2006; Liddicoat 2011). A conversation is key to language development and the exchange of ideas and thoughts. In addition to these, people meet at various locations to discuss different issues, agreements, and job functions. A deep understanding of any conversation's insight helps to improve communication skills and workplace productivity. We proposed an ML-based conversation detection method using strain gauge respiratory sensor data.

Conversation understanding research started in the early 70s (Schegloff 1992) which mainly focus on speaker identification (Lu *et al.* 2011; Ahmed, Kenkeremath, & Stankovic 2015), group size determination (Xu *et al.* 2013), emotion, stress level determination (Lu *et al.* 2012; Rachuri *et al.* 2010), and to find out the contents of a speech. However, most of those studies were based on audio and video recording of speakers and analyzing those contents. Audio and video-based conversation analysis depend on the utterances of the speaker, which relies on the position of the audio input device, amount of oscillation, and standard environment for audio and video recording. Also, posing to a microphone or camera has the potential of psycho-physical stress on the speaker, and may influence not to take a turn when a speaker has the urge to speak (Duncan 1972; Kumar *et al.* 2014; Mengis & Eppler 2005). In addition to these, audio and video recordings have the issue of privacy concern to the subject (Klasnja *et al.* 2009).

Conversation analysis with wearable sensors can address all of these limitations. Researchers over the years have used an automated recording of social interaction using various wearable devices (Olguín & Pentland 2008; Charan 1991) to see the effect of group conversation in the workplace. They used sensors to analyze different metrics on speech dynamics, i.e., the average time of speaking during a discussion, the percentage of time a single speaker participates in that conversation, etc. They applied the wearable device called sociometric badge in different types of organizations, including banks, call centers, and hospitals. They found that informal conversation has a significant impact on teamwork and management processes in these organizations. The productivity of any creative teamwork varies a lot based on casual communication among team members (Waber *et al.* 2010). The everyday conversation has a high impact economically, even in online call centers. Researchers also have used wearable devices to analyze customer behaviour (Kim *et al.* 2009) and behavioral health (Madan *et al.* 2010). However, these wearable devices are intrusive and uncomfortable to wear.

While speaking, the mean breathing rate decreases but also becomes more variable, especially in expiratory duration. The utterance also affects tidal volume (Lorig 2007). The ratio of inhalation and exhalation duration reduces during speech compared with normal state. Thus, these breathing dynamics can be a good source of information to disambiguate between speech and silence. Though research on the conversation is decade old, respiration-based conversation analysis is not explored to a great extent. Respiratory based conversation analysis not only remove the existing issues of privacy, but it will also open the door to another domain, i.e., stress analysis. Wearable sensors provide information about the complex dynamics of social interaction as they occur in the real-world (Yamada *et al.* 2011). A few respiratory measurement sensors are commercially available, as well as for research. Wearable strain gauge (SG) sensor, although typically utilized to measure respiration, can potentially index the dynamics of speech. This sensor is relatively affordable and non-invasive, perhaps allowing researchers to identify: (a) when someone is talking, and (b) who is talking in an audio recording of group interaction if group members are wearing strain gauges.

Respiration strain gauges are composed of a pressure transducer that is placed above the sternum with a belt that wraps around the upper thorax. This pressure transducer most directly measures chest movements that are strongly correlated with respiratory changes, which are only suggestive of actual breathing dynamics (timing of inspiration/expiration, tidal volume, etc.). Thus, it is reasonable to speculate that strain gauge data could provide rich information about physical processes that affect chest movement other than breathing (e.g., speech, sighs, gross motor movement,etc.). Speech can severely confound estimates of respiratory dynamics derived from the strain gauge signal. This

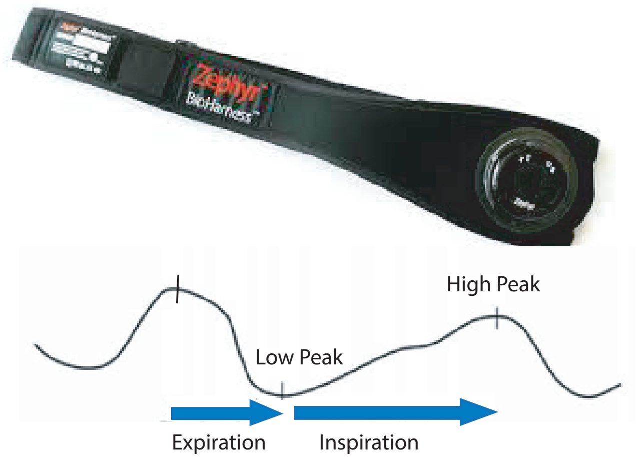


FIG. 4.1: Zephyr bio-harness sensor can sense the change in the tidal volume of a human chest during inspiration and expiration, which can help to identify breathing patterns.

confounding occurs in two significant respects, a) Speech strongly affects the strain-gauge "respiration" signal, and b) Speech adds substantial noise to the voltage-time series extracted from the strain gauge pressure transducer. Such noise can be characterized as motion artifact– i.e., the relatively gross movement of the transducer on the chest resulting from typical speech. This noise typically obscures researchers' ability to accurately measure the morphology of the SG signal and, in turn, biases estimates of respiration rate and tidal volume. Though the strain gauge shows a strong correlation to speech dynamics, the added noise poses a significant challenge to use it for conversation analysis.

The experiment design, workplace setup, and training a speech dynamics analysis project presents two novel research contributions. First, we collected a significant amount of group conversation data with real-time workplace setup. Second, we developed a machine learning model that can detect who is speaking at a specific time in a group conversation with 98% accuracy.

4.2 Data Collection

The collection of high-quality data from a workplace, a real workplace setup is critical. Also, it is not possible to tag sensor data as speech or silence directly using only sensor data. However, if we have the audio stream together with the sensor stream, we can know the segment of speech or silent from the sensor data stream comparing with the audio stream. It is also essential to record both audio and sensor streams in the same time frame.

4.2.1 Workplace Setup

We set up a workplace precisely similar to a regular workplace. The designated room consists of four different desks equipped with a workstation, high oscillation free microphone, and a Samsung note 7. For further improvement of sound quality, we divided the room into four separate cubicles, and each cubicle contains a single setup. This setup also guides the participants to talk with each other online and improves recorded audio quality. We used Zephyr Bio-harness sensor (see Figure 4.1) to collect respiratory responses of each participant. It is a little bit uncomfortable to wear Zephyr around the chest, but this is widely being used to collect respiratory data (Monowar Hossain & Hnat 2019 accessed July 8 2019). Each Bio harness sensor directly connects to its corresponding smartphone. All four smartphone connects to the central server. We also connected all four workstations to the same central server to store recorded audio and sensor data of all four participants in a structured way. The central server also syncs timestamps to all four smartphones, as well as all four workstations. We developed software to generate the same LSL (lab stream layer) for all four channels. It also makes sure to store streaming data from all eight channels together for each session in key-value pair based streaming data.

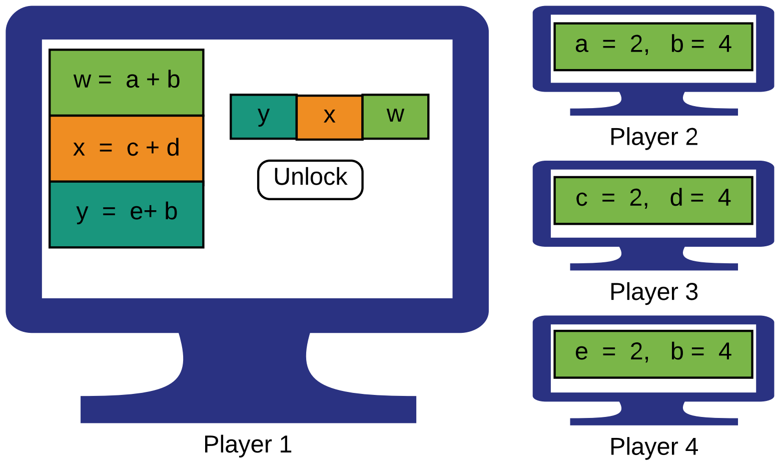


FIG. 4.2: Each player have their own screen with different information. Player 1 requires the value of a, b, c, d and e from other players. Other players have the answers which they can use to help player 1 to unlock the puzzle.

4.2.2 Group Conversation Data

People in a workplace can make group conversations for sharing ideas, business roadmap, brainstorming, project analysis, entertainment, and many more. A good workplace where all members participate creates a lot of new ideas, tackle problems, make collective improvements, etc. To simulate a real workplace environment, in addition to workplace arrangement, we also provided the participants tasks that keep them engaged with one another. We encouraged them to play two computer games- bomb-game, and empty-epsilon. The sole purpose of these two games is to keep everyone engaged while playing or solving problems together.

Bomb-Game

Bomb-game is a puzzle game. The bomb defuser of the team has to defuse a bomb with a secret key. However, the secret key is not readily available. Instead, it can be generated based on different information residing in all four computers. Based on the bomb type, manufacturer, country, and many other criteria, the team as a whole has to find out the key. The defuser will ask each of his teammates for different codes based on the various parameters of the bomb. Each member's computer contains a manual based on their expertise. Based on the question, the member will look into the manual, find the appropriate parameter and code, and will tell the defuser. Once the defuser gets the full code, s/he can enter that code to defuse within a predefined time. If the defuser takes more time than the defined one, it will explode as well. A simple example of the game is explained in Figure 4.2. All the communication happens through an audio channel as the participants are sitting at their workstation. The expiration time makes sure that the work is not always relaxing that can capture audio of a different mode of the workers.

Empty-Epsilon

Empty-epsilon game (daid 2020) is a space combat game. Four people -a captain, a helms, a scientist, and a weapons officer, participate in space combat when exploring the sky with a space ship. The helms search for enemy ships scanning all radar signals. If it captures any new object, s/he inform the captain about the probable attack. The captain asks the scientist who has access to a database of all ships, their size, power, weapon, and origin. Based on the information, the captain orders the weapons officer to load appropriate ammunition suggested by the scientist. Once the enemy ship is inside the circle of attack, the soldier attacks the ship. All four participants have their expertise and information and they work as a team. All the communication happens through a single audio channel.

We hired three different teams of four members and trained them with a session on how to play these games while wearing the respiratory sensor and communicating with

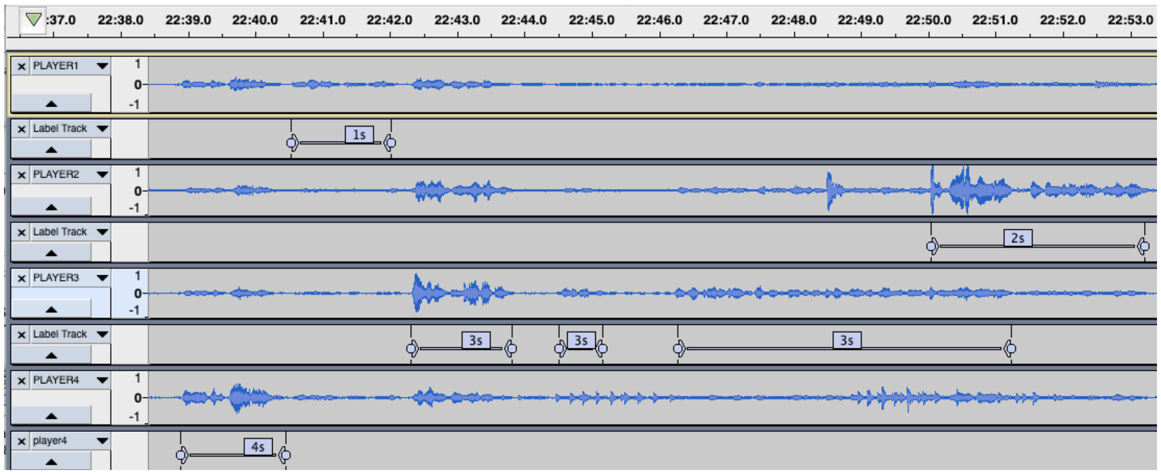


FIG. 4.3: An example of recorded audio annotation from a group conversation using audacity. Each audio stream can have their own label stream. Annotated segments can be edited at anytime and can also be exported to a single text file.

one another with the microphone. The age of those participants was between 18-40. To avoid the effect of individual intellectual ability, we randomly rotated the role of each person in each session. Each session was about 30 minutes long, and we recorded ten sessions for each team. The database thus consists of 30 hours of audio and sensor data.

4.3 Data Annotation and Preprocessing

The data consists of two separate LSL streams- audio stream and respiratory sensors stream. The two streams are recorded at two different frequencies, but they are time-synchronized. The starting time and end time of each stream are similar. A temporal segment in the sensor stream represents the same segment in the audio stream. The audio stream is in human-readable form as we can use any software to listen to the audio to determine the parts that represent speaking and which segments represent non-speaking. We can label sensor data using the audio annotation as the time segments are synchronized.

4.3.1 Data Cleaning

The respiratory sensor is susceptible to movement as it gives the value based on chest stretching. It senses the change of a human chest when a person talks. However, other movements of a human affect the sensor as well. It is not typical that a human will do nothing while speaking. In our workplace settings, all participants play team games with lots of excitement, decision making, and do lots of other daily activities like eating snacks or drinks coffee, etc. All these activities affect the sensor. All these activities produce lots of noise. The data collected in this way requires to be purified to get the actual changes that are related to speech-related movement only. Some events, i.e., movement of the chair to using hands that produces sudden unwanted movement, creates spikes. However, the other actions create regular changes to a sensor signal.

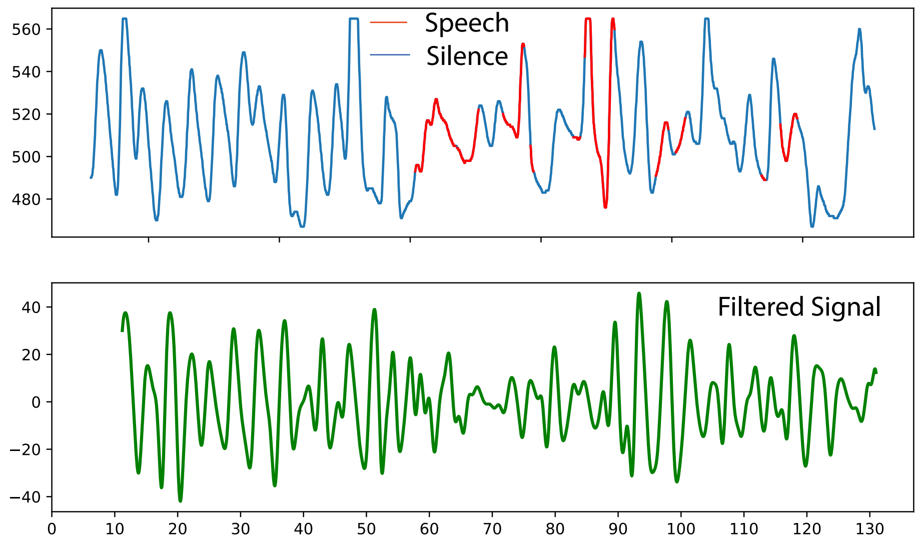


FIG. 4.4: Bandpass filtered signal clearly shows the effect of speech and non-speech in a respiratory signal. During the silence period, the signal remains regular but during a speech, the shape of the signal changes.

We applied a Butterworth bandpass filter to get rid of these noises and keeps signals that are related to chest movement only. Humans breathing band $f_b = [f_s, f_e]$ range from twelve to sixteen per minute during the normal time. It can go up to forty-eight to fifty per minute during extreme activities, i.e., exercise and workout. However, the frequency can go as low as nine breaths per minute when a person speaks. To capture the whole frequency range that represents the human chest movement, we set the bandpass filter range $[f_s = 9/60, f_e = 50/60]$. The plot in Figure 4.4 reflects the speaking effects on the respiratory signal.

4.3.2 Sensor Data Labeling from Audio

Annotated audio data streams sampling frequency is $f_a = 44.1KHz$. However, the sampling frequency of the sensor data stream is $f_a = 44.1KHz$. Both streams are synchronized. To annotate sensor data using audio, we need the starting data points and end data points of a segment from the sensor data stream given the segment of the audio data stream. We used a simple equation to convert any data points from the audio stream to the sensor data stream. Let's say N_a is the N^th data point in the audio stream, and N_s is the corresponding data points in the sensor stream. Then $N_s = N_a * f_s/f_a$ where f_s is the sampling frequency of the sensor signal while f_a is the sampling frequency of audio data. For example, if an annotated audio segment is $(88200, 176400)$, the corresponding sensor segment is $(36, 39)$.

The annotation process tagged only temporal segments related to speaking only. In this case, we only have two classes- speaking and non-speaking. During speaking, a person takes a short break to keep their breathing regular. We plot the human annotations of speaking in figure 4.5, where the red color signal represents speaking, and blue represents silence. The typical gap between two speaking segments varies from zero to six

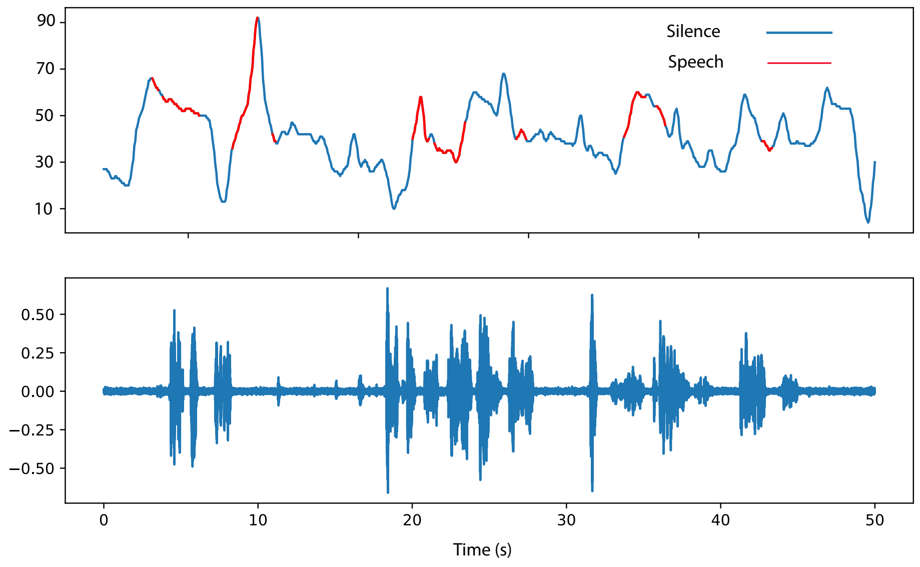


FIG. 4.5: Labeling of respiratory sensor stream based on audio annotation. Each audio stream has the current player's vocal together with other players speech signals. Respiratory sensor is labeled based on audio annotation.

seconds (Jepson 2005). We can merge two consecutive sections if the difference between them is less than six seconds. Once we get the list of all segments related to speaking, in the next step, we can invert the annotation to get the tagged temporal parts for silence. The final annotation database was created merging the above two lists of segments.

4.4 Feature Extraction

The shape of the respiratory sensor waveform changes with the change of chest movement of the subject. A transition from an exhalation point to inhalation defines the actual difference of chest movement during inhalation. Similarly, a transition path from a pick to bottom represents the change during exhalation. Quantization of the signal enables us to calculate the characteristics of chest movement (Lorig 2007). However, it only captures a single movement avoiding the relationship from one move to the next. A sliding window-based instead of single cycle-based classification might capture the characteristics of human speaking dynamics.

4.4.1 Window Size, Creation and Labeling

Window size is vital to capture more granular characteristics. A single person speaking behavior study uses window size of thirty seconds. However, humans' breathing rate can go as low as nine breathing cycles per minute while speaking. A thirty-second window can capture four and a half breathing cycles. We compared thirty, forty-five, and sixty seconds window experimentally and found that sixty seconds window capture enough characteristics and improve the accuracy. Previous research has shown that a 50% overlapping in the sliding window performs better (Van Laerhoven & Cakmakci 2000), and we adopted that idea in our proposed model.

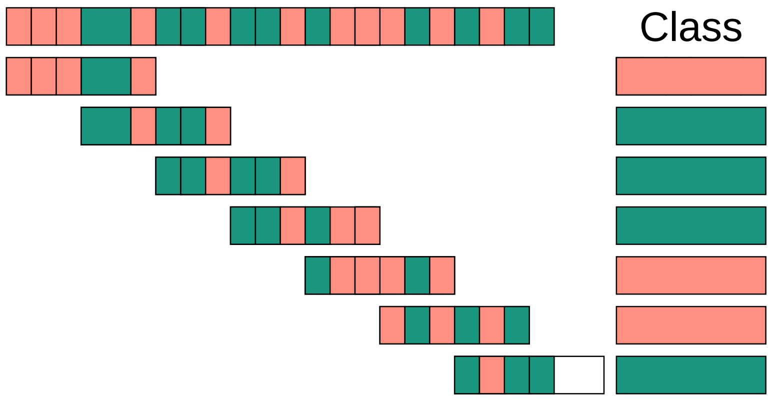


FIG. 4.6: The sliding window slides a specific percentage from one window to next. For each window, the portion of speech and non-speech majority defines the label of that window. This figure applies 50% overlapping and majority algorithm to create the window and assign labels.

Another critical aspect of speech analysis is the label of each window. Speech analysis is different from other wearable sensors in the sense that the frequency is dependent on human breathing frequency. The length of a single speaking segment varies from a few seconds to minutes. It is tough to capture a full speaking segment in a window or to fill a single-window with either only speaking or only silence. In a group conversation of four people, at least three fourth of the time, people remain silent or listen to others. We experimented with threshold 75%, 60%, 50%, and 40% rule that defines a specific window as a speech segment if the total percentage of speech crosses the threshold. The number of windows that cross the threshold of 60% is significantly low. The threshold 50% perform better in classification. The algorithm of creating and labeling window is explained in **Algorithm 2**.

Detecting Breathing Cycle

If a respiratory sensor signal is the result of a regular breathing pattern, a threshold-based algorithm works well (Figure 4.7). It detects a breath when a sensor signal's waveform passes a predefined threshold either in rising or in falling direction. However, the defined threshold varies for different persons and varied situations. It is tough to find a gold standard threshold value for a diverse population and various speech dynamics. In addition to these reasons, the Zephyr bio-harness sensor is susceptible to the human movement that regularly occurs when a person speaks. The spurious peak in the respiratory sensor is very common because of these reasons. We implemented the peak detection algorithm proposed by (Duarte 2015). From the empirical analysis of our collected data, we found that with a minimum peak difference (MPD) of 10 and a threshold of 0.01 unit works best and produces less spurious peaks.

We calculate all maxima and minima accurately by executing peak detection algorithms by two iterations and removed spurious peaks at the third iteration. The first iterations detect all peaks, and the second iteration detects all valleys. In the third iteration, we

```
Input: A list Ł={$(t_s, t_e, l)$} segments tuples, where $t_s$ is starting time, $t_e$ is the
       end time, and l is the defined class of the segment
Output: A list $L_t$ of equally sliced tuples of segments {$(t_s, t_e, l_r)$} where $l_r$ is
        defined label
$w_s \leftarrow \emptyset$
$w_e \leftarrow w_s + ws$
$s_t \leftarrow \emptyset, c_t \leftarrow \emptyset$
$N \leftarrow end(l_n)$ where $l_n$ is the last segments of Ł
while $c_t \leq N$ do
    while $end(c_t) \leq c_t$ do
        if $c_t$ = $s_t$ then
            $C[class(c_t)]$ = $end(c_t)$-$w_s$
        else
            $x = end(c_t) - start(s_t)$
            $C[class(c_t)]$ = $C[class(c_t)] + x$
        end
        $c_t = c_t + 1$
        if $w_e > end(c_t)$ then
            $v = w_e - start(c_s)$
            $C[class(c_t)]$ = $C[class(c_t)] + v$
        end
        $l_r = maxind(C)$ $L_t.append((c_s, c_t, l_r))$
        $C \leftarrow \emptyset$ $w_s = w_s + ws - o_l$
        $w_e = w_e + ws + o_l$
        while $start(s_t) > w_s$ do
            $s_t = s_t + 1$
        end
    end
end
return $L_t$
```

Algorithm 2: WINDOW labeling sensor data (Ł, ws, o_l).

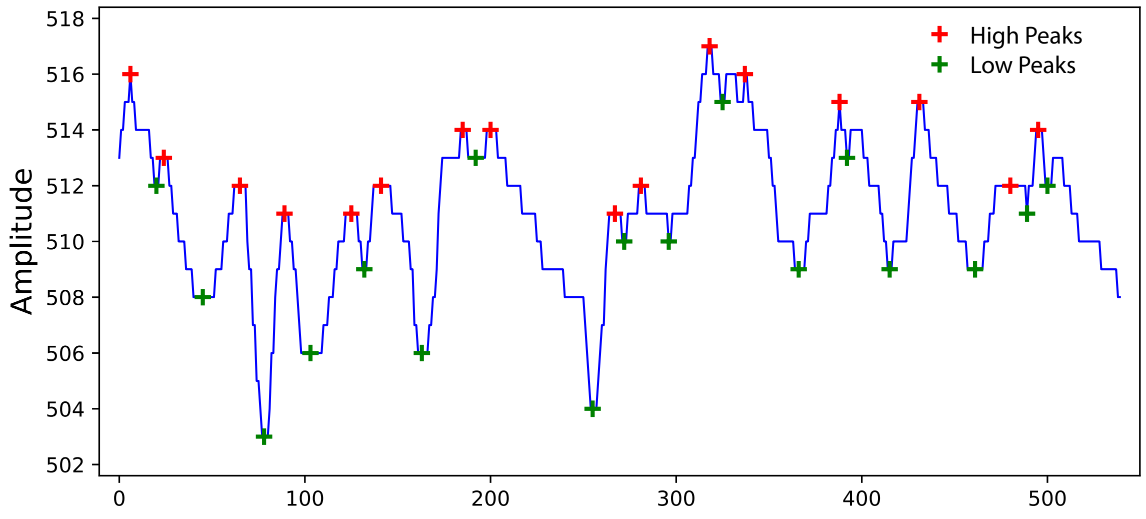


FIG. 4.7: High and low peaks are detected using peak detection algorithm. The high and low peaks are merged to get all peaks.

compare the time-stamps of consecutive valley points. If there are multiple peaks between two valleys, the algorithm takes the one with the highest peaks. The algorithm follows the same procedure to remove spurious valleys as well. The algorithm returns time-stamps merging all these peaks and valleys. We plotted all the signals with the peaks returned by the algorithm, visually observed peaks for 1 hour of sensor signals. This algorithm can detect the peak points from the breathing sensor with an accuracy of 98%. A small portion of the results is plotted in figure 4.7.

Breathing Cycle Features

The inspiration duration (T_i) is the time to take air inside the lungs, and the expiration duration (T_e) is the time to remove air from the lungs. This activity reflects on the sensor signal, which takes upward slopes during inspiration and takes a downhill during expiration. The expiration and inspiration of a respiratory sensor signal and the transition path from one state to the other can provide the breathing characteristics. The quantization of these three parameters enables to capture the speech-related features. These measurement provides both temporal and spatial properties. Previous researches on respiratory sensor signals used these to engineer signal features (Bari *et al.* 2018; Rahman *et al.* 2011).

- *Inspiration duration((T_i)):* The duration to take air inside the lungs.
- *Expiration duration((T_e)):* The duration to remove air from the lungs.
- *Inspiration-expiration duration ratio((T_i/T_e)):* The ratio of T_i and T_e.
- *Inspiration volume(A_i):* The volume of the air drawn inside during inspiration. Integration of the uphill slop provides this information.
- *Expiration volume(A_e):* The amount of air it leaves during expiration. Integration of the downhill slop provides the expiration volume.
- *Inspiration-expiration volume ratio:* The ratio of A_i and A_e
- *Inspiration-expiration Stretch:* The respiration phase can also be different at different breathing cycles, which pose different stretch on both peak and bottom. We have calculated the stretching of these points at each period.

We derived the mean, median, and standard deviation from these characteristics.

Window-Based Features We proposed a window-based speech classification model that makes the window level features as useful as the cycle based features. Time series analysis is a matured field of study. Both time domain and frequency domain features are consolidated in TSFresh (Christ *et al.* 2018). Some of those features are- median absolute deviation (mad), energy, inter-quartile range, signal magnitude area, skewness, kurtosis, the entropy of power, and the spectral centroid of frames, etc.

4.4.2 Feature Selection

The number of sensor data features is vast, which puts a big challenge in designing a high-performance model with relevant features only. Some of the features have a strong relationship with one another. With redundant features, the dimensions of variables increases as well as the time to train and test the model. Feature selection methods help us in our mission to create a more predictive model by identifying and removing redundant, unnecessary, and irrelevant features from the targeted model. It reduces the complexity of the model by reducing the desired attributes for the model. It also provides a better understanding of the underlining process that generated the data (Guyon & Elisseeff 2003).

Three different types of feature selection processes are popular in designing a high-performance machine learning model- filtering, wrapper, and embedded. We employed the Correlation-based Feature Selection (CFS) (Hall 1999) filtering approach. The CFS method filtered out redundant and unwanted features and kept only the features that will be more related to produce better predictions. The implementation of the CFS algorithm may use forward, backward, or best fit heuristics search. We use the best-fit algorithm to search ahead for features through the search spaces, adding the stopping criteria to five, as described here to stop the search to the entire search space. The search space will incrementally add features and stop if the five consecutive steps do not show any improvements. The search space started with several features that researchers have previously used and are strongly correlated with speech dynamics. We selected a total of sixteen features for classification.

4.4.3 Train, Test and Classify

In our experiment, we have found that one-fourth of the time, a human makes conversation, and three fourth of the time, they remain silent. The number of windows that represents speaking segments is much less than the number of segments that represents silence. We used a ski-learn standard imputer, which randomly computes sections from existing segments and adds to the existing dataset. We applied 80-20 rules and divided them into three sets. The first 80% of our data was used to train the model, 10% was used to test, and the rest of the 10% was used to validate our trained model. We applied the SVM binary classification to train speech and non-speech and evaluated the results. We have achieved 98.8% accuracy classifying the segments (see ROC curve in Figure 4.8).

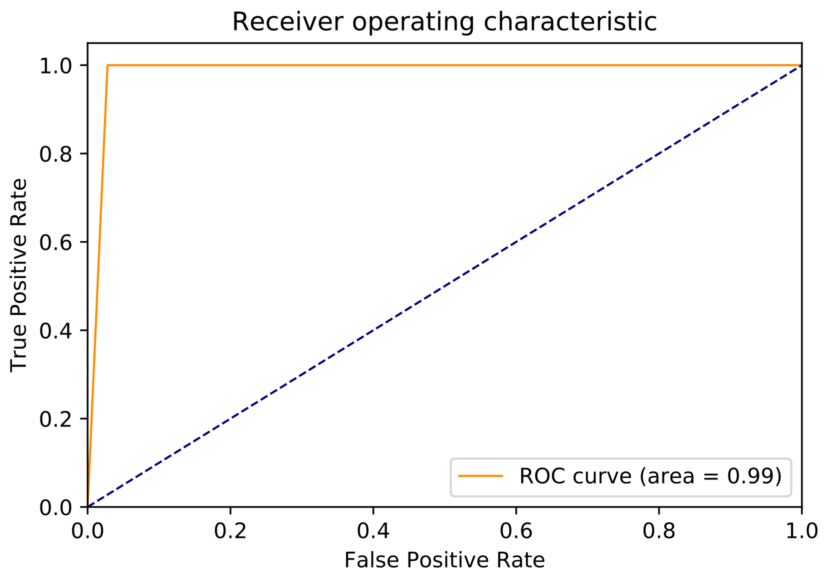


FIG. 4.8: ROC Curve shows the accuracy of our trained model.

4.5 Future Work

Our proposed speech dynamics analyzing method addresses only two different classes: speaking vs. silent. However, some other important events take in the workplace, i.e., laughing, shouting, and various other emotional expressions. In addition to classifying into two speech segments, we can design a comprehensive model to detect any parson's speech dynamics. The study will enable us to analyze whether a person is more interactive than others or not, how their daily speaking pattern affects the workplace? It will also help to improve the quality of conversation in the workplace. The future study may analyze more of these matrices in addition to detecting the conversation.

4.6 Conclusion

This study addresses a unique scenario in conversation analysis in the form of group conversation in the workplace. The conversation analysis started long ago. However, most of the cases, the study is being done on single person conversation or two-way conversation in a doctor's or lawyer's office, or at a random place. In addition to that, most of the researches is based on recording audio that has some privacy concerns. In this study, we have focused the investigation on analyzing group conversations using respiratory strain gauge sensors. The state of the art conversation study using a wearable sensor considered only a single conversation, and also the recording was done in isolation rather than natural workplace setup. We have collected quality data by setting up a real workplace. We clean our collected data using both noise removal techniques as well as the frequency filter to make the data more useful in detecting conversation among participants. We followed the state of the art machine learning process to design, train, and test model and have achieved high accuracy in identifying a group conversation as speech or silence. This work might help to analyze the workplace environment and how to improve its condition.

Chapter 5

HUMAN INTELLIGENCE IN FEATURE PHASE

In previous chapters, we have discussed how human intelligence helps machines in the data acquisition and decision making phase. The decision model of an application might depend on multiple features. Some of those features are extractable by machine algorithms, and some are more complicated and open-ended that require human intervention. The process, algorithms, and integration are different for different types of applications. In this chapter, we discuss the bus stop resource localization for the visually impaired. We explain here why human intelligence is required to extract features instead of machines. We also discuss the detail design of a collaborative system integrating human knowledge in the feature extraction phase.

5.1 Problem Statement

Over 3 million individuals have a visual impairment (WHO 2010) globally. The ability of the visually-impaired to use public transport is a key component of independent living. Unfortunately, most existing infrastructure incorporates only visually distinct features, i.e., signs and tags for identification, which are inaccessible to visually-impaired users. The commonly used technique for blind riders to identify and to localize a bus stop is GPS-based geographic information systems (GIS). It can guide users to the vicinity of a bus stop with a median horizontal error of 58 meters (Zandbergen & Barbeau 2011). However, such precision is not suitable for blind users. They are still required to manually explore the vicinity of a bus stop location with a cane (Hara *et al.* 2015). Moreover, in many cases, points of interest (POI) of bus stops in navigation systems are usually not accurate. They deviate from their actual positions slightly, which introduces additional error to positioning. These two problems are illustrated in Figure 5.1(a). On the other hand, though RF-based (Lymberopoulos *et al.* 2015) systems provide better accuracies, they require infrastructure modification and maintenance. These also suffer from multipath effect or Electromagnetic Interference (EMI) due to the highly dynamic real-world environment. With improved cameras and processing power on mobile devices, a vision-based approach is a plausible remedy for the "last-mile" problem.

High-level object detection and localization using computer vision are still underdeveloped. Moreover, the accuracy and speed of such algorithms on embedded platforms are still impractical for near real-time applications (Manduchi & Coughlan 2012; Terven, Salas, & Raducanu 2014). Our proposed system takes a more straightforward and reliable approach instead of general object detection by identifying an object found to be universally present in a bus stop– the bus stop sign. Fortunately, the sign itself is designed to be visually conspicuous, and efforts are made to place it in a highly visible location. Although the intention of visual design and placement was for sighted users, we can avail of these attributes to aid people with visual impairments as opposed to requiring additional infrastructure modifications (Dogs, Catapult, & Microsoft). Our vision-based system, `LastStep`, can accurately localize a bus stop and can help blind riders to navigate to other resources (e.g., benches) at the stop as well as to a specific position (e.g., where

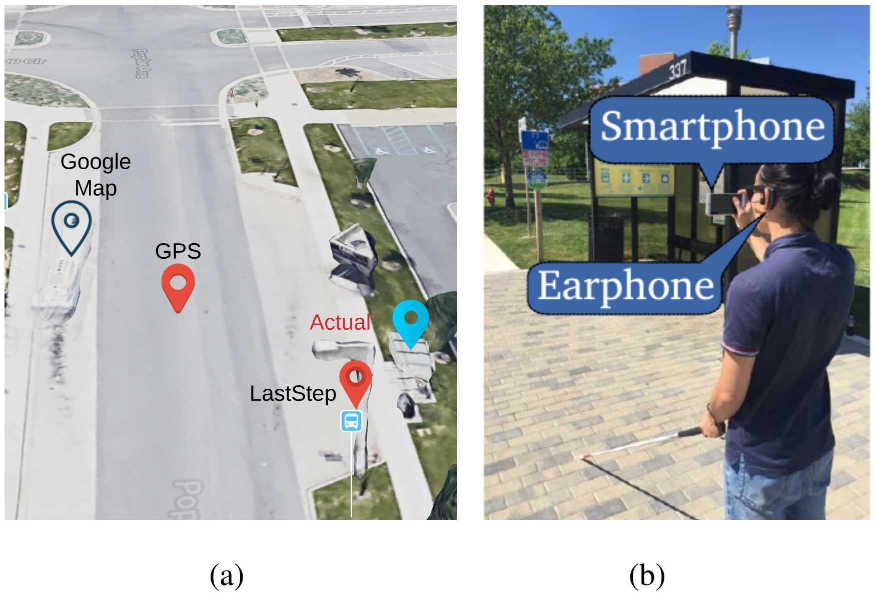


FIG. 5.1: Motivating scenario. The map (a) obtained from Google Earth shows the lay of land in the area of the bus stop (b). The user's location obtained from GPS is deviated from the actual location along with a large uncertainty as indicated with the big blue circle. Additionally, the pin of the bus stop is deviated from the actual location of the bus stop. A prototype of `LastStep` implemented on an off-the-shelf smartphone is shown in (b).

to stand or wait for the bus). A vital characteristic of this approach is the use of computer vision to detect and localize the visual sign at a bus stop. `LastStep` uses the sign as a reference point by which to precisely find other objects and key positions in the stop. The prebuilt fine-grained map guides the `LastStep` to determine the location of a specific resource. Figure 5.1(b) illustrates the front-end hardware of `LastStep`. It consists of only an off-the-shelf smartphone device and a wireless earphone for verbal feedback. `LastStep` uses the smartphone camera to detect the sign with SIFT (Lowe 2004) features and estimates the distance to the sign with homography transformation and scaling (Li *et al.* 2016). A bus stop sign has standard dimensions for a specific state of the united states. The dimensions can be collected with minimal effort. After detecting the sign and estimating the relative position to the sign, the position of a user is mapped onto a fine-grained bus stop map. A fine-grained map constitutes of spatial location and name of all resources at that bus stop. The map is constructed with a custom scalable approach with the help of Google Earth and Google Street View and human annotations.

Research Contributions

The design, implementation, and evaluation of `LastStep` presents the following research contributions.

- **An intelligent vision-based localization system that provides positioning information of bus stops and facilities in fine granularity to improve accessibility for blind riders.** We present a system, `LastStep`, that uses computer vision algorithm to detect and localize a bus stop with step-level accuracy for people with visual impairment. It also improves the accessibility of the whole facility by providing fine-grained positioning information of other facilities at the stop for users. Specifically, it provides directions on where to wait, where to sit, where to take shelter from rain and sun, or the location of other amenities. `LastStep` describes a method for utilizing existing familiar infrastructure using computer vision with minimal overhead and setup. It should be considered complementary to RF anchors and could be integrated with such systems for outdoor and indoor navigation to achieve increased navigational performance and robustness.
- **A lightweight and scalable methodology to extract features from non-expert human crowd** We designed human tasks of creating spatial 2-D map of bus stops-dividing the complex tasks into optimized pieces and aggregating multiple responses into one. We generated fine-grained labeling maps of bus stations using human an-

notations with a combination of Google Earth and Google Street View data. The generated map was able to localize amenities at a bus stop with high accuracy. This approach can also be generalized to obtain fine-grained labeled maps of other outdoor facilities.

- **System implementation and evaluation on an off-the-shelf smartphone.** We implement a functional end-to-end prototype of `LastStep`. The front-end is implemented on a smartphone without any auxiliary hardware modules such as stereo cameras or laser scanners. We evaluate our system using blindfolded participants and demonstrate that participants can localize bus stops with a higher success rate and with a lower latency compared to manual searching with a cane.

5.2 System Overview

Figure 5.2 illustrates two different components required to localize objects by `LastStep`- a) apriori contextual database and b) vision-based sign detection. The reference image of the sign is pre-stored along with the physical dimensions of the sign in the contextual database. The physical dimensions of a bus stop sign are required to estimate the actual distance of that sign from the camera.

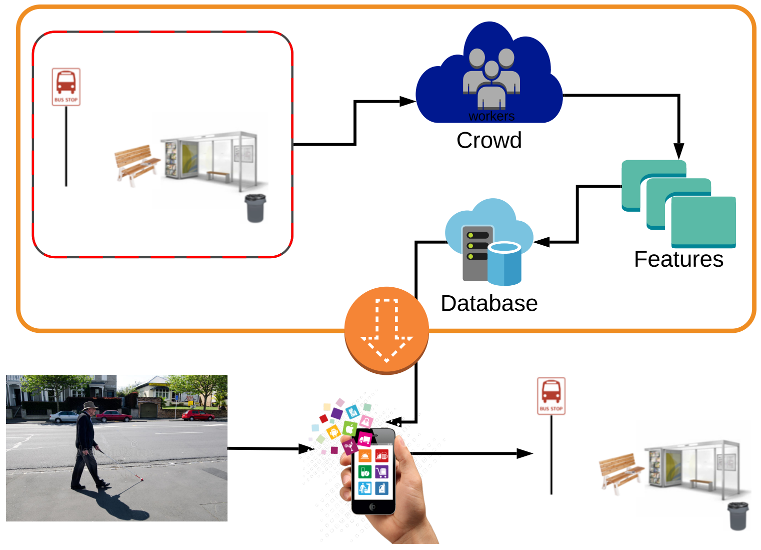


FIG. 5.2: The apriori database (top). The bus stop image is posted to the human crowd. They identify the amenities in the image, mark their location, and provide the front-facing direction of the bus stop. Some images of the bus stop sign and their dimensions are uploaded to the database as well. The vision component (bottom) downloads the information of the bus stop from apriori database, detects bus stop sign and user location in the two-dimensional map, calculates distance and direction from the user to any amenities at that bus stop.

The contextual database also contains a two-dimensional map of each bus stop. The two-dimensional map composes of a bus stop sign, other resources at that bus stop, and the distance from the bus stop sign to each resource. A user visits a bus from one of two directions along the sidewalk. The two-dimensional map of resources is different from two different sides. Two-dimensional maps from the two different sides are stored in the contextual database. The vision component detects bus stop signs using the reference image stored in the apriori database. `LastStep` turns on the camera to identify a bus stop sign by matching SIFT keypoints. It determines the in-situ image as a bus stop sign if it finds a sufficient number of matched key points between the in-situ image and the reference image. A group of pairs of matched keypoints is used to calculate the 3D homography matrix from the sign in the reference image to the detected sign in the in-situ image. With the homography matrix and an estimated gravity vector from an inertial measurement unit (IMU), the system can position the user onto the fine-grained map. Finally, the mapping information is fed back to visually-impaired users as an audio description of the facilities at the bus stop, along with the respective directions and distances from the user. It should be noted that if the only distance from reference points are known, distance from three such reference points is required to perform localization. In our system, only one reference object is required to calculate not only the distance but also the three-dimensional direction of the object.

5.3 Apriori Database

The SIFT based scene detection algorithm requires a reference image. Collecting a reference image is easy as every state or country has a standard designed bus stop sign. For example, the state of Maryland has the same bus stop visual signs all over the state. We need to collect a few of the images of a particular bus stop sign, calculate SIFT features, and upload it to the apriori database. Another one-time information is the dimension of the visual signs and average height from the ground to the sign.

The contextual database comprises of a) key points, b) the physical dimensions of a bus stop sign, and c) a fine-grain map of a bus stop. The key points of a visual sign are pre-calculated using a reference image of a sign. Our system obtains the physical dimension of the sign together with a reference image during system setup. Public bus stop signs are standardized for a state, and it uses the same pattern and physical measurement. We used a single reference image and physical measurement of a single sign for the state of Maryland.

A minimal effort is required to set up the system in a new area. Moreover, in some cities, the image of a bus stop sign and it's physical dimensions can be obtained directly from government websites (Authority).

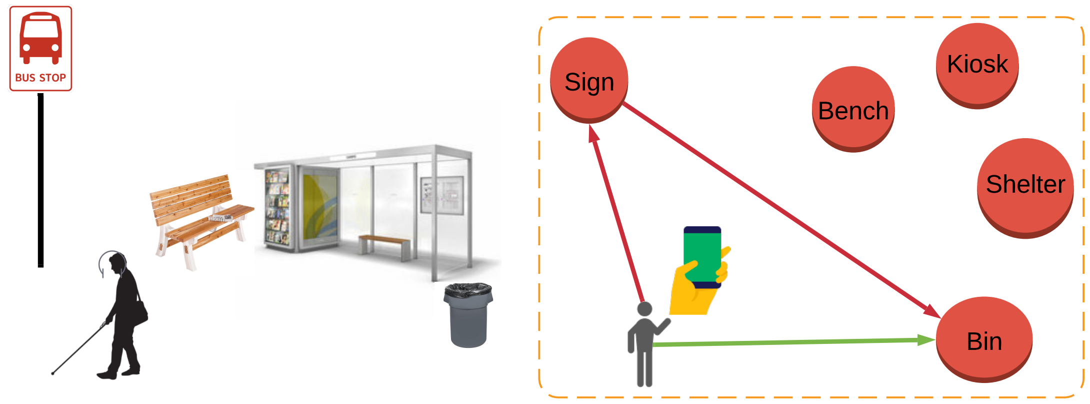


FIG. 5.3: A visual impaired user can locate any amenities using mobile or embedded application if the application can detect a bus stop sign using computer vision and know the two-dimensional map of the bus stop. The app can find the precise user location in the two-dimensional map and guide the user to his/her desired amenities.

The SIFT algorithm is very fast and useful to detect simple objects in real-time, even in the low powered devices, i.e., embedded camera. The vision algorithm can detect signs and can also determine the distance of the user from the visual sign using the height of the hanging point and dimension of the sign. Once we know the precise user location, the remaining challenge is to detect the accurate location of other amenities at that bus stop. Some bus stops have a resource that may not be present at another bus stop. The resources themselves are different from one another visually; for example, there are multiple types of trash cans and benches. Detecting an object in real-time in an embedded system is very difficult even if we use only the trained model. The varied dimensions of an object and occlusion of one object by another make it very hard to calculate distance from the user to a particular object. However, it would be easy if we know the location of any object from an anchor point. In this case, the bus stop sign is detectable in real-time. If we have a two-dimensional map (see figure 5.3) of other objects from this anchor point, we can locate the objects. However, calculating the map using an image is difficult because of their dissimilarity and the camera position, which usually sees the side view of an object. Instead of an image, we can use Google maps to generate a 2D map of resources at a bus stop.

Google Maps is very robust and has several types of information about a specific location. We can see the top-down view of a bus stop in a Google map satellite view and can drop a marker at any place. However, the image is not that clear. On the other hand, Google earth is much clearer but does not provide the location information. So, combining these two services, we can observe a bus stop and mark all the resources efficiently. For getting a bus stop's two-dimensional map, these two pieces of information are enough to get the GPS location of objects. However, designing a human intelligent task (HIT), and getting the desired response is tricky. We describe the process of generating a fine-grained map of a bus stop in the following section.

5.3.1 Web Interface for Human Annotation

We designed a web-based tool (see Figure 5.4) to extract the required features for a bus stop. The annotation tool consists of two components- (a) an interactive Google Map tile, and (b) a simple drawing tool. The interactive Google map tile displays the bus stop in a top-down view with a specific zoom configuration, but it is also possible to zoom in or out to see the resources. This interactive tool also has the option to toggle to Google earth view, which helps a crowd worker see the resources in a 3D 360^0 view image. The facility helps the worker identify the correct object. For each bus stop, we need answers for three of our questions that are required by `LastStep`,

- Does the bus stop have amenity X? X could be one of the amenities commonly available at bus stops. X could be a trash can, a bench, a rest area, or a visual sign, etc.
- If amenity X is present, is it possible to mark the object in the top-down view? and
- What is the direction of the bus stop front-facing side?

If we aggregate the responses of these three questions, we will be able to create a two-dimensional map of that bus stop. The standard procedure of crowdsourcing is to create simple tasks so that humans can process quickly and are motivated enough. However, creating more tasks will increase monetary and computational cost. We designed two Human Intelligent Tasks (HIT) to get this information from a human crowd.

HIT 1 - Object's Availability and Its Location: We created a task (see Figure 5.4) to know whether a specific resource is present at the bus stop. And, if the resource is present, what is its location? More than one resource can be present at a bus stop. We created a simple task to get information about each object. The task covers two questions a) *Do you see object X here?* and b) *If you see the object, drop a marker on the object X.* The task guarantees whether object X is present or not. The drawing tool contains a button to drop a marker on the object. If the Google Map view is not clear enough to decide, the crowd worker can select Google earth mode to see the bus stop clearly and mark the correct resource in Google Maps.

HIT 2 - Orientation of the Visual Sign: We designed a HIT for the human crowd (figure 5.5). The task contains a frame where the bus stop location can be viewed in the Google maps. In addition to the frame, a drawing tool is also attached. The drawing tools contains a line drawing option. A person is asked to draw a line from a point towards

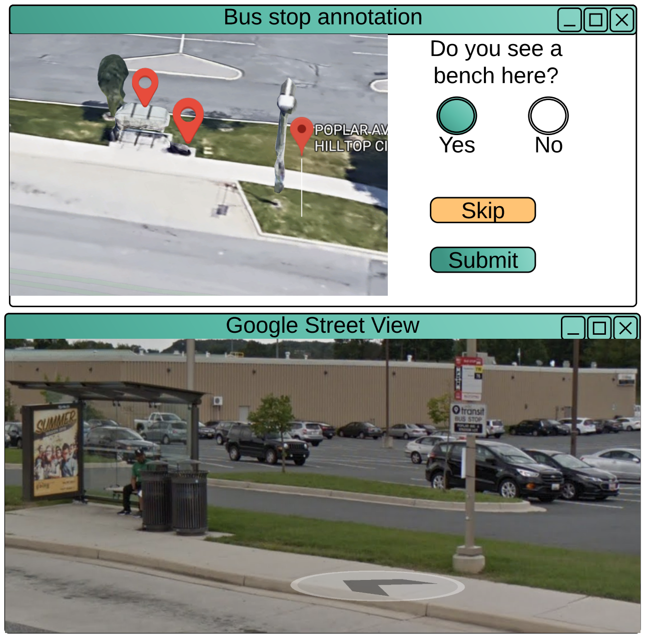


FIG. 5.4: Crowdsourcing task: a) (Top) Question to get an object's availability and location in Google map, b) (Bottom) Google earth view of the bus stop, which helps to see what is at that bus stop.

the bus stop sign. The direction is always based on the bus stop sign. We asked the worker to see the bus stop sign on Google earth and determine which side the sign is facing. We recorded the clicking events from the crowd worker to determine the direction of the sign. If the sign is not visible clearly in Google map view, the interactive tool helps to see the clear image in Google Earth mode.

5.3.2 Data Aggregation

We received responses from multiple humans for the same task as a single response is not reliable. The first task that we designed has two parts. First, the system aggregates the response based on majority voting rules from n number of responses. Sec-

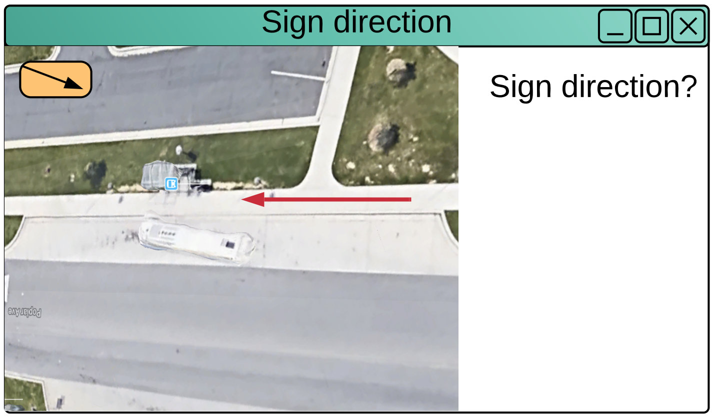


FIG. 5.5: The crowdsourcing task to get visual sign direction.

ond, if the answer to this response is positive, the spatial position of a resource is aggregated using the geometric mean of the x and y position. In our designed tool, human workers switch between street view and map view to recognize a resource and it's spatial location on the map. Thus, the labels will be dropped under different zoomed scales. Our system records the absolute GPS coordinates when workers drop the pins to avoid the zooming effects on location. The system uses geometric mean instead of arithmetic mean of GPS coordinates to minimize the interference of outliers. In the case of the second HIT, we use the majority voting rule to get the sign direction.

5.3.3 Building Fine-grained Map

A fine-grained map of a bus stop is generated based on responses provided by humans. The system obtains an image of the Google Map tile on a fixed zoom scale and converts all GPS coordinates into pixels unit. We calibrated the scaling factor (distance per pixel), as shown in Figure 5.6. It also calculates the distance between two specific points with GPS coordinate and their corresponding pixel distance of that same two points in the map image. We calculate the distance between two specific points with GPS coordinates and also find the corresponding pixel distance of that same two points in the map image. This extracted information is then consolidated to populate the contextual database in a format of a 2D map of the bus stop that is used to provide positional information of other objects to improve the accessibility of the entire bus stop.

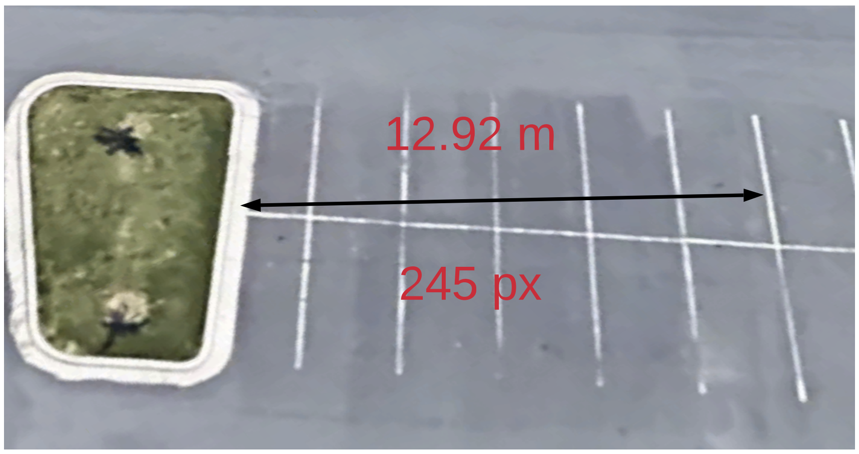


FIG. 5.6: Calculate distance from image: Distance between two points in Google map can be calculated from the GPS location. The number of pixels between those same two points are calculated to get the distance per pixel at a specific zoom level in Google map.

5.4 Object Localization for a Bus stop

The apparent object in a bus stop is the visual sign of that bus stop, which has a particular set of standard features. The visual sign also has a standard height and width. Other amenities at a bus stop surrounds the visual sign in a 2D map. Our proposed method can easily detect bus stop sign using vision algorithm (Li *et al.* 2016). Once the visual sign is detected, using the position of the mounted camera at the user body or stick and standard height and width of the visual sign, the related distance of the user from the visual sign can be detected. To localize the user in the bus stop and compute useful directions to other facilities such as shelter, bench, and trash cans at the bus stop, we need the user's precise location in a fine-grain map. The fine-grained map is generated using close-up views provided through Google Earth data. The objects in the bus stop are tagged with human annotation, as described in section 5.3.

Typical positioning techniques use distances to multiple reference points to localize an object. Unfortunately, in most of the cases, all of the signs are mounted on a single pole representing a single point in a 2D map. So, the use case requires a richer inference of position from a view of a single sign. In our system, we use a vision-based approach to position the user's location by utilizing the orientation of the sign in the map (Li *et al.* 2016).

5.5 Feeding back the Information to Users

Feedback is an essential part of `LastStep` as the users are visually impaired. The positional information in our system could possibly be reported based on a few different coordinates- absolute coordinate (map coordinates) or relative coordinate (camera coordinates). In the absolute coordinate system, more information (the layout in the map) must be fed back to the user. The process requires a higher output bandwidth when choosing the feedback medium. Moreover, users are required to scan through the whole map to understand the layout of resources in the real world. It introduces higher latency in the runtime interpretation. A more direct feedback strategy is to report the object distance and direction in an ego-coordinate system (relative to the camera).

The detailed feedback flow is shown in Figure 5.7. A user points the smartphone in one direction and taps the screen to start the process. The system will first indicate to the

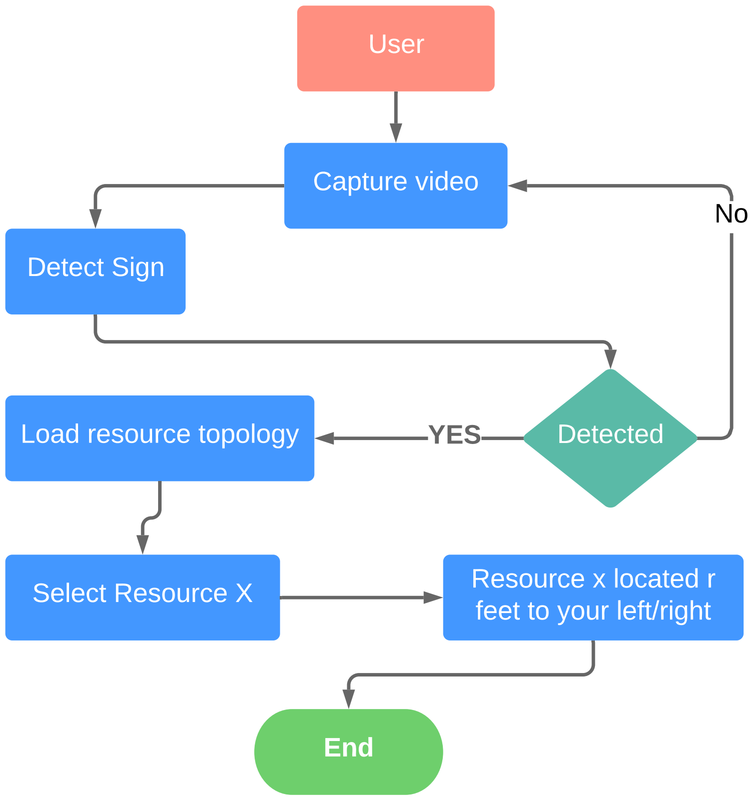


FIG. 5.7: The operation and feedback flow of `LastStep` in a smartphone. It is triggered by the user facing a direction and tapping the screen to start capturing image and finally reporting the position of targets when detected successfully.

user, the start of the process, then capture an image and detect the bus stop sign in it. The yes/no detection result will be a verbal output to the user. If not, the user can change the direction and trigger the detection again. If yes, the system will start to localize the user in the ground map and output a list of objects in the vicinity of the bus stop. Based on the user's selection, the system will report the distance and direction of the object based on the camera coordinate, which is verbally as "left/right xx meters, front zz meters".

5.6 Evaluation

The LastStep is a human-machine collaboration where a computer vision algorithm uses the human-provided resources layout map of a bus stop. The availability of a resource, its relative distance from the bus stop sign, and its actual angle from a visual sign, the orientation of the spatial map from the user's walking direction plays a vital part in providing correct information to a visually impaired user. We evaluate the human-provided responses together with the whole LastStep system evaluation that uses human-provided data.

5.6.1 Evaluation with Human Participants

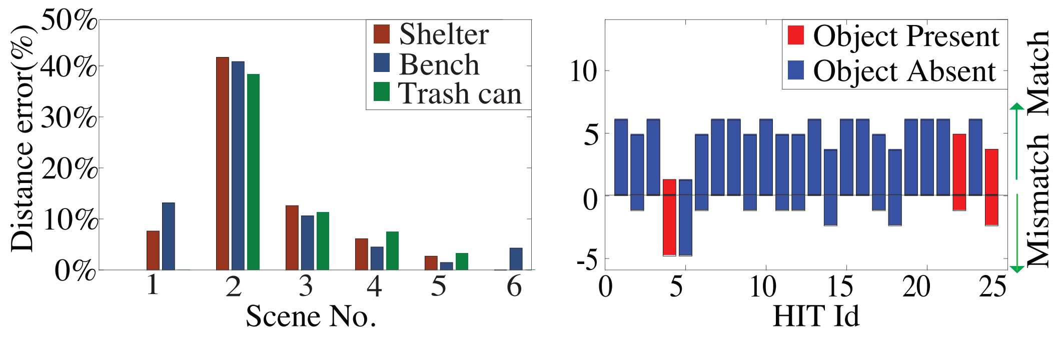
Evaluating LastStep on ordinary people instead of visually impaired poses several difficulties. The significant challenge is to mimic the scenario of no visual with visually healthy people. We evaluated bus stops around our campus by graduate students who might be familiar with those bus stops. They might have the memory of the layout of those bus stop and the resources around it that might profoundly accelerate their search speeds even without the assistance of our system. Besides, we need multiple trials at the same location to compare results with or without LastStep. The participants might have an impression of the layout of a bus stop after the first trial that might bias the results at the second trial. To reduce these biases, we randomize the trials in our experiments. Specifically, we collected 36 trials in the user study using 3 participants on six different sites and two trials (with or without LastStep) on each site. The order of performing the trials was randomized. It means that the participants did not know which site. On a site, a person was only asked to perform a single trial either with or without our system. Then they were driven to the next site to perform another single trial and come back to the previous site later in random order. Also, in between two trials, the participant was taken back to the car and randomly driven around. During the whole process, the participants were kept blindfolded using an eye mask.

Table 5.1: Amount of time (in minutes) used for searching the bus stops with or without the aid of `LastStep`.

Parti-cipants		Sites					
		I	II	III	IV	V	VI
A	w/	2.3	4.1	1.2	2.1	1.7	3.5
	w/o	×	×	8.7	10.5	×	×
B	w/	1.6	3.5	1.3	1.9	2.3	3.1
	w/o	×	×	12.3	×	11.6	×
C	w/	×	2.5	3.8	2.7	4.5	2.3
	w/o	×	×	×	×	×	5.4

** The entries with "×" mean the participants failed in the searching after a threshold of 10 minutes.*

The experiment results are shown in table 5.1. In each trial, the participant was asked to try his best to search the bus stop at his surroundings. Only after a minimal searching time of 10 minutes, the participant can commit to failure in the trial that is shown as "×" in TABLE 5.1. Overall, without the aid of `LastStep`, participants only succeed in finding the bus stops $5/18$ times. With `LastStep`, the success rate was $17/18$. In the case of failure (subject C on site I with the aid of our system), the detection of the sign failed when the subject initially pointed to the correct direction that includes the sign in the camera view. When the subject re-oriented himself and tried to find directions, he never turned in the correct direction. In the post-interview, he stated that he thought he was turning around and trying all directions multiple times. It hints that self-orientation training will help the users to utilize `LastStep` properly. Such an orientation training is common for people with visual impairment who would like to travel independently (AFB). Moreover, in the trials where the participants found the bus stops successfully with the manual search without `LastStep`, the search time was 4-times longer on an average.



(a) (b)

FIG. 5.8: (a) Comparison between experts' and Turkers' responses. Matched responses are plotted in positive and mismatched in the negative direction. Blue color indicate the objects that are actually present, while those with red color indicate the objects that are actually absent. With 5 responses from AMT, objects in 22/24 HITs can be identified correctly. (b) The error of distance from different objects to the anchor point in the 2D fine-grained maps. Distances from the anchor point are calculated using experts' and Turkers' annotations respectively. The percentage error is the ratio between these two distances, showing how much the objects' positions from turker's responses deviate from that of experts.

5.6.2 Accuracy of Human Annotations

We evaluated human annotations that helped to generate the fine-grained map. The map comprises three different types of information a) direction of a user moving towards the bus stop, b) labels of the objects at the bus stop, and c) spatial position of resources from the visual sign. A user can walk towards a bus stop from two different sides of it along the sidewalk. The first one determines the direction of the users walking towards the bus stop. If the system can determine that correctly, it will be able to find the relational position of other objects from the sign position. The second one helps to identify which objects are at the bus stop, and the third one provides the objects' spatial position. In our experiment, we collected responses from two different sets of human workers. The first set of workers are experts who have experience in annotating objects correctly. We work here as the experts and annotated objects by ourselves. These responses were used as ground truth or gold-standard data for comparison. The second set of workers are paid human workers from Amazon Mechanical Turk (AMT). We collected responses from AMT for six different bus stops around our campus and two of our neighborhood apartment complex. For each bus stop, we found four different types of objects (sign, shelter, bench, and trash can). We published twenty-four Human Intelligent Tasks (HIT) for human annotations and collected around two hundred and fifty responses from available online workers. We found (see in Figure 5.8) that at least five answers are necessary to get correct annotation.

First, we evaluated the angular error of the user walking direction annotations comparing Turkers' and experts' responses. For each bus stop, we aggregated the responses of direction annotation using geometric means to minimize the interference of outliers. Across all six sites, the aggregation of Turkers' data showed an average angular error of 13.8^o compared to experts' annotation. Second, we evaluate the accuracy of object detection. The online tools designed the first tasks that ask a binary question about the presence of an object. In further evaluation, this information also works as an initial filter for outliers. We compared the result of this question to the experts. Using majority voting aggregation method, `LastStep` can identify the existence of the objects with 97% accuracy as shown in figure 5.8(a). Third, we evaluated the error of the spatial position of an object in Turkers' responses. In `LastStep`, the position information of an object is inferred using the distance of that object from the anchor point (sign). Hence, we used this distance to evaluate the location accuracy of the turker's annotation. We calculated all objects' distance from the anchor point for all bus stops. We computed the distance of a resource calculated from the crowd with experts' responses. The average error for distance calculation is 11% as shown in Figure 5.8(b). The difference in Scene 2 is about 40%. It is because there are two bus stop signs mounted on different poles (one from the government authority and another from the university transit department) at this bus stop, which might have confused the Turkers.

5.6.3 `LastStep` Evaluation

The performance of `LastStep` depends on the quality of human-provided data. However, the `LastStep` performs both sign detection and distance estimation before it uses human provided data. The SIFT method can successfully detect visual signs in the range of 3 to 9 meters with 30 fps video recording at resolution 1280×720). The distance estimate algorithm (Li *et al.* 2016) can estimate relative distance in a centimeter range, significantly better compared to GPS, whose error rate is in few meters. Localizing other object's accuracy depends on the accuracy of the precise user location in the fine-grain map. We evaluated the position accuracy, and it is less than 0.5 meters.

5.7 Conclusion

In this chapter, we present `LastStep`, a vision-based system for navigating to resources in a bus stop for visually impaired riders. `LastStep` can localize anchor points such as bus stop signs with an error less than 0.5 meters when the user is at a distance of 3 to 9 meters from the sign. `LastStep` also introduces a scalable approach to generating fine-grained maps of bus stops using human annotations on data gathered from Google Street View with the help of Google Earth. We evaluated our proposed system `LastStep` and showed that users using `LastStep` could perform localization with a higher success rate and four times faster than manual searching. Since `LastStep` utilizes existing infrastructures and the system setup time is minimal, it can be considered as a complementary approach in cases where RF anchors are unavailable or infeasible to deploy. It could also be used for indoor and outdoor navigation, potentially in conjunction with RF anchors for increased system reliability and accuracy.

Chapter 6

HUMAN INTELLIGENCE FOR TEMPORARY SOLUTION

In the previous chapter, we have discussed our proposed human-machine collaborative system to localize resources at a bus stop. There is a standard set of resources, i.e., bench, shelter, trashcan, etc., that could be available at a bus stop. However, there is no guarantee that the whole set of resources is present at all bus stops. In that system, humans help to build the spatial topology of all resources at a bus stop. Each bus stop has its own set of resources, and each bus stop is different from another bus stop. Once a bus stop spatial map is generated, no need to create the same spatial map again. Hence, collecting data over time will not help in the long run to reduce the necessity of human knowledge. However, If an application uses the dataset where each instance has a standard set of properties or features, then a new instance can be added to the existing dataset as the new input will add another instance to the current dataset.

While designing the human-machine collaborative system, the goal is to minimize human contribution as much as possible to make the system more automated. If the number of instances is insignificant and data acquisition is very hard or time-consuming, a machine learning model is not a good idea. Instead, we can integrate human intelligence to leverage the data. The human-machine collaborative system will add new instances every time it servers a new query and will enrich the database. The enriched dataset can help to reduce or even eradicate the dependency of humans in the long run and can make the system fully automated. In this chapter, we walk through an application called `PreSight` where a new query will add a new instance to the database. We showed here how humans could help machines providing features extracted from the data instances. Each instance has a common set of features that can help to build a machine learning model in the future when we would get enough data to train a machine learning model. Each new query will enrich the dataset, and the enriched dataset will help to reduce human intervention in the long run.

6.1 Problem Statement

A key element to improving the standard of living for people who have a visual impairment is independence (WHO 2010). Navigation on sidewalks and other walkways comprise a major ingredient of independent living. Unfortunately, sidewalks today are ridden with accessibility problems such as obstacles in path and unexpected flights of stairs. Existing systems such as smart canes (Shoval, Ulrich, & Borenstein 2003; Borenstein 2001) or smart wheelchairs (Ivanchenko *et al.* 2008) suffer from the limitations of sensors such as ultrasound or infra-red used in these systems (Hersh & Johnson 2008). These sensors are not suitable for detecting multiple obstacles, or for applications that require object recognition, such as landmark detection for navigation.

A camera-based system is a plausible solution to the problem. Embedded cameras worn by users or built into canes combined with computer vision algorithms can be used for object detection and recognition. Unfortunately, current discriminative object de-

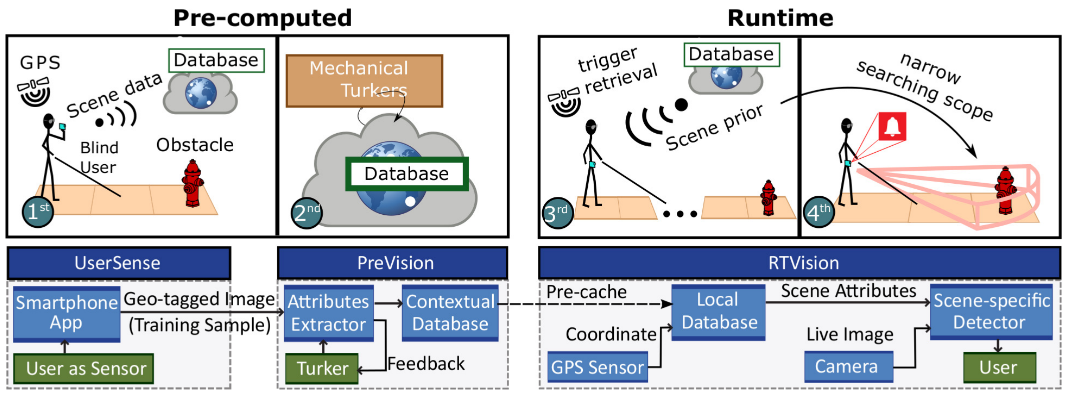


FIG. 6.1: A cartoon figure illustrating the operation of `PreSight`. In the pre-computing process, the blind user-A reports the scene with the accessibility problem to our system by taking an image of the scene in the first step. Then our system publishes a HIT of the scene image and then the Turkers extract the parameters for the scene-specific detector in the second step. In the real time subsystem, the smartphone-based system worn in another user-B retrieves the parameters from a pre-cached local database when approaching the scene, as shown in the third step. Based on the scene attributes, the system focuses on sparse areas in the view to detect the object in real time, as shown in the fourth step.

tectors (Dalal & Triggs 2005; Felzenszwalb *et al.* 2010) require extensive training in a high-dimensional feature space. During sensing and classification, the high-dimensional feature extraction and matching can be prohibitively expensive to compute in real time on embedded platforms.

To address this problem, we derive insight from the following human vision attribute: a person at a given scene can quickly select candidate regions of an object-based on approximate descriptions of location, color, and size. These coarse-grained selections are considered top-down pre-attentive computation with prior knowledge (Itti & Koch 2001), that can guide a human's attention (Moran & Desimone 1985; Wolfe & Horowitz 2004) into narrow regions for further fine-grained processing. This suggests that a faster searching scheme on limited hardware can be performed for a given object in a given scene if the object guidance attributes are known on a per-scene basis.

Inspired by this observation, we propose a system `PreSight` that accelerates general object searching using per-case *a priori* characteristics. `PreSight` detects obstacles on sidewalks accurately in real-time with limited computational resources. The system guides its "attention" into narrow areas by incorporating prior knowledge of individual objects. For example, if we know *a priori* the color of the object, the regions in the view that have similar color can be pre-selected for further processing. Note that while the detection process is accelerated with per-case priors, additional complexity is introduced in the off-line data collection and extraction. Fortunately, the growth of crowdsourcing and smartphones provide a low-cost scalable solution for this task.

The overall operation of `PreSight` is illustrated in Fig. 6.1. In `PreSight` we build *a priori* scene specific information using data provided by the system users. When a user pass by an obstacle and do not get avoidance notification, s/he can provide information (image) of obstacle using preSight mobile application. These geo-tagged images are then annotated using Amazon Mechanical Turk (AMT). Scene specific object data such as the type, color, and physical size of the obstacle is collected using Mechanical Turk. The scene-specific attributes lazily populate a backend database. In the real-time detection system, an embedded camera retrieves data of the scene from a pre-cached local database. It is noteworthy that the data cached per scene is as minimal as 415 Bytes (only specific extracted features will be cached) and it can also be pre-cached when there is Wi-Fi available. The embedded system (a generic smartphone) utilizes the prior data to accelerate the detection process.

Research Contributions

The design, implementation, and evaluation of `PreSight` presents two novel research contributions.

A heuristic strategy to accelerate object detection by utilizing case-specific *a priori* characteristics to pre-filter live images: For individual scenes, this heuristic strategy enables attentive machine vision for which machines are able to focus the computation power on narrowed areas in the view with the help of *a priori* characteristics of the target. In another word, we offload the on-line machine computation by performing off-line prior extraction using human computation. This enables real-time computer vision object detection in embedded platform.

An end-to-end implementation and evaluation of the framework with application to accessibility problem detection: We demonstrate a functional prototype of `PreSight` that can accelerate general object-detection algorithms and achieve real-time obstacles detection in a commercially off-the-shelf smartphone. Our prototype `PreSight` system includes an interactive crowdsourcing web interface for prior data extraction, and a fully functional smartphone-based vision system. The evaluation shows that `PreSight` gains an 8x speedup on two benchmark object detection algorithms HOG (Dalal & Triggs 2005) and DPM (Felzenszwalb *et al.* 2010)) without degrading detection accuracy.

6.2 System Overview

`PreSight` comprises of three key components illustrated in Figure. 6.1- (1) UserSense; (2) PreVision; and (3) RTVision (Real-time Vision). Our system senses scenes with accessibility issues and extracts scene-specific prior in UserSense and PreVision, and performs real-time obstacle detection using RTVision. In UserSense, blind users capture images of accessibility problems on sidewalks using the mobile application when they find a problem on a sidewalk. The application transfers the geotagged image to a backend server. The image is then encoded as a HIT (Human Intelligent Task) and disseminated to AMT in PreVision. Turkers annotate accessibility problem in the image, classify them, and provide data on the color and physical dimension of the accessibility problem. The collected data is then aggregated to build a geotagged database of scene-specific information. When a new user visits the scene, the front-end smartphone-based application starts detecting the object. The application utilizes the prior data in the contextual database to accelerate general object detection algorithms to detect accessibility problems in real-time. This front-end, called RTVision, triggers the preprocessing and object detector only when the user is within a radius r of an accessibility problem registered in the PreVision database. Next, we discuss the three key components of our system in detail.

6.3 PreVision: Extracting Scene-specific Features

In PreVision, the geotagged images collected by users are used to generate HITs (Human Intelligence Tasks) automatically. The HITs are then disseminated to Amazon Mechanical Turkers to extract scene- and object-specific features. While data collected using Turkers can be subjective and biased, it is possible to design HITs to control quality, and de-bias by aggregating data from multiple Turkers. In this section, we first describe the design of HITs. We then describe how we aggregate data from multiple Turkers.

6.3.1 HIT Design

Our system (described in Section 6.4) uses two characteristics that humans use in their pre-attentive vision (Wolfe & Horowitz 2004) (color and size) to accelerate object detection. Specifically, two key attributes of the scene are used to select candidate regions by a color segmentation and multi-scale size matching: (i) color of the obstacle in the image; and (ii) physical dimensions of the object. However, the HITs must be presented using an intuitive and easy to understand interface (Kittur *et al.* 2011; Dow *et al.* 2012; Finnerty & Kucherbaev 2013). In PreVision we provide explicit real-time feedback to the

Turker on how well s/he has answered the HIT. Through this feedback, the Turker has the opportunity to correct his/her answer and converge to a more precise answer. We explain this feedback mechanism in the context of collecting the two scene-specific attributes.

HIT 1: Extracting Object Color

The first HIT focuses on extracting the color of the obstacle in the image. Finding the color of a specific object from an image using a vision algorithm is a complex task as lots of other objects of the same color may coexist in the image. One way to get the color of an object in the image is to tag that specific object, segment object by a color segmentation algorithm, create a color histogram that represents the object. The accuracy of color information thus depends on the segmenting accuracy. However, this process is useful only for the color retrieval process. For example, if we require the ground color instead of background color, this process may fail. The background of an object may consist of different type of objects, i.e., trees, sky, grass, and more. We can determine background color using the process, but it does not guarantee us to give the ground color. We use here the process of finding a specific feature of an object from an image asking a human to provide information about that feature. In the first HIT, we ask humans to provide best representative color of an obstacle on the sidewalk. Instead of directly asking the color of the obstacle (fire hydrant in Fig. 6.2(a)) we have designed a HIT where the user annotates a quadrilateral inside the object (shown in Fig. 6.2 (a)). Drawing a quadrilateral has two distinct advantages. First, a single object may have more than one color. Hence, a quadrilateral inside the object helps us infer the majority color. Second, it is easier for a Turker to annotate a shape inside the object rather than accurately point out the color of the object.

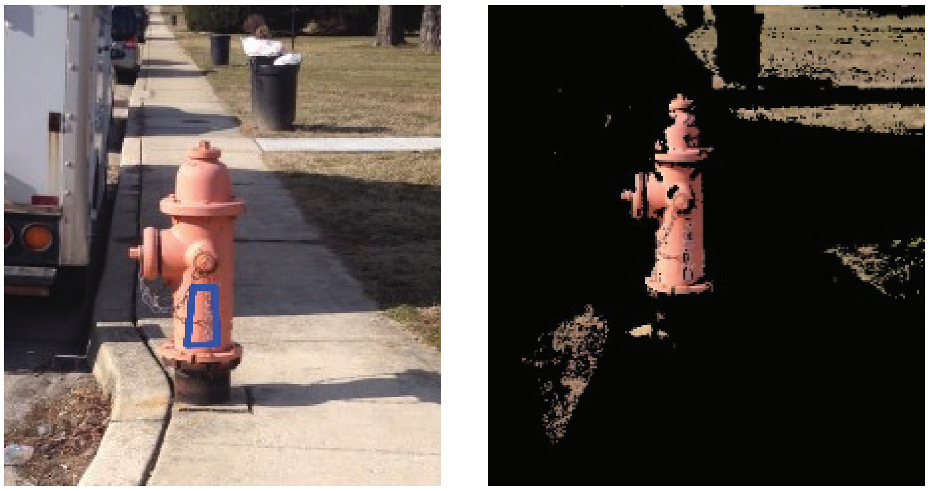


FIG. 6.2: Amazon Mechanical Turk task for collecting data on color. (left) Selecting object area and (right) corresponding feedback on the selection to the Turker.

A key component of PreVision is that it provides real-time feedback to Turkers while they perform the annotation. The real-time feedback serves two purposes. First, it allows the worker to correct mistakes by allowing them to see the impact of their input. In the most obvious case where they misunderstood the directions, the results of the feedback should make the error visible. For example, in the color extraction HIT, the system computes a segmentation immediately using the input and returns a visible result to the worker. If the Turker accidentally selects a region which does not represent the object color, the feedback results make a mistake visible. The Turker is allowed to correct before completing the HIT. This feature minimizes the bias when aggregating data from multiple Turkers. Thus, a human that does not necessarily understand the image processing operations (a non-expert) may still work to produce a result meant for the machine-vision system. The Turker can modify the annotations as many times as he wants. Fig. 6.3 shows three iterations where the Turker annotates the image, observes the segmented image, and then modifies the annotations to provide more accurate results.

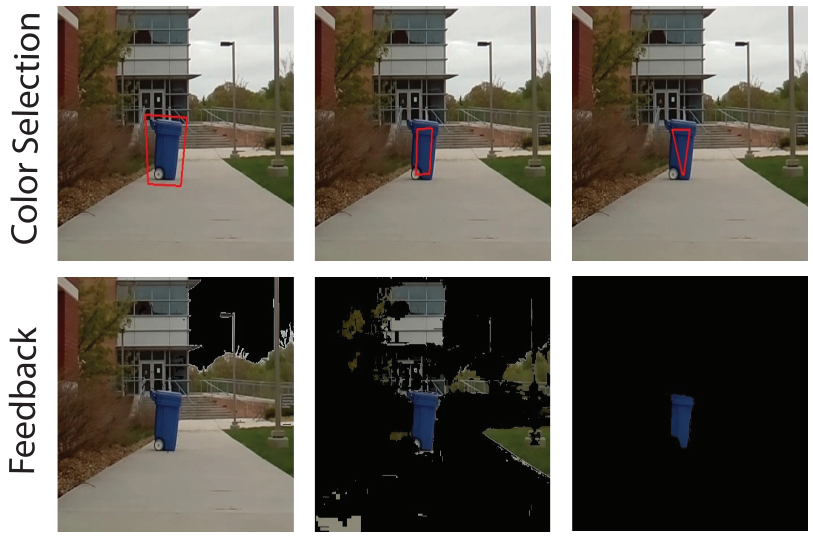


FIG. 6.3: Interactive color selection of an object by a Turker. (1^{st} column) is the first attempt the Turker made and s/he can see the segmented image on the bottom. In (2^{nd} column), s/he tried to correct her/his annotation based on the feedback shown on the bottom. When s/he is satisfied with her/his annotation (3^{rd} column), then s/he submits the response.

HIT 2: Annotating Physical Dimensions

Another attribute that the real-time detection algorithm used is the physical dimensions (height and width) of the vertical object. To elicit this data, we have designed another HIT. In this task, the Turker draws a bounding box around the object and provides the physical height and width of that bounding box. However, it is difficult for common Turkers to provide the absolute dimensions of the object accurately. Humans are better at relative rather than absolute measurements (Wu, Ooi, & He 2004). So, to this end, we provide a reference object, a United States 100-dollar bill, which measures $6.14\,\text{in} \times 2.61\,\text{in}$. When the Turker selects an object and provides the dimension, the selected part or object is cropped from the image and is displayed side by side with the dollar bill. The object is transformed into a new dimension using a perspective transform of the object using the height and width provided by the user. For example, if a Turker provides the dimension of an object as $22\,\mathrm{in} \times 9\,\mathrm{in}$, the Turker will see an object image, juxtaposed to a dollar bill and scaled to about three times the size of the dollar bill. In Fig. 6.4, we can see that one of the Turker annotations together with the preview image wherein the Turker provided $34\,\mathrm{in}$ as the height of the vertical pole. So, the vertical pole looks almost 5 times longer than the one-dollar bill. Based on this feedback, the Turker can change the dimensions he provided for the object dimensions.

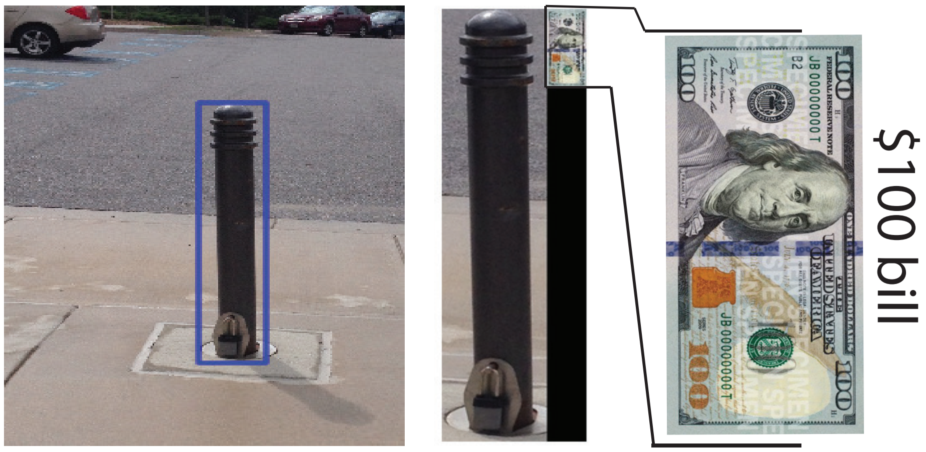


FIG. 6.4: Interactive physical dimension estimation. Turker selects an object using a bounding box (left) and provides height and width, gets the feedback (right side). The feedback is the comparison of the dimensions between a US 100-dollar bill and the selected object which is rescaled based on the Turker's provided dimension.

6.3.2 Data Aggregation

The PreVision subsystem of `PreSight` uses Turkers to extract *a priori* scene-specific data in the form of approximate geolocation of the accessibility problem, the color and physical dimensions of the object. For a single image this annotation is provided by multiple Turkers, and for a single scene, multiple volunteers may provide an image of the scene from different perspectives. This data, therefore, must be aggregated for the front-end RTVision system described in the next section. On average, we received 15 responses from Turkers for each HIT.

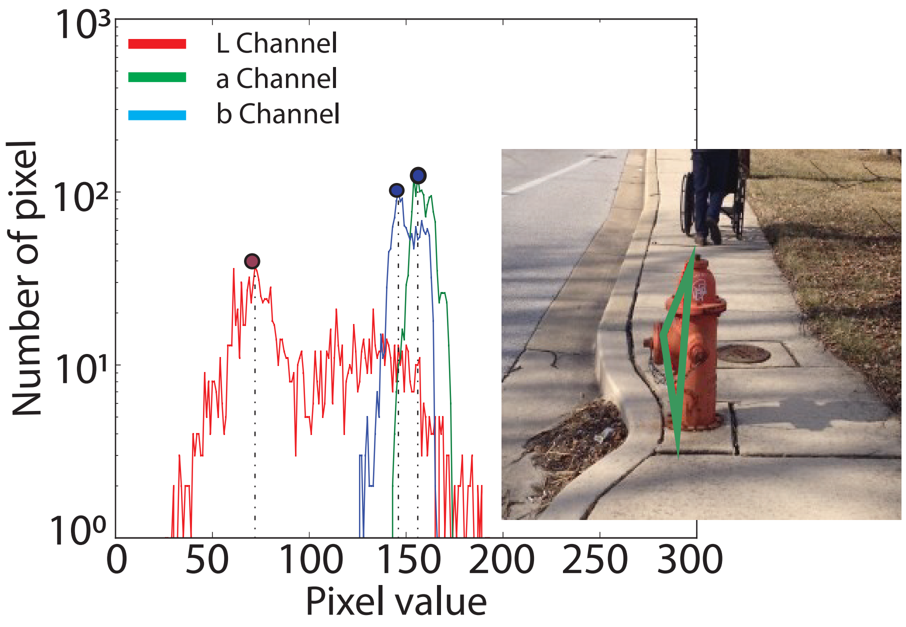


FIG. 6.5: Color extraction from user annotation. Histograms of annotation for all three channels are generated by PreVision. From the histograms, three peak values are extracted as the representing color of that annotated area.

PreVision aggregates data from Turkers responding to HIT 1 (object color). For this HIT the Turkers select an area within the object of interest (accessibility problem). The RTVision algorithm described in the next section uses a color value in the CIELAB color space. To extract the representative color of the object, we first crop the image based on the Turker selection and then generate a distribution of the pixel color values in the CIELAB color space for the cropped image. Fig. 6.5 illustrates a histogram for a single Turker selection. From the histogram, we pick the `mode` color value in each of the color channels. This color represents the most frequently occurring color value for pixels in the selection made by the Turker. This algorithm is robust to poor selections made by the Turkers. Because of the color channel histogram mechanism, the algorithm will always return the object color unless the selected area is dominated by objects other than its background. We aggregate the results across Turkers by taking an average of the pixel color values selected by each Turker.

The goal of the second HIT is to extract the physical dimensions of the object. For aggregating data from multiple Turkers, we can calculate the arithmetic mean of the dimension values, however, arithmetic mean is not robust to outliers. The Harmonic mean can handle outliers but it requires a large set of samples. We can also use a pruning-based outlier removal technique if we have a large training data. But, in our case the cost of recruiting Turkers prohibits the collection of a large training sample. In PreVision, therefore, we use the geometric mean of the dimension values to aggregate data from multiple Turkers. The geometric mean can handle a small training sample size and produces favorable results for our vision algorithm. The aggregate value of the dimensions populates a back-end contextual database in PreVision and used in the RTVision subsystem. The RTVision system is described below.

6.4 RTVision: Real-time Accessibility Problem Detection and Localization

Our front-end system, RTVision, is implemented on a smartphone as shown in Fig. 6.6. RTVision takes GPS readings and queries a locally cached version of the PreVision database for accessibility problems in the vicinity of the GPS coordinates. It then uses the embedded camera to take images and uses the scene-specific features to preprocess the images, which will select a set of regions in the images for further processing. Then, general object detection algorithms can be run on only specific regions instead of the entire image to detect the accessibility problems. The underlying RTVision preprocessing algorithm comprises two parts: (1) segmenting the image using the color information; and (2) searching for a target in possible scales.

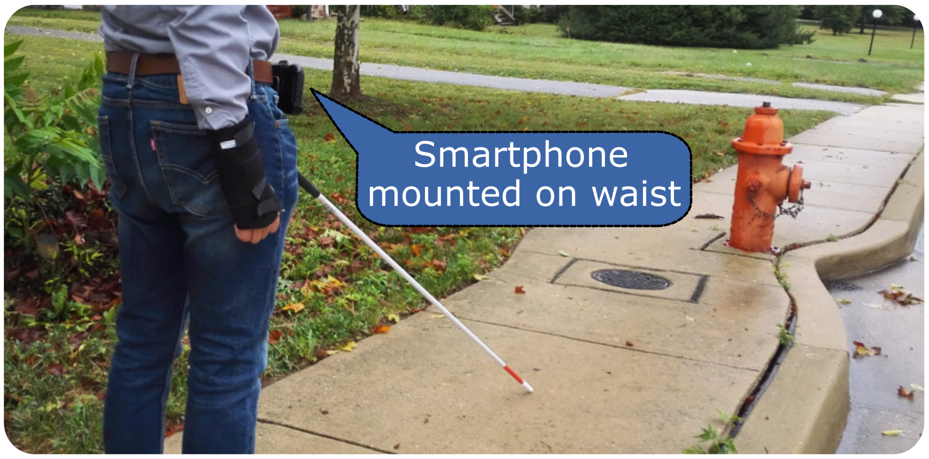


FIG. 6.6: Prototype of the front-end system, RTVision. It is implemented in a generic smartphone and utilizes the camera, the GPS sensor which is used to trigger the camera, and the Wi-Fi module used to download the contextual database (Li *et al.* 2017).

Color Image Segmentation Based on the color information provided by the PreVision system, a typical way to segment the colored image is transforming the image from RGB into HSV and segmenting the image in a particular range in the Hue-Saturation plane. However, the reference image might be taken with another user's mobile device. The HSV color value might drift or scatter across different devices. Thus, instead of using HSV value, the proposed system segments the color image in CIELAB space, which is device-independent.

Distance-based Multi-Scale Searching By knowing the aspect ratio of the target, we can also select regions with reasonable scales by multi-scale searching. Traditional scale-space (Lindeberg 1994) searching approaches down-sample the image by a scaling factor σ into sufficient levels to construct an image pyramid. Then a fixed-size feature template is used to match across the entire image in each layer of the pyramid. This method searches several scales of the object exhaustively. While it might get all possible matching, it also generates more false positives areas at different scales. It produces more candidate areas for the following expensive detection operation, and degrade the acceleration.

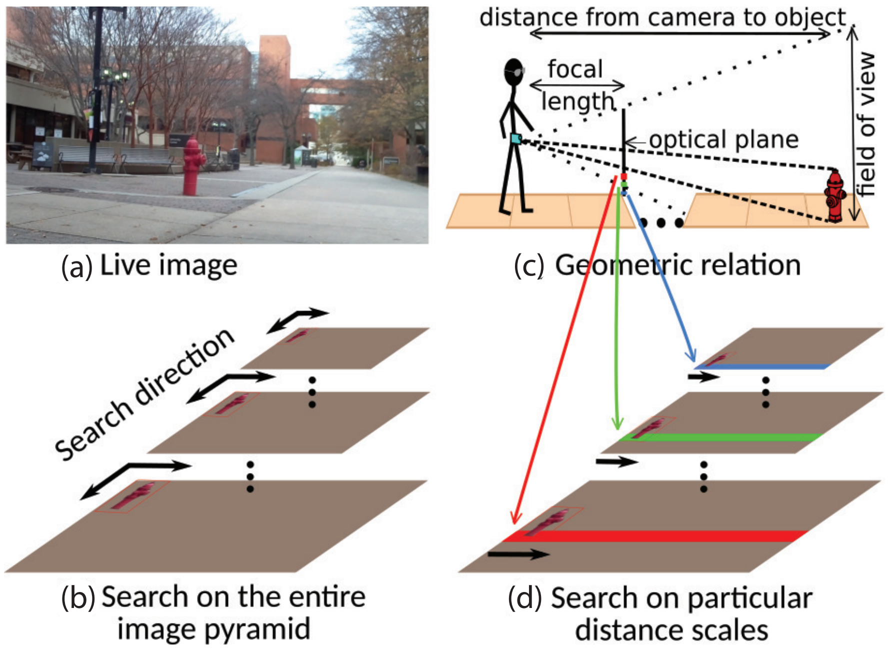

(a) Live image
distance from camera to object
focal length
optical plane
field of view
(c) Geometric relation
Search direction
(b) Search on the entire image pyramid
(d) Search on particular distance scales

FIG. 6.7: Multi-scale searching in the scale-space pyramid and proposed distance-based pyramid (Li *et al.* 2017). (a) the image to be searched in. (b) the scale-space pyramid in which the target template is searched across both directions. (c) the geometry relation of the mounted camera, target and the projections of that in the image. (d) the narrow band needed to be searched in a specific layer in the pyramid. Compared to using a scale-space pyramid, our method only requires searching in much narrower area, hence it incurs less false positives and is faster.

As the scale is varied with the distance to the target, the search time can be reduced significantly by only searching one scale at a specific distance. The smartphone is mounted onto the waist of the user at a fixed height for retrieving the distance (depth) information in the image pixels. With pre-calibration for each user, the height can be used for computing the distance based on the geometry relation. As illustrated in Fig. 6.7, the distance value can be warped onto the image pixels with the assumption that the ground between the camera and the target is parallel to the look-at vector of the camera. The system can eliminate false positives at unreasonable distance-scales using this approach. Also, the computation time for this preprocesses itself will also decrease since less convolution is needed compared to the scale-space method. RTVision generates the matching template T for the obstacle in its lowest scale based on the physical height and width of the obstacle provided by the PreVision.

6.5 Evaluation

The proposed system comprises of two parts. The accuracy of contextual database depends on the accuracy of turkers' provided data. In this section, we evaluate the turkers data as these are the vital information to develop successful obstacle detection and localization system.

6.5.1 Analysis of Turkers Data

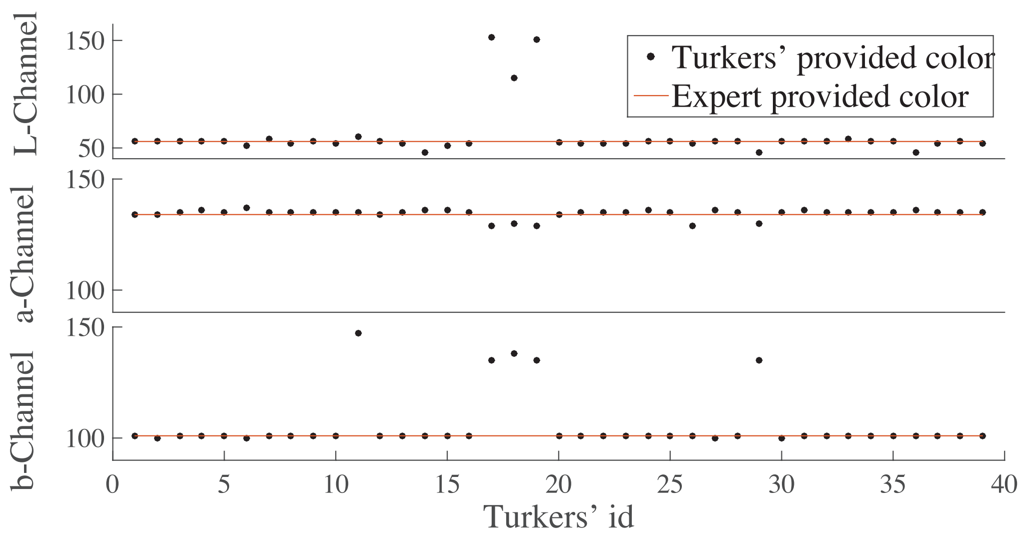


FIG. 6.8: Turkers' responses of object color in L*a*b color space. The distance from solid line to circular points represent the differences between Turkers' and experts' responses.

We next evaluate data provided by Turkers that has been used to generate contextual database off-line. The obstacle detection algorithm depends on three different features of an object. The PreVision thus has two different HITs to collect information about those two features from human crowd of Amazon Mechanical Turk. To evaluate PreVision data, we take images from 10 different sites around the campus and publish those to Amazon Mechanical Turk for human annotation. The color that largely represents the object is obtained from those responses using the process described in Section 6.3. The graph (see Fig. 6.8) shows that the difference between human responses and actual color is very small.

The aggregated colors of obstacles (see Fig. 6.9) show that for channels a and b in L*a*b color space, the human provided and expert provided colors are almost equal all the time. The only difference is the L-channel value. L-channel represents the luminosity of that object. In our designed HITs, the turkers select an area of their choice from the object in order to find its color. The brightness of the object is not equal all over its surface though the color of the object looks the same. And, there is no guarantee that both turkers and experts select the same area of that object. So, the L-channel value in turkers' annotation differs from that in experts'.

Our HIT of finding object dimension takes responses of human crowds for the object's height and width from an image. Humans have the intelligence of understanding object's dimensions if they ever have seen those objects in real world by their own; or at least if an object is present in the image that they are familiar with. If so, they will be able to compare the unknown objects to the known one to guess the dimensions of unknown object. But they cannot guess the dimensions if they have never seen the object present in the image, even with the feedback. But, most of the objects are known to the human crowds which are generally presents on sidewalks. To evaluate the correctness of Turkers' responses, we measured the objects height and width using a measuring tape. We also use geometric mean to aggregate human provided height and width of that object. The graph of

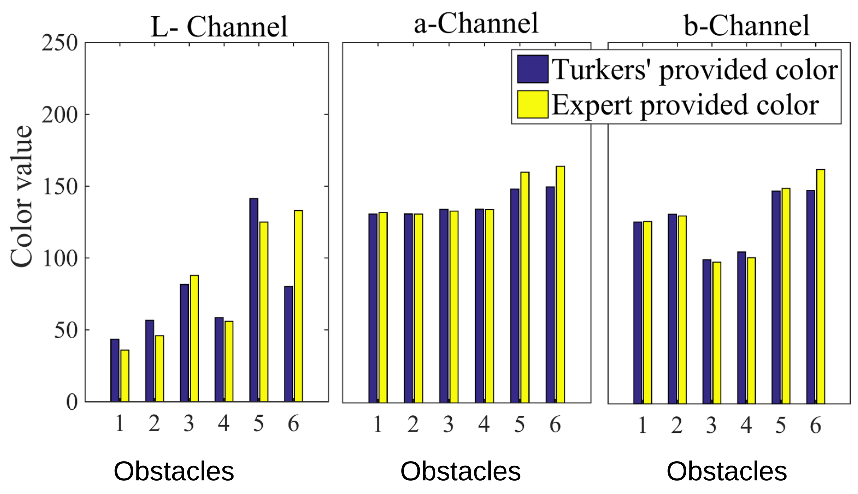


FIG. 6.9: Aggregated color value from Turkers' and experts' responses in three different channels (L, a and b).

the object's height and width (see Fig. 6.10) shows that most of the Turkers can correctly provide the object's height and width if the objects are familiar to them or if any other familiar object is present in that image.

This is obvious when we see the aggregated responses for all the objects. The results (see Fig. 6.11) from human crowd comes close to actual values for each object if that object is known to turkers, i.e, trash cans and fire hydrant. But if the object is an unknown one or a known one with variable dimensions such as poles, turkers would fail to guess correct dimensions most of the time.

`PreSight` uses human crowds to collect information for building contextual database. But data from human crowds have a cost in the form of money and time. In Amazon Mechanical Turk, we pay at least 1 US cent to get a single response. If we want at least 15 responses for unbiased data, it costs approximately 15 cents for each obstacle. We take into account the time to obtain 15 responses of each request to calculate the annotation

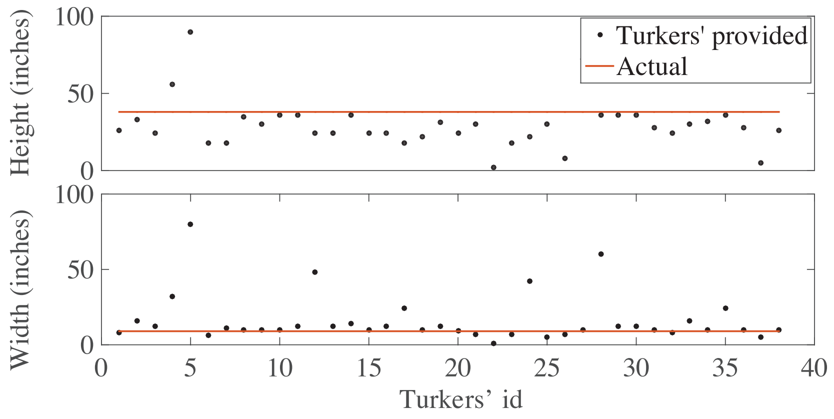


FIG. 6.10: Turkers' responses of object height and width. The distance from solid line to circular points represent the differences between Turkers' responses and the ground truth of that object.

time and can see that it takes almost a day (see Fig. 6.12) to get 15 responses for a request.

6.5.2 Evaluating `PreSight`

We evaluated feature extraction by human turkers. These features are the core part of the algorithm used in `PreSight` obstacle detection. The RTVision part of the `PreSight` uses these features. However, the performance of `PreSight` depends on the combination of PreVision and RTVision. We perform `PreSight`'s evaluation using data collected with our prototype system. We identified ten scenes with "obstacles-in-the-path" types of accessibility problems (Hara, Le, & Froehlich 2013) around our university campus for our experiments. We collect videos of these scenes by walking on sidewalks with our system. To take account of the variance in real-world luminance and viewing angle, we also collect video clips at different times of the day and different approaching directions. Totally, 40 video clips from the ten scenes are collected for the evaluation. All the videos are collected with a frame rate of 25 fps and a resolution of 1024×768.

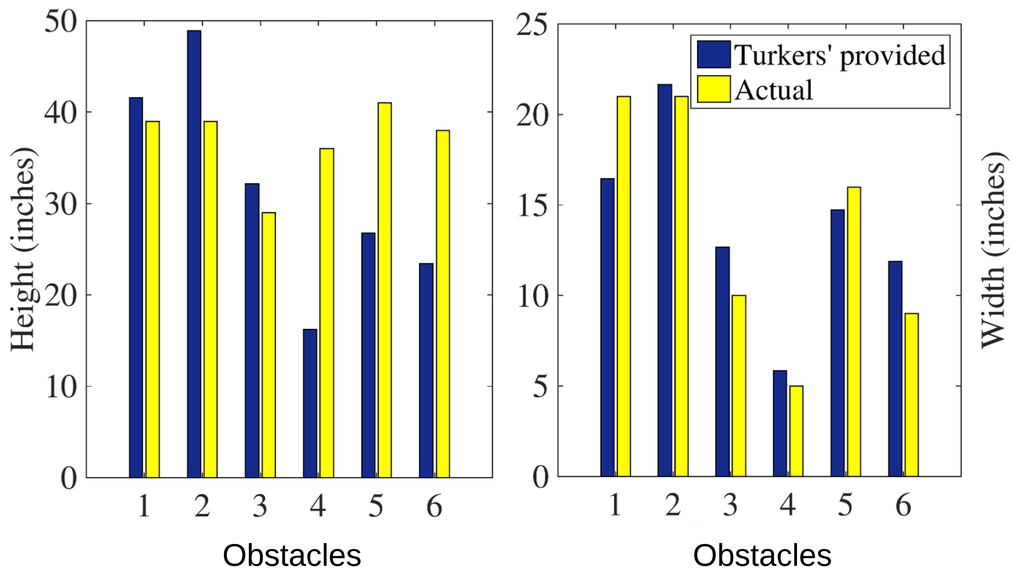


FIG. 6.11: Aggregation of height and width from Turkers' responses.

To evaluate the latency while also demonstrating that `PreSight` can enable real-time object detection in the embedded platform, we implement and evaluate the prototype on a generic smartphone with a 1.3-GHz quad-core ARM Cortex-A7 processor which was released on 2014.

We discussed the accuracy and latency of `PreSight` and compared with HOG and DPM. With our extracted features from the human crowd, the average operating time of HOG and DPM on the dataset is decreased by $1/12$. There is an overhead of computation in the preprocessing stage, but overall `PreSight` latency is decreased by $1/7$ and $1/10$ compared to HOG and DPM, respectively. The accuracy of the system is also affected by preprocessing steps. The F-score of HOG with `PreSight` is slightly higher than without `PreSight`. But, the F-score of DPM with `PreSight` is marginally lower than without `PreSight`. In the case of HOG, the false positive of the ignored area is re-

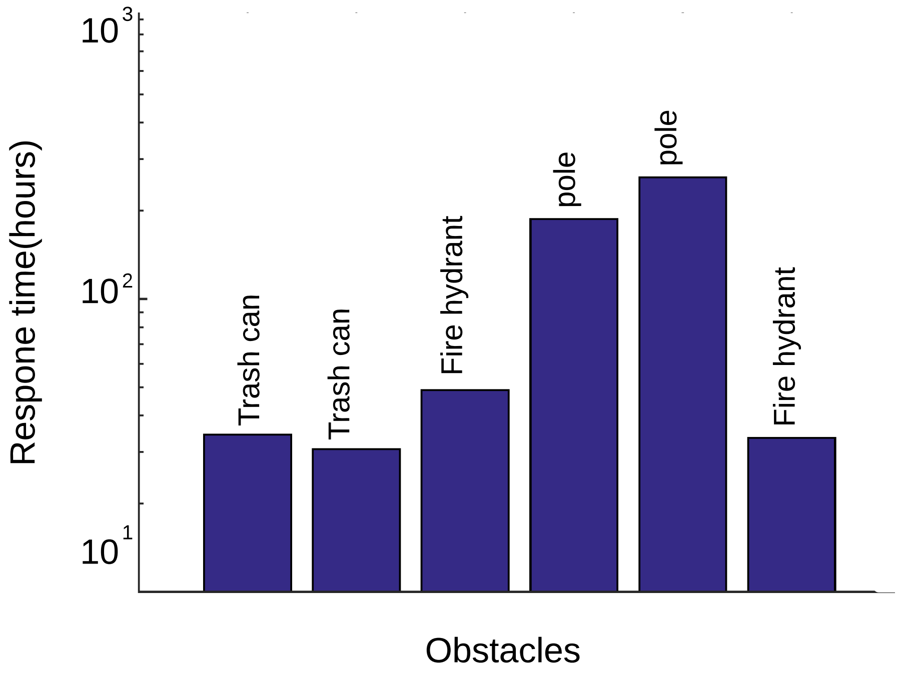


FIG. 6.12: The time of collecting fifteen responses from Amazon Mechanical Turk.

sponsible for better F-score, but DPM performs better because DPM is more accurate than that in the coarse-grained preprocessing. We also evaluated the distance-based multi-scale preprocessing, which performs computation two times faster than the multi-scale pyramid.

6.6 Conclusions

Table 6.1: Summary of cost and automation of processes in `PreSight`

Component	**Description**	**Automated**	**Cost per Scene**
UserSense	Collect scenes	x	One image
PreVision	Generate HITs	☑	< 1 sec
	Turkers annotate	☑	\$0.15, < 24 hr
	Aggregate data	☑	< 1 sec
RTvision	Cache database	☑	415 Byte
	Preprocess	☑	0.23 sec

In this chapter, we present `PreSight`, a prior-based vision system for speeding up general object detection and enabling real-time obstacle detection in a smartphone platform. The key idea behind `PreSight` is to preprocess the image with scene specific *a priori* information to select smaller areas for detection. The *a priori* used in `PreSight` includes color of the object and dimensions of the object. This scene-specific *a priori* information is collected in our system using mobile user "sensors" who take images of accessibility problems on sidewalks and Turkers who annotate information in the images. We have implemented a full functional prototype and automated most processes in low cost, as shown in TABLE 6.1. In the evaluation, we show that `PreSight` can accelerate general object detection algorithms, like HOG or DPM, in an average of 8 times faster without degrading the detection accuracy by much. While we demonstrate the feasibility of our approach in the context of sidewalk accessibility problem detection, we believe that such a strategy could be applied to navigation, indoor location, and augmented reality.

Chapter 7

TRADE OFF OF INTEGRATING HUMAN INTELLIGENCE

Integrating humans and computers to perform a specific task is not easy (Sheridan 2002). Human integration will bring human-level intelligence to a system which could be useful to solve many cognitive problems that are hard for computer alone. Humans are the ultimate designer of software; they know what would be the acceptance criteria. It may sound like integrating humans will make every cognitive problem easy to solve. However, there are no free lunch (Wolpert, Macready, & others 1997). Human intelligence aiding computer will bring both cognitive powers as well as their limitations. The flaws are related to human weakness in performing various tasks. Humans possess extraordinary cognitive skills, but their limitations in processing a high number of data, storing big data, social biases, and fatigue affects the system. In addition to those, adding humans instead of machines brings less automation, accuracy and increases latency, complexity, and cost. Thus, it would be a good idea to add humans where they perform better than machines considering different performance benchmarks. The quality of human input depends on various design techniques. However, the quality of data sometimes depends on the application requirements. We can consider the system response time as one such example. Some application requires real-time responses, but some may wait a few minutes or a few hours to get responses from the collaborative systems. Integrating humans also affect different metrics of an application. Human intelligence adds monetary cost, complexity, and response time.

In the following sections, we have discussed a) the trade-off of different techniques of integrating human input into machine algorithms, and b) different metrics of collaborative systems.

7.1 Trade-off of Different Integration Techniques

We have proposed a different human-machine collaborative system where we introduced different integration techniques. We optimized the selection of one method over another based on our application requirements. Each method has it's own pros and cons. We discuss the trade-off of these techniques below.

7.1.1 Real-time vs. Offline System

Human-machine integration is older than the internet. The general integration process is to harness knowledge from a user or an expert and integrate it into machine algorithms. With the advent of the internet and readily available online crowdsourcing platforms, intelligence can now be collected from a crowd who are working online. From experiments, we found that response time for a simple audiovisual task using crowdsourcing can range from three hours to ten hours compared to a machine that can do this thing within few milliseconds, even with the embedded system. This characteristic affects the system design to a great extent. With this high latency, the offline model is preferable to a real-time system. However, real-time systems are possible to develop with crowdsourcing responses with different design techniques and requirements. If we integrate human intelligence at feature extraction or preprocessing stage instead of in the decision loop, crowdsourcing response time will not be a bottleneck. In the `PreSight` and `LastStep`, the systems can respond in real-time. It is possible because the system builds the apriori database offline, and uses that information during evaluation steps. However, in weed identification, humans are involved in the decision loop. The users have to wait for responses on the scale of hours than to seconds or minutes.

7.1.2 Feedback Helps but No Incremental Improvement

The ability of human individuals to recognize thousands of object classes in cluttered scenes is one of the most surprising capabilities of visual perception, though the scenes may have different pose, change in illumination, and occlusion (Oliva & Torralba 2007). The cognitive power is not limited to image, but it is widespread over all real-world experience that humans encounter. However, knowledge and understanding of requirements vary from person to person. A human individual cannot perform any task if the proper guideline and feedback are absent. They can even perform well if the task provides the feedback (Dow & Kulkarni 2012) of their responses in real-time. The feedback could be self-assessment or external. It also could be the result of the very responses they made because humans have to understand the inherent process of the algorithm that applies their responses. For example, one algorithm requires only red color and will fail if there is noise in color. Another algorithm can perform well even if the color is a mixer of red and others, which form some reddish color. It requires to let the human crowd know how the intended algorithms are performing based on their responses. Moreover, human individuals can provide a better response if they have the option to correct their erroneous response and can identify errors seeing the feedback of their response (Hanrahan & Isaacs 2001; Vella 1996) and learn from their mistakes.

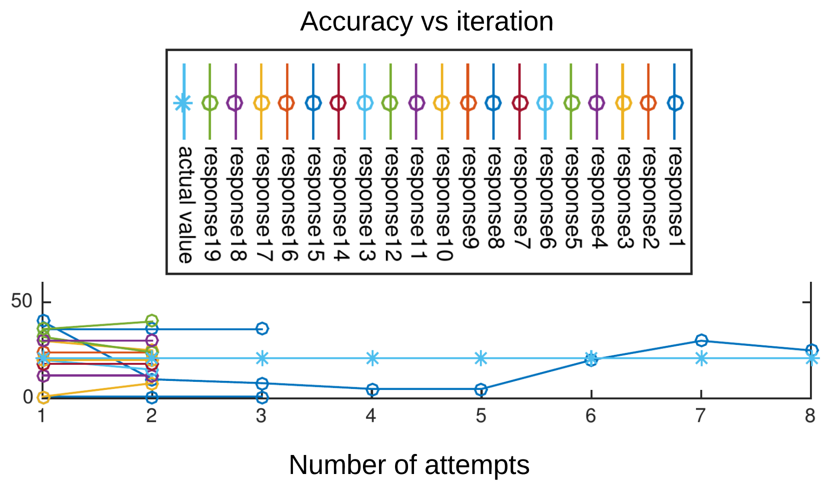


FIG. 7.1: Turker's are not very interested to correct. They do the minimal tasks they are required to do.

We have designed our crowdsourcing tasks providing feedback and an option to change their responses as many times as possible to make the response more accurate. We plot human behaviors (Figure 7.1) of acting on the feedback of their responses. The plot shows that most of the responders use the feedback process to correct their responses. One important finding here is that even though repeated feedback helps to improve the response, most of the people finalized their decision within two iterations. It is also worthwhile to mention that trying more times to correct did not end up with better responses.

7.1.3 Cost vs Accuracy

The cost of getting human responses is free if there is enough motivation for people to do it. The online crowdsourcing platform offers money for human intelligence and vice versa. People work in crowdsourcing platforms to earn money. They respond to queries by requesters and get paid by requesters for their responses. Logically, human individuals working for money will go for those tasks that pay more cash but require fewer efforts. A significant number of articles have proposed different designs for human tasks to optimize the cost against task difficulties. We studied the idea and provided separate payments, 5¢ and 1¢, for the same efforts. We analyzed payment vs. accuracy and payment vs. response time based on their responses. The experiments' outcomes clearly showed that the response time is linearly related to money it pays to perform the job. However, it showed that the accuracy of the response is not associated with the money it pays for the task. It implies that most of the crowd workers use their full effort to analyze and respond, but they always choose those tasks which pay more.

7.1.4 Hierarchical Crowdsourcing vs Unknown Crowd

The online portal, i.e., *Amazon mechanical Turk* or *crowdflower*, provides a communication interface between a requester and human crowd. Anyone can request to work on this platform without proving their expertise. Throwing tasks towards them give leads to get responses from a general crowd instead of domain experts. However, it is possible to train a group of individuals to perform specific tasks that assure the response's reliability. Upon completing training, the related tasks are posted, targeting those trained experts only, to get quality responses. There is another type of crowd who does not work on an online portal. They could be a group of coworkers or domain expertise. In the speech analysis project, we form a group of graduate students who form an expert group on labeling audio series from a group conversation. For `PreSight` and `LastStep`, we aggregated online crowd responses instead of going for experts. However, for weed identification, we have trained a set of an online crowd in the amazon mechanical turk platform. We formed a group of agricultural experts. All of these three groups responded to different situations with varied conditions. We call the process as hierarchical crowdsourcing. The non-expert crowd from online responds based on their knowledge, but the trained group provides quality responses using their expertise in that domain. However, the cost is increased to train each individual. Also, the latency increases as the trained group is a subset of the whole crowd, and the availability of significant worker decreases with the new constraint. The proposed hierarchical crowdsourcing request the unexpert crowd for responses. If they failed to respond, we would forward the requests to the expert crowd. If the combined group is unable to produce quality results, we would transfer the works to domain experts. Using hierarchical crowdsourcing, we were able to increase the accuracy to a great extent with minimized cost.

7.1.5 Crowd Demography

The availability of the human crowd is vital to improve latency. Most of the people working in crowdsourcing platforms are from Asia and the USA (Difallah, Filatova, & Ipeirotis 2018), especially one-fifth of them are from India. However, the activity of the Indian crowd increases at midnight in the USA. If we publish tasks in the evening or the morning from the USA, the latency is slightly lower than that of the tasks posted at day time because the activity of both geographical locations remains active at that time. We have published several tasks, 50% of them were published during midday, and 50% of them were published at the end of the day. The latency of evening distributed tasks is slightly lower than those published in the middle of the day. There is a cultural difference between Asian and western people. Because of that, the infrastructure, metric systems, and household and public properties have different dimensions, texture, and names. In our experiment, we asked the crowd about the trash can, pole, or fire hydrant height and width.

These objects are ubiquitous in the USA, but they are not alike in china or India. It is difficult for them to estimate the object's height and width, comparing them with unknown objects. We received some of the responses regarding trash can and fire hydrant far from actual dimensions from the crowd workers of that part of the world. So, it would be a good idea to ask workers from the same demographic if the answers for questions depend on some localized features.

7.1.6 Training vs. Aggregation

Another important trade-off is selecting trained crowds over aggregation or vice versa. The purpose of training is to improve the quality of workers to perform specific tasks. In crowdsourcing platforms, we can arrange a preparatory session where a set of golden standard tasks are published. Any workers can pick up the task to learn the domain. The task will provide feedback to the worker after they perform their part. They can do the training several times until they can answer all the questions correctly. Once the training completes, the crowdsourcing platforms add a badge against their identification. Upon completing the whole process, we publish the actual tasks targetting only those workers. Now they are trained on the domain and will be able to provide quality responses. The weed identification system takes decisions based on crowd responses. We followed the procedure to get the final answers from the decision model. However, the training process limits the accessibility of the number of crowds that we can ask for the actual answer. Moreover, each training session requires to pay a certain amount of money. We can also improve the quality of data with a better aggregation algorithm. We can design a robust aggregation algorithm that can avoid low-quality responses removing outliers and consider only genuine responses. If we need a quicker response, we can avoid training and go for better aggregation. Aggregation only method is more cost-effective than the training process. However, the accuracy may go down if the aggregation algorithms do not cover all edge cases. In that case, a trained crowd is more appropriate.

7.2 Humans Impact on System Metrics

Humans are a significant component of human-machine intelligent systems; thus, humans have a substantial impact on system metrics. Based on our research projects, we address the effects of humans on system accuracy, design complexity, latency, and cost.

7.2.1 Humans Effect on System Latency

The response time of human and machine are not similar. Some of the time, humans can respond quickly in the milliseconds' range when it comes to known and easy cognitive tasks. However, the process of asking humans, getting the response back, and integrate it into any machine algorithm is a time-consuming process. Integrating humans impacts the latency of the system as a whole. However, the requirements of all applications are not similar. Applications can wait from a few milliseconds to a few hours, and it depends on the problem domain. In the case of weed identification, latency does not matter. It can wait for a few hours or even for a day if the farmers get accurate results. However, all applications are not resilient to time. The response should be real-time in the case of `PreSight` and `LastStep` to inform the user about the possible obstacle and location of objects, respectively. If the process depends on humans for the whole process, a real-time response would not be possible. If a system depends on humans only at the training phase, with a smart design, real-time systems can be developed. The system can process the training set offline, taking help from humans, and build a knowledge database. Once apriori information is available, the system can process the evaluation using apriori data and provide feedback to users. The latency of the system depends on the complexity, size of the trained model, and the device's processing power. For smartphones, it is possible to process requests online. If the trained model is huge, it still needs lots of processing power. For sensors and embedded systems, it is still a big problem. The apriori database should be small enough for these small devices to respond in real-time. In the case of `PreSight` and `LastStep`, we proposed algorithms that process complicated features extraction offline, build an apriori database. We use the apriori database later to process the actual query in real-time. However, if the query processing phase requires feature extraction with the help of humans, real-time system development is not possible.

7.2.2 Human Effect on System Accuracy

Both humans and machines are a source of truth for a collaborative system. The accuracy of a system is the combined accuracy of humans and machines. There are lots of researches on machine accuracy of different problems, but here we are focusing on the human impact on collaborative system accuracy. The less we use human-provided information, the less impact it will have from human limitations and intelligence. We have studied the integration of human knowledge at three different functional phases of a collaborative system. We explain the impact of human integration on a system accuracy at all three steps below.

In our speech dynamics research, we integrated human intelligence at the data preprocessing stage. We applied both automated labeling algorithms and humans to detect speech and silence from a group conversation. First, we used the ICA (Hyvärinen & Oja 2000) algorithm, which can isolate channels for all participants. However, in a group conversation, all four participants' voices are mixed and impact the algorithm performance. We noticed that some participants speak louder than others. The louder voices add noise to each channel and make the isolation process very hard. The louder speaker interferes with the ICA channel isolation. Because of this reason, the filtering algorithm fails to detect the segment representing silence and speaking. We also applied the silent detection algorithm. The algorithm works well if the source is a single speaker. However, the algorithm fails for a group conversation. Humans deliver better performance in labeling audio, listening to all four voices as their cognitive power in these situations is excellent. However, the accuracy of the final results depends significantly on the labeling step. The accuracy of the labeled audio guarantees the accuracy of the trained model. The proper tools can assist humans in providing high-quality data and improving the efficiency of label data.

The accuracy of the extracted features reflects the accuracy of the final trained module. Features are a very abstract concept to human beings, and getting appropriate responses from them depends on the tasks we design by providing proper guidelines, training, and presenting questions in human-readable form. We expect quality results from people with natural thinking. If task design instructions are adequately forwarded, we can expect a quality response from human workers, but still, some individuals provide biased and random responses. We found from our experiment that a proper aggregation algorithm can remove those responses as outliers and can provide highly accurate information to the collaborative system.

The impact of humans is pivotal if the decision model of any system depends on human intelligence. In weed identification, we found that a proper decision algorithm can ensure a high accuracy system. We also tried hierarchy crowdsourcing, where we forward the tasks to expert groups if the crowd failed to provide a correct response. The process ensures the system with a near hundred percent accuracy.

7.2.3 Design Complexity

Traditional machine learning is widely studied and has lots of existing tools to develop, train, and test. If the problem domains are well-known, there are lots of tools and tools available for labeling, feature extraction, and training. However, if we add humans to any stage, it adds an extra level of complexity to the system. In our speech dynamics, group conversation monitoring is not traditional. However, the mixture process of different audio channels is one of the important audio features. We use audacity audio tools that have the features and helps us to listen to all four channels together to annotate the group conversation. The feature extraction is more complicated than labeling as the features are very different from application to application. In the case of `LastStep`, we designed a web interface to extract color and dimensions. In the case of `PreSight`, we developed an interface to annotate google map tile. For our weed identification, we designed the mobile interface to integrate expert responses and an amazon mechanical turk interface, to get crowd input to the applications. In all four cases, lots of research and brainstorming were put behind the idea and development of different interfaces and algorithms. The human intelligence integration in decision making requires extra care in designing algorithm as well as the interface. The decision algorithm consolidates machine intelligence as well as a cognitive response from the human crowd using a probabilistic decision model and threshold techniques. All these steps are necessary because of human intelligence integration. If we could avoid humans, we could easily avoid all these complications in designing these systems.

7.2.4 Cost

The cost of getting a response from an algorithm is an essential part of any system design. The machine cost, i.e., the server, power, and computation cost, is typical for any machine algorithm as well as for the collaborative system. In addition to these costs, human intelligence adds extra expense. There is a monetary cost for every response from humans. Each labeling and categorizing tasks costs money, and it linearly increases with the increase in task complexity or with the decrease of latency.

The monetary cost can be of two types. In preprocessing steps, labeling is a one time process and does not increase cost over time. However, feature extraction may be needed in the decision process as well as in the training time. If it is not required during decision making, the cost is also a one time process. However, if the system accepts help from human intelligence in the decision phase, for each response, a certain amount of money has to be spent. We can optimized cost designing robust aggregation algorithms. In our experiment with bus stop localization and obstacle detection, we designed simple tasks, and we kept the number of responses bellow ten, which costs almost ten cents per task. Two separate tasks provide the necessary information for both of the projects, which costs twenty cents per bus stop or obstacle information collections. In the case of weed identification, the decision model works in probabilistic fashion and requires at least ten responses or as much as thirty, which costs us an average of twenty responses. The average cost for each request was twenty cents.

Chapter 8

CONCLUSION AND FUTURE DIRECTIONS

Human and machine collaboration is one of the essential subsets of AI. Researchers proposed several techniques to solve different problems integrating humans and machines. Many researchers suggested methods of resolving issues integrating humans in the decision loops. Some others proposed guidelines for improving feature extraction techniques, and some of them provided procedures and guidelines to improve data quality. However, no comprehensive analysis and instruction was offered, covering all these situations. We proposed a collaborative system design paradigm that provides a detailed analysis of a collaborative system, different techniques to improve performance, and trade-off of integrating humans in a system.

Analyzing a collaborative system, we found three functional phases where human intelligence can be integrated - data preprocessing, feature extraction, and decision. Humans can help to improve quality in one or more of these functional phases. We also have analyzed how a human-machine system can reduce human impact in the long run. We have developed an end to end system covering all of these conditions and provided detailed steps of designing a collaborative system.

We designed an agricultural system to help farmers identify weeds in their crop fields, analyzing weed images through hierarchical crowdsourcing. We did not have enough data to train a machine learning model. We trained a model partially to retrieve multiple competitive outcomes and integrated human intelligence to identify the correct weed from a set of competitive results. We designed a bus stop localization system for visually impaired people applying image processing algorithm to detect bus stop signs and human intelligence to localize other resources. A two-dimensional layout of resources is a vital step for the system. However, it is not easy to obtain layout features with unsupervised learning or with less powerful embedded devices. Human intelligence is widely used in preprocessing data. We developed a machine learning model to analyze speech dynamics in a group conversation by analyzing respiratory sensor data. Human intelligence helped to segment sensor data listening to the audio recording of a group conversation, which is very hard for a machine. We also address the situation and necessary condition where a human can be integrated as a temporary solution and can be removed when certain conditions are fulfilled. We designed a sidewalk obstacle detection system for visually impaired people in low powered devices. We developed an object detection algorithm that uses only color and dimensions provided by humans to detect an object in real-time. However, color and dimensions can automatically be extracted if there is a huge amount of data. When the dataset meets the necessary condition, human intervention can be reduced or removed in the long run.

Each of the systems has a different level of dependency on humans and data instances. So, there is a significant performance, cost, and development complexity difference among each system. The system designer will have to decide what system will fit with their needs and what they have to consider to design a human-machine collaborative system to be successful.

We provided a design paradigm by developing applications in the audio and visual domains. We provided techniques to annotate audio and video data to extract features from images and to make decisions analyzing visual features. However, we have not covered other domains that are important to provide a generic guideline. A generic architecture can be developed in the future with a detailed analysis of all critical fields. This may help the system designer to get all information at one point and decide a template pattern based on their problem domain.

. REFERENCES

[1] Acapella. Accessed: 2019-08-23. Your voice matters. https://www.acapela-group.com.

[2] Blindness and low vision — american foundation for the blind. (Accessed on 04/20/2019).

[3] Ahmed, M. Y.; Kenkeremath, S.; and Stankovic, J. 2015. Socialsense: A collaborative mobile platform for speaker and mood identification. In *European Conference on Wireless Sensor Networks*, 68–83. Springer.

[4] Authority, S. P. T. Accessed April 4, 2019.

[5] Bari, R.; Adams, R. J.; Rahman, M. M.; Parsons, M. B.; Buder, E. H.; and Kumar, S. 2018. rconverse: Moment by moment conversation detection using a mobile respiration sensor. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies* 2(1):1–27.

[6] Barowy, D., and Curtsinger, C. 2012. Automan: A platform for integrating human-based and digital computation. *ACM SIGPLAN* ....

[7] Borenstein, J. 2001. The guidecane-applying mobile robot technologies to assist the visually impaired. *Systems, Man and Cybernetics, Part A: Systems* ....

[8] Buhrmester, M.; Kwang, T.; and Gosling, S. D. 2011. Amazon's mechanical turk a new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science* 6(1):3–5.

[9] Campbell, M.; Hoane Jr, A. J.; and Hsu, F.-h. 2002. Deep blue. *Artificial intelligence* 134(1-2):57–83.

[10] Charan, R. 1991. How networks reshape organizations–for results. *Harvard Business Review* 69(5):104–115.

[11] Christ, M.; Braun, N.; Neuffer, J.; and Kempa-Liehr, A. W. 2018. Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package). *Neurocomputing* 307:72–77.

[12] Committee, I. S. A., et al. 2006. Invasive species definition clarification and guidance white paper. national invasive species council. *US Department of Agriculture, National Agricultural Library. Washington, DC Available at http://www.invasivespeciesinfo.gov/docs/council/isacdef.pdf.*

[13] Cook, K. A., and Thomas, J. J. 2005. Illuminating the path: The research and development agenda for visual analytics. Technical report, Pacific Northwest National Lab.(PNNL), Richland, WA (United States).

[14] daid, n. 2020. Emptyepsilon - multiplayer spaceship bridge simulator. https://daid.github.io/EmptyEpsilon/. (Accessed on 02/04/2020).

[15] Dalal, N., and Triggs, B. 2005. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, 886–893. IEEE.

[16] De Rocquigny, E.; Devictor, N.; and Tarantola, S. 2008. *Uncertainty in industrial practice: a guide to quantitative uncertainty management*. John Wiley & Sons.

[17] Demuth, H. B.; Beale, M. H.; De Jess, O.; and Hagan, M. T. 2014. *Neural network design*. Martin Hagan.

[18] Difallah, D.; Filatova, E.; and Ipeirotis, P. 2018. Demographics and dynamics of

mechanical turk workers. In *Proceedings of the eleventh ACM international conference on web search and data mining*, 135–143.

[19] Dogs, G.; Catapult, F. C.; and Microsoft. Citiesunlocked – citiesunlocked. (Accessed on 04/20/2019).

[20] Douglas, S. a., and Kirkpatrick, A. E. 1999. Model and representation: the effect of visual feedback on human performance in a color picker interface. *ACM Transactions on Graphics* 18(2):96–127.

[21] Dow, S. P., and Klemmer, S. R. 2011. Shepherding the Crowd : Managing and Providing Feedback to Crowd Workers. *Operations Research* 1669–1674.

[22] Dow, S., and Kulkarni, A. 2012. Shepherding the crowd yields better work. *. . . Cooperative Work* 1013–1022.

[23] Dow, S.; Kulkarni, A.; Klemmer, S.; and Hartmann, B. 2012. Shepherding the crowd yields better work. In *CSCW*, 1013–1022.

[24] Duarte, M. 2015. Notes on scientific computing for biomechanics and motor control.

[25] Duncan, S. 1972. Some signals and rules for taking speaking turns in conversations. *Journal of personality and social psychology* 23(2):283.

[26] Eakin, H., and Luers, A. L. 2006. Assessing the vulnerability of social-environmental systems. *Annu. Rev. Environ. Resour.* 31:365–394.

[27] Felzenszwalb, P. F.; Girshick, R. B.; McAllester, D.; and Ramanan, D. 2010. Object detection with discriminatively trained part-based models. *IEEE TPAMI* 32(9):1627–1645.

[28] Ferguson, G.; Allen, J. F.; et al. 1998. Trips: An integrated intelligent problem-solving assistant. In *Aaai/Iaai*, 567–572.

[29] Finnerty, A., and Kucherbaev, P. 2013. Keep it simple: Reward and task design in crowdsourcing. *Proceedings of the …* 2–5.

[30] Fitts, P. M. 1951. Human engineering for an effective air-navigation and traffic-control system.

[31] Guo, B.; Wang, Z.; Yu, Z.; Wang, Y.; Yen, N. Y.; Huang, R.; and Zhou, X. 2015. Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm. *ACM Computing Surveys (CSUR)* 48(1):7.

[32] Guyon, I., and Elisseeff, A. 2003. An introduction to variable and feature selection. *Journal of machine learning research* 3(Mar):1157–1182.

[33] Hall, M. A. 1999. Correlation-based feature selection for machine learning.

[34] Hanrahan, S. J., and Isaacs, G. 2001. Assessing self-and peer-assessment: the students' views. *Higher education research and development* 20(1):53–70.

[35] Hara, K.; Azenkot, S.; Campbell, M.; Bennett, C. L.; Le, V.; Pannella, S.; Moore, R.; Minckler, K.; Ng, R. H.; and Froehlich, J. E. 2015. Improving public transit accessibility for blind riders by crowdsourcing bus stop landmark locations with google street view: An extended analysis. *ACM TACCESS* 6(2):5.

[36] Hara, K.; Le, V.; and Froehlich, J. 2013. ing and google street view to identify street-level accessibility problems. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13* 631–640.

[37] Hersh, M. A., and Johnson, M. A. 2008. *Assistive Technology for Visually Impaired and Blind People*. Springer London.

[38] Holland, J. H., et al. 1992. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.

[39] Horvitz, E. 1999. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 159–166. ACM.

[40] Hoy, M. B. 2018. Alexa, siri, cortana, and more: an introduction to voice assistants. *Medical reference services quarterly* 37(1):81–88.

[41] Hyvärinen, A., and Oja, E. 2000. Independent component analysis: algorithms and applications. *Neural networks* 13(4-5):411–430.

[42] Iafrate, F. 2018. *Artificial intelligence and big data: The birth of a new intelligence*. John Wiley & Sons.

[43] Ipeirotis, P. G. 2010a. Analyzing the Amazon Mechanical Turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students* 17(2):16.

[44] Ipeirotis, P. G. 2010b. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students* 17(2):16–21.

[45] Ismail, Y. 2019. Internet of things (iot) for automated and smart applications.

[46] Itti, L., and Koch, C. 2001. Computational modelling of visual attention. *Nature reviews neuroscience* 2(3):194–203.

[47] Ivanchenko, V.; Coughlan, J.; Gerrey, W.; and Shen, H. 2008. Computer vision-based clear path guidance for blind wheelchair users. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*, Assets '08, 291–292. ACM.

[48] Jepson, K. 2005. Conversations—and negotiated interaction—in text and voice chat rooms. *Language Learning & Technology* 9(3):79–98.

[49] Jordan, N. 1963. Allocation of functions between man and machines in automated systems. *Journal of applied psychology* 47(3):161.

[50] Kazman, R.; Klein, M.; Barbacci, M.; Longstaff, T.; Lipson, H.; and Carriere, J. 1998. The architecture tradeoff analysis method. In *Proceedings. Fourth IEEE International Conference on Engineering of Complex Computer Systems (Cat. No. 98EX193)*, 68–78. IEEE.

[51] Kim, T. J.; Chu, M.; Brdiczka, O.; and Begole, J. 2009. Predicting shoppers' interest from social interactions using sociometric sensors. In *CHI'09 Extended Abstracts on Human Factors in Computing Systems*, 4513–4518. ACM.

[52] Kittur, a.; Smus, B.; Khamkar, S.; and Kraut, R. 2011. Crowdforge:cCrowdsourcing complex work. *Proceedings of the 24th annual ACM symposium on User interface software and technology* 43–52.

[53] Klasnja, P.; Consolvo, S.; Choudhury, T.; Beckwith, R.; and Hightower, J. 2009. Exploring privacy concerns about personal sensing. In *International Conference on Pervasive Computing*, 176–183. Springer.

[54] Kononenko, I. 2001. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine* 23(1):89–109.

[55] Kulkarni, A.; Can, M.; and Hartmann, B. 2011. Turkomatic: automatic recursive task and workflow design for mechanical turk. *CHI'11 Extended Abstracts on Human* ....

[56] Kumar, S.; Al'Absi, M.; Beck, J.; Ertin, E.; and Scott, M. 2014. Behavioral monitoring and assessment via mobile sensing technologies. *Behavioral Healthcare Technol.: Using Science-Based Innovations to Transform Practice* 27:621–624.

[57] Ledford, H. 2015. How to solve the world's biggest problems. *Nature News* 525(7569):308.

[58] Lee, M. H. 2013. *Intelligent robotics*. Springer Science & Business Media.

[59] Li, Z.; Rahman, M.; Robucci, R.; and Banerjee, N. 2016. Laststep: Vision-based bus stop localization and mapping for improving accessibility for blind riders. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, 257–258. ACM.

[60] Li, Z.; Rahman, M.; Robucci, R.; and Banerjee, N. 2017. Presight: Enabling real-time detection of accessibility problems on sidewalks. In *2017 14th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, 1–9. IEEE.

[61] Liddicoat, A. J. 2011. *An introduction to conversation analysis*. Bloomsbury Publishing.

[62] Lindeberg, T. 1994. Scale-space theory: a basic tool for analyzing structures at different scales. *Journal of Applied Statistics* 21(1-2):225–270.

[63] Lorig, T. S. 2007. *The Respiratory System*. Cambridge University Press, 3 edition. 231–244.

[64] Lowe, D. G. 2004. Distinctive image features from scale-invariant keypoints. *International journal of computer vision* 60(2):91–110.

[65] Lowry, C. S., and Fienen, M. N. 2013. Crowdhydrology: Crowdsourcing hydrologic data and engaging citizen scientists. *Groundwater* 51(1):151–156.

[66] Lu, H.; Brush, A. B.; Priyantha, B.; Karlson, A. K.; and Liu, J. 2011. Speakersense: Energy efficient unobtrusive speaker identification on mobile phones. In *International conference on pervasive computing*, 188–205. Springer.

[67] Lu, H.; Frauendorfer, D.; Rabbi, M.; Mast, M. S.; Chittaranjan, G. T.; Campbell, A. T.; Gatica-Perez, D.; and Choudhury, T. 2012. Stresssense: Detecting stress in unconstrained acoustic environments using smartphones. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, 351–360. ACM.

[68] Lux, M., and Chatzichristofis, S. A. 2008. Lire: lucene image retrieval: an extensible java cbir library. In *Proceedings of the 16th ACM international conference on Multimedia*, 1085–1088. ACM.

[69] Lymberopoulos, D.; Liu, J.; Yang, X.; Choudhury, R. R.; Handziski, V.; and Sen, S. 2015. A realistic evaluation and comparison of indoor location technologies: Experiences and lessons learned. In *IPSN*, 178–189. ACM.

[70] Madakam, S.; Lake, V.; Lake, V.; Lake, V.; et al. 2015. Internet of things (iot): A literature review. *Journal of Computer and Communications* 3(05):164.

[71] Madan, A.; Moturu, S. T.; Lazer, D.; and Pentland, A. S. 2010. Social sensing: obesity, unhealthy eating and exercise in face-to-face networks. In *Wireless Health 2010*, 104–110. ACM.

[72] Manduchi, R., and Coughlan, J. 2012. (computer) vision without sight. *Communications of the ACM* 55(1):96–104.

[73] Mantere, M.; Uusitalo, I.; Sailio, M.; and Noponen, S. 2012. Challenges of machine learning based monitoring for industrial control system networks. In *2012 26th Interna-*

tional Conference on Advanced Information Networking and Applications Workshops, 968–972. IEEE.

[74] Marr, D. 1977. Artificial intelligence—a personal view. *Artificial Intelligence* 9(1):37–48.

[75] Martin, J. 2012. *The meaning of the 21st century: A vital blueprint for ensuring our future*. Random House.

[76] Maunder, A., et al. 1972. *Agricultural Extension. A Reference Manual*. ERIC.

[77] Maynard, D., and Peräkylä, A. 2006. *Language and Social Interaction*. 233–257.

[78] McCarthy, J., and Hayes, P. J. 1981. Some philosophical problems from the standpoint of artificial intelligence. In *Readings in artificial intelligence*. Elsevier. 431–450.

[79] Mengis, J., and Eppler, M. J. 2005. Understanding and enabling knowledge sharing in conversations: a literature review and management framework. In *2nd Annual Conference on Knowledge Management in the Asian Pacific (KMAP), Wellington (New Zealand)*. Citeseer.

[80] Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529.

[81] Monostori, L., and Prohaszka, J. 1993. A step towards intelligent manufacturing: Modelling and monitoring of manufacturing processes through artificial neural networks. *CIRP annals* 42(1):485–488.

[82] Monowar Hossain, Nusrat Nasrin, A. N. N. A., and Hnat, T. 2019 (accessed July 8, 2019). Advancing biomedical discovery and improving health through mobile sensor big data.

[83] Montes, A.; Salvador, A.; and Giró i Nieto, X. 2016. Temporal activity detection in untrimmed videos with recurrent neural networks. *CoRR* abs/1608.08128.

[84] Moran, J., and Desimone, R. 1985. Selective attention gates visual processing in the extrastriate cortex. *Science* 229(4715):782–784.

[85] Müller, H.; Müller, W.; Squire, D. M.; Marchand-Maillet, S.; and Pun, T. 2001. Performance evaluation in content-based image retrieval: overview and proposals. *Pattern Recognition Letters* 22(5):593–601.

[86] Newell, A.; Simon, H. A.; et al. 1972. *Human problem solving*, volume 104. Prentice-Hall Englewood Cliffs, NJ.

[87] Olguín, D. O., and Pentland, A. S. 2008. Social sensors for automatic data collection. *AMCIS 2008 Proceedings* 171.

[88] Oliva, A., and Torralba, A. 2007. The role of context in object recognition. *Trends in cognitive sciences* 11(12):520–527.

[89] Povey, D.; Ghoshal, A.; Boulianne, G.; Burget, L.; Glembek, O.; Goel, N.; Hannemann, M.; Motlicek, P.; Qian, Y.; Schwarz, P.; et al. 2011. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number CONF. IEEE Signal Processing Society.

[90] Price, H. E. 1985. The allocation of functions in systems. *Human factors* 27(1):33–45.

[91] Rachlin, H. 2012. Making ibm's computer, watson, human. *The Behavior Analyst* 35(1):1–16.

[92] Rachuri, K. K.; Musolesi, M.; Mascolo, C.; Rentfrow, P. J.; Longworth, C.; and Aucinas, A. 2010. Emotionsense: a mobile phones based adaptive platform for experimental social psychology research. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, 281–290. ACM.

[93] Rahman, M. M.; Ali, A. A.; Plarre, K.; Al'Absi, M.; Ertin, E.; and Kumar, S. 2011. mconverse: Inferring conversation episodes from respiratory measurements collected in the field. In *Proceedings of the 2nd Conference on Wireless Health*, 1–10.

[94] Roh, Y.; Heo, G.; and Whang, S. E. 2018. A survey on data collection for machine learning: a big data-ai integration perspective. *arXiv preprint arXiv:1811.03402*.

[95] Russell, S. J., and Norvig, P. 2016. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,.

[96] Schegloff, E. 1992. Introduction, harvey sacks' lectures on conversation. *Lectures on conversation*.

[97] Selfridge, O. G. 1958. Pandemonium: A paradigm for learning. *the Mechanisation of Thought Processes, 1958*.

[98] Shahaf, D., and Amir, E. 2007. Towards a theory of ai completeness. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, 150–155.

[99] Sheng, V. S.; Provost, F.; and Ipeirotis, P. G. 2008. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 614–622. ACM.

[100] Sheridan, T. B. 2002. *Humans and automation: System design and research issues.*

Wiley series in system engineering and management: HFES issues in human factors and ergonomics series. Santa Monica, CA, US: Human Factors and Ergonomics Society.

[101] Shoval, S.; Ulrich, I.; and Borenstein, J. 2003. Navbelt and the guide-cane: obstacle-avoidance systems for the blind and visually impaired. *Robotics Automation Magazine, IEEE* 10(1):9–20.

[102] Simon, H. A. 2019. *The sciences of the artificial.* MIT press.

[103] Son, D.; Lee, J.; Qiao, S.; Ghaffari, R.; Kim, J.; Lee, J. E.; Song, C.; Kim, S. J.; Lee, D. J.; Jun, S. W.; et al. 2014. Multifunctional wearable devices for diagnosis and therapy of movement disorders. *Nature nanotechnology* 9(5):397.

[104] Stone, P.; Brooks, R.; Brynjolfsson, E.; Calo, R.; Etzioni, O.; Hager, G.; Hirschberg, J.; Kalyanakrishnan, S.; Kamar, E.; Kraus, S.; et al. 2016. Artificial intelligence and life in 2030. one hundred year study on artificial intelligence: Report of the 2015-2016 study panel. *Stanford University, Stanford, CA, http://ai100. stanford. edu/2016-report. Accessed: September* 6:2016.

[105] Sun, Y.; Wang, X.; and Tang, X. 2014. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1891–1898.

[106] Szolovits, P.; Patil, R. S.; and Schwartz, W. B. 1988. Artificial intelligence in medical diagnosis. *Annals of internal medicine* 108(1):80–87.

[107] Terveen, L. G. 1995. Overview of human-computer collaboration. *Knowledge-Based Systems* 8(2-3):67–81.

[108] Terven, J. R.; Salas, J.; and Raducanu, B. 2014. New opportunities for computer

vision-based assistive technology systems for the visually impaired. *IEEE Computer* 47(4):52–58.

[109] Valdés-Pérez, R. E. 1999. Principles of human—computer collaboration for knowledge discovery in science. *Artificial Intelligence* 107(2):335–346.

[110] van Etten, J. 2011. Crowdsourcing crop improvement in sub-saharan africa: A proposal for a scalable and inclusive approach to food security. *IDS bulletin* 42(4):102–110.

[111] Van Laerhoven, K., and Cakmakci, O. 2000. What shall we teach our pants? In *Digest of Papers. Fourth International Symposium on Wearable Computers*, 77–83. IEEE.

[112] Vella, F. 1996. Enhancing learning through self-assessment. *Biochemical Education* 3(24):183.

[113] Waber, B. N.; Olguin Olguin, D.; Kim, T.; and Pentland, A. 2010. Productivity through coffee breaks: Changing social networks by changing break structure. *Available at SSRN 1586375*.

[114] Watkins, C. J., and Dayan, P. 1992. Q-learning. *Machine learning* 8(3-4):279–292.

[115] WHO. 2010. Eastern mediterranean region. (Accessed on 04/20/2019).

[116] Wolfe, J. M., and Horowitz, T. S. 2004. What attributes guide the deployment of visual attention and how do they do it? *Nature Reviews Neuroscience* 5(6):495–501.

[117] Wolpert, D. H.; Macready, W. G.; et al. 1997. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation* 1(1):67–82.

[118] Wu, B.; Ooi, T. L.; and He, Z. J. 2004. Perceiving distance accurately by a directional process of integrating ground information. *Nature* 428(6978):73–77.

[119] Xu, C.; Li, S.; Liu, G.; Zhang, Y.; Miluzzo, E.; Chen, Y.-F.; Li, J.; and Firner, B. 2013. Crowd++: unsupervised speaker count with smartphones. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, 43–52. ACM.

[120] Yamada, T.; Hayamizu, Y.; Yamamoto, Y.; Yomogida, Y.; Izadi-Najafabadi, A.; Futaba, D. N.; and Hata, K. 2011. A stretchable carbon nanotube strain sensor for human-motion detection. *Nature nanotechnology* 6(5):296.

[121] Yan, T.; Kumar, V.; and Ganesan, D. 2010. CrowdSearch : Exploiting Crowds for Accurate Real-time Image Search on Mobile Phones. *Image Rochester NY* 77–90.

[122] Youseff, L.; Butrico, M.; and Da Silva, D. 2008. Toward a unified ontology of cloud computing. In *2008 Grid Computing Environments Workshop*, 1–10. IEEE.

[123] Zandbergen, P. A., and Barbeau, S. J. 2011. Positional accuracy of assisted gps data from high-sensitivity gps-enabled mobile phones. *Journal of Navigation* 64(03):381–399.

[124] Zheng, N.-n.; Liu, Z.-y.; Ren, P.-j.; Ma, Y.-q.; Chen, S.-t.; Yu, S.-y.; Xue, J.-r.; Chen, B.-d.; and Wang, F.-y. 2017. Hybrid-augmented intelligence: collaboration and cognition. *Frontiers of Information Technology & Electronic Engineering* 18(2):153–179.

[125] Zhou, X. S., and Huang, T. S. 2003. Relevance feedback in image retrieval: A comprehensive review. *Multimedia systems* 8(6):536–544.