NORTHERN ARIZONA UNIVERSITY

TEAM INTELLICHIRP

Requirements Specification

Authors
Steven ENRIQUEZ
Michael EWERS
Joshua KRUSE
Zhenyu LEI

Clients
Colin QUIN
Patrick BURNS
Mentor
Fabio SANTOS

Version: 2.00

December 10, 2019

CONTENTS

Requirements Specification

Steven Enriquez, Michael Ewers, Joshua Kruse and Zhenyu Lei

1 Introduction

It is becoming ever more important to track and manage the biodiversity that lives on Earth. More and more animals and plants are becoming extinct everyday which can create a major impact to other parts of the ecosystem. The group that our team is involved with is, Soundscapes2Landscapes, a science-based project looking to further advance biodiversity monitoring in order to save the lives of plant and animal species. Biodiversity is the study that ″refers to the variety of living organisms on Earth, how they relate to each other, their ecological function, and genetic diversity. All aspects of biodiversity are intimately linked to the functioning of ecosystems, where species interact with their physical environment. Biodiversity plays a vital role in many ecosystem functions, such as clean water, clean air, nutrient cycling, food production, and responses to disturbances, such as fires.″ [1] It is vitally important to conduct proper biodiversity monitoring, in order to understand the ever changing environments that humans share with plant and animal species.

At the moment, Soundscapes2Landscapes place low cost audio recording devices (AudioMoth) in various sites in Sonoma Country, California. Once placed, these devices record one minute of every ten minutes for three to five days at each site. This has so far resulted in a total of over 500,000 minutes of gathered audio data. Sonoma County is a rich environment full of many different species of birds, plants, and other forms of life. Soundscapes2Landscapes' (S2L) end goal is to expand their models for tracking biodiversity from Sononma County, California to the rest of the work. The audio recorded by these devices then go through a soundscape manual analysis process. The files are saved and analyzed manually by S2L researchers using the Arbimon II tool. This process consists of manually classifying audio components by drawing boxes around different sounds and labelling them by hand. The current process Soundscapes2Landscapes uses to measure biodiversity changes, is illustrated in the *Current Workflow* section of [Fig. 1].

Our clients Colin Quinn and Patrick Burns are part of the Global Earth Observation Dynamics of Ecosystems Lab (GEODE). Colin Quinn is a PhD student and Patrick Burns is a Research Associate at Northern Arizona University. They work with Soundscapes2Landscapes to help achieve their biodiversity monitoring goals. Our clients use a specific type of monitoring called, passive acoustic monitoring (PAM). This process allows more spatially extensive and continuous metrics for biodiversity. In Sonoma County, PAM has the ability to provide land managers and users with a better idea of animal species affected by development and conservation efforts. Our clients have assigned our team with the task of automatic sound identification from a soundscape, as currently sound identification is conducted in a manual time consuming way.

1.1 Problem

To understand the problems of the current implementation, the workflow process will be discussed. The sponsor's key business workflow is shown in Figure 1 below.

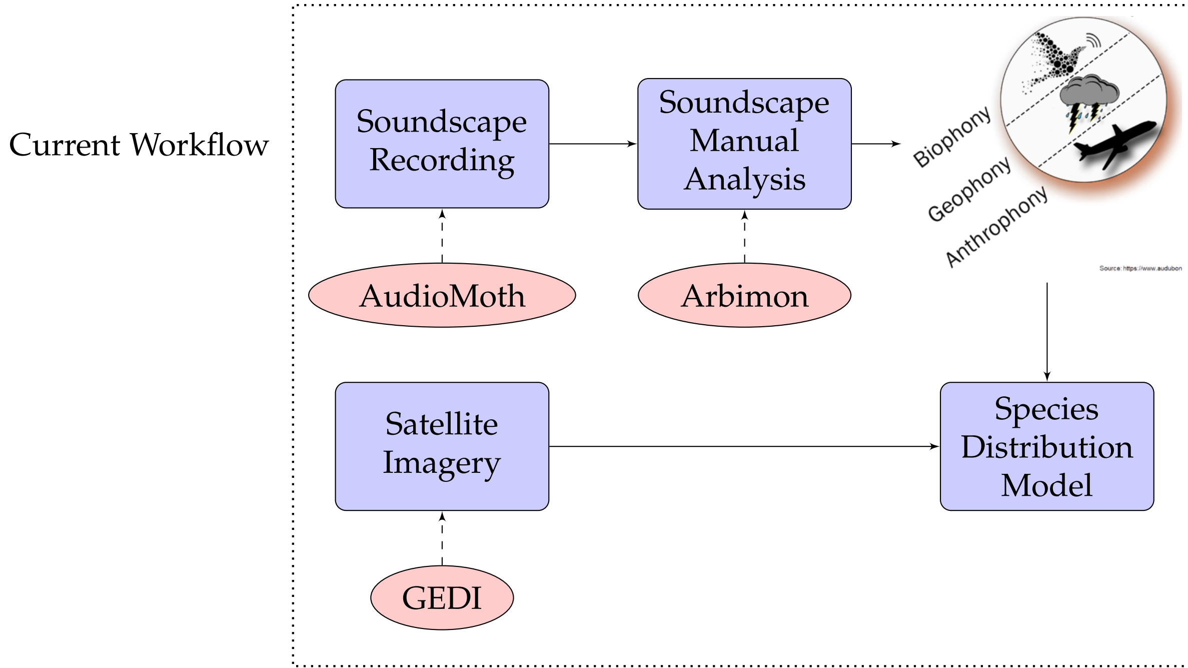


Fig. 1. Visualization of Current Workflow

First, soundscape recording data is collected from low cost audio recording devices that are placed in different landscapes across Sonoma County, California. These devices record 1 minute every 10 minutes, for 3 to 5 days at each site. The soundscape recording data then moves on to sound analysis where biodiversity can be identified along with the specific layers of biophony, geophony, and anthrophony. Once the identification and analysis are done, satellite imagery from the International Space Station is used to create visual representations of the surveyed sites. Finally, the satellite data and sound data are put together to create a species distribution model, which can be used to track locations of bird species and potential environmental changes in their ecosystems.

The part of the process we are involved with is the Soundscape Manual Analysis. The current identification tool, Arbimon, allows scientists to manually identify sounds present in a soundscape by drawing individual boxes around each identified sound component. This tool is useful but takes too long for scientists to effectively research the biodiversity in Sonoma County, California. Audio is recorded for 3 to 5 days per site, resulting in terabytes of data for each individual site. Additionally, our clients would like for volunteers, or citizen scientists, to be able to analyze their own files. Currently the identification tool does not allow volunteers to analyze files as the Soundscapes2Landscapes data needs to be kept private.

Overall, the problems include:

- The manual identification process is very time consuming. Terabytes of audio is collected from each site, and requires people to listen to the audio and manually draw boxes around the sound components being searched for.
- Current interface is not easily accessible to volunteers. Soundscapes2Landscapes wants this tool to be able to be used by anyone, and the current interface blocks out most volunteers.

1.2 Solution

Our solution to these problems will be an application called the Soundscape Noise Analysis Workbench. This solution involves a user-friendly user interface that hosts a machine learning model. The goal of this application will be to allow any user to upload their audio files for analysis, as shown in Figure 2 below.

Overall, our solution will consist of:

- User-friendly web application.
- A machine learning algorithm that automatically classifies different audio components that are being searched for in the inputted files.
- Visualizations of the analyzed audio. A standalone version of the application for offline use in the field.

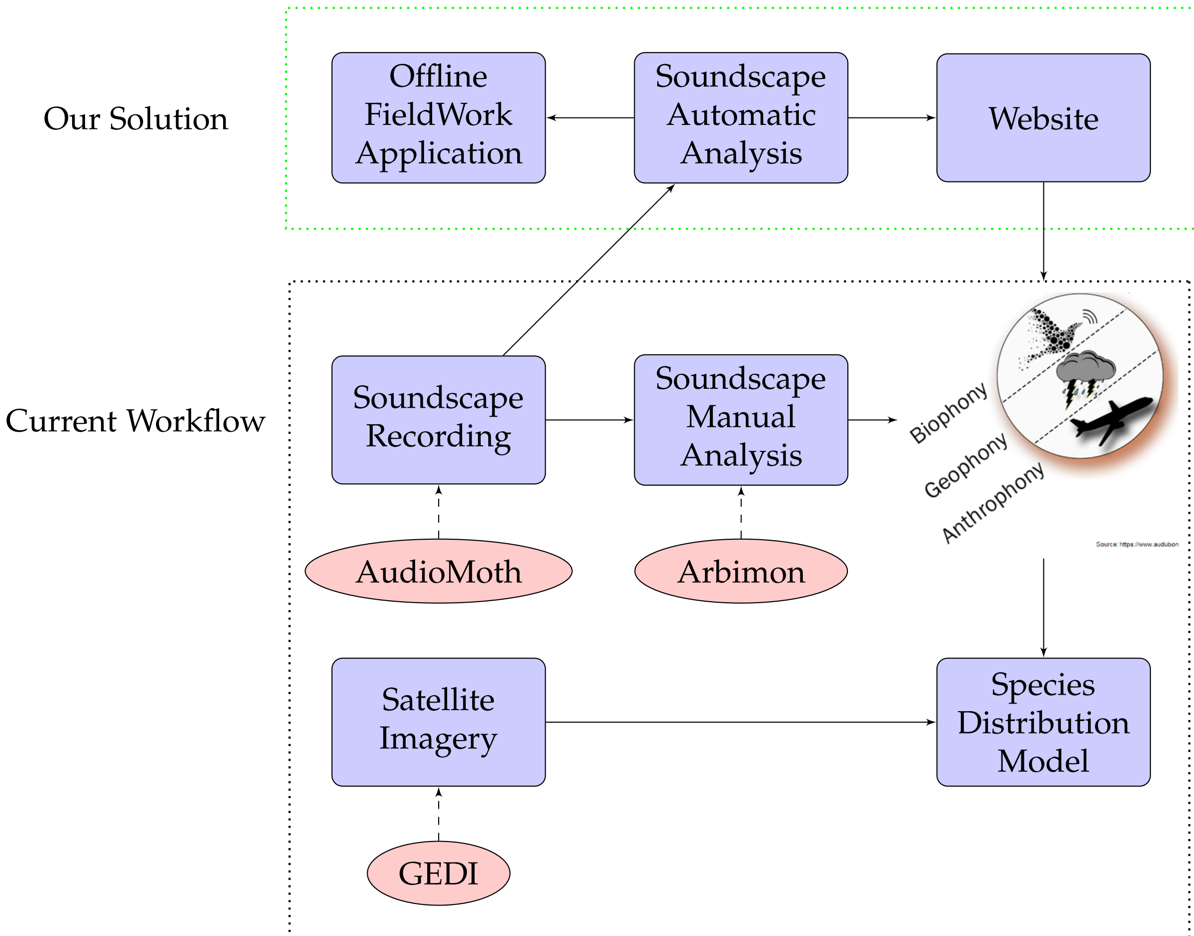


Fig. 2. Visualization of Current Workflow and the Added Solution

The solution will ingest audio files. Researchers from Soundscapes2Landscapes will use audio files that they collect with the low-cost audio devices being used around Sonoma County, California.

We will be using a machine learning algorithm to automatically classify different types of sounds in these recordings. The previous process of identifying audio components was through manual identification. Researchers would be manually listening to audio files and drawing boxes around various sounds. For a one minute clip, a researcher must listen to the clip, determine what the sounds are, draw boxes, and label each box with the corresponding audio component that is occuring. With noisy files, a researcher may spend over a minute going through a single file. Our machine learning model plans to accomplish this task in a fraction of the time, classifying an audio file in under a few seconds. The three broad categories of sound that we will be classifying are biophony, geophony, and anthrophony.

The machine learning model requires training to accurately classify audio components. We must train the model on previously classified audio data. This previously classified audio data consists of many audio files that have been labelled with each category of sound that we are looking for. The sound categories will include birds, cars, rain, wind, and others. Collecting data to train the machine learning model will come from multiple sources, including open source data and data from Soundscapes2Landscape's audio recorders.

The results of this classification will be visualized in a variety of ways. Two of the visualizations will be a labeled spectrogram, showing the classified components in the inputted audio, as well as a pie chart of the proportions of each sound category. These categories include geophony, biophony, and anthrophony. The solution will also be created as an application for offline use in the field. This application will provide users the ability to classify their audio without an internet connection.

2 Project Requirements

In order to acquire the domain level requirements for this project, our team has agreed that we will meet on a weekly basis with our clients, Patrick Burns and Colin Quinn. We have found that meeting on a frequent basis has provided crucial information to refine our requirements into even more specific lower level requirements. Through our initial meetings in the semester, we began with understanding our client's work process and their problems. Before our team began working on a solution to produce, we wanted to make sure that we completely understood our clients, and figured out exactly what the main problem has been for their work.

Once our team etched out the main problem our clients were experiencing, we then shifted our meetings focus. Our team and our clients began working towards discussing an envisioned solution along with the specific requirements towards that solution. The requirements had different iterations through each of our meetings, and finally were refined down as much as we could get them. Refining the product's requirements was an important process as we want to provide an exact solution to our clients. The solution and requirements are coupled tightly and we want to follow these specific guidelines that our clients agreed on to create the right product.

In the past 3 months of our first semester, we have understood the problem, envisioned a solution to produce, and defined the proper domain level requirements to follow. Our team has determined that this project contains the following key domain level requirements:

1) Users will be able to upload audio files.
2) Users will be able to analyze all uploaded audio files.
3) Users will be able to see the results of the analysis visualized.
4) Users will be able to receive the results of the analysis in a timely manner.
5) Users will be able to export all results.

The above domain level requirements are generally more broad and certainly contain more layers to them, such as their functionality and performance levels. Once our team refined our domain level requirements down, we then continued the discussion with our clients to ensure that we provide a fast performing and specific functioning solution. The following three sections, **Functional Requirements**, **Non-Functional Requirements**, and **Environmental Constraints** will provide the lower and more specific layers that are present within each of the above domain level requirements.

2.1 Functional Requirements

The first requirements provide a specific baseline for the functionality of our product, and are known as the functional requirements. The functional requirements give detail on the specific features that the product will provide to our users when accessing the application, along with specifically how the features will work within the application. The product that our clients and our team have envisioned will have the following functionalities to perform properly.

- **F.R.1:** The application will be able to ingest audio files.
- **F.R.2:** The application will use a machine learning algorithm to classify sound components in an uploaded audio file.
- **F.R.3:** The application will calculate acoustic indices an uploaded audio file or files.
- **F.R.4:** The application will display the results of the analysis in multiple ways.
- **F.R.5:** The application will be able to export the results of the analysis.
- **F.R.6:** The application will be able to extend to an offline version for fieldwork.

Once we defined the above functional requirements, we then wanted to go deeper into these requirements to truly define what the application will be doing at each step of its execution. The following sections are committed to the deeper definition of each of the above functional requirements:

2.1.1 F.R.1: The application will be able to ingest audio files:

The application depends on audio files in order to produce results, and therefore is a crucial functionality towards our application.

(1) The application will be able to ingest audio files.
  (1) The application will require audio files specifically in the .WAV format in order to function properly.
    (1) In the event that a user will try to upload .MP4 files, the application will be able to convert these files into a .WAV format.
    (2) In the event that a user will try to upload .MP3 files, the application will be able to convert these files into a .WAV format.
  (2) The application will use a file explorer to search for the user's files.
  (3) The application will be able to use a "Drag and Drop" functionality to upload user files, for ease of use.
  (4) The application will be able to ingest a single audio file.
  (5) The application will be able to ingest multiple audio files.
    (1) The application must be able to upload the selected file to an accessible server.
      (1) The application will then be able to continue its execution.

2.1.2 *F.R.2: The application will use a Machine Learning Model to classify sound components in an audio file.*

Classification of sound components in a specific audio file is the core of the entire project, thus it is also a very crucial functionality for our product.

(1) The application must be able to access a specific audio file that has been uploaded.
  (1) The application must be able to segment the file into 1 second audio clips.
  (2) Once the file is successfully accessed, the application must be able to make Flask API calls to access the machine learning model.
    (1) Upon making the proper Flask API calls, the machine learning model will be run against the provided audio clips.
      (1) The application will provide a set of predictions of specific sound components that may be present from the machine learning analysis.
        (1) The application will be able to identify the sound of Birds from an audio file.
        (2) The application will be able to identify the sound of Crickets from an audio file.
        (3) The application will be able to identify the sound of Bugs from an audio file.
        (4) The application will be able to identify the sound of Frogs from an audio file.
        (5) The application will be able to identify the sound of Wind from an audio file.
        (6) The application will be able to identify the sound of Thunder from an audio file.
        (7) The application will be able to identify the sound of Fire from an audio file.
        (8) The application will be able to identify the sound of Rain from an audio file.
        (9) The application will be able to identify the sound of Cows from an audio file.
        (10) The application will be able to identify the sound of a Chicken from an audio file.
        (11) The application will be able to identify the sound of Cars from an audio file.
        (12) The application will be able to identify the sound of Semi Trucks from an audio file.
        (13) The application will be able to identify the sound of Trains from an audio file.
        (14) The application will be able to identify the sound of Walking from an audio file.
        (15) The application will be able to identify the sound of Sirens from an audio file.
        (16) The application will be able to identify the sound of Construction from an audio file.
        (17) The machine learning algorithm will be able to search for sound components within the category of Biophony.

- (1) The application will be able to create predictions for Biophony by using the results from the individual sound identification for Birds, Bugs, Frogs.
  - (1) The application will be able to predict the amount of Biophony in a file.

(18) The machine learning algorithm will be able to search for sound components within the category of Geophony.

- (1) The application will be able to create predictions for Biophony by using the results from the individual sound identification for Wind, Thunder, Fire, Rain.
  - (1) The application will be able to predict the amount of Geophony in a file.

(19) The machine learning algorithm will be able to search for sound components within the category of Anthrophony.

- (1) The application will be able to create predictions for Biophony by using the results from the individual sound identification for Wind, Thunder, Fire, Rain.
  - (1) The application will be able to predict the amount of Anthrophony in a file. Once predictions are completed, the application will be able to continue its execution.

2.1.3 ***F.R.3: The application will calculate acoustic indices in an uploaded audio file or files.***

Classification of sound components in a specific audio file is the core of the entire project, thus it is also a very crucial functionality for our product.

(1) The application must be able to access a specific audio file that has been uploaded.
- (1) Once the file is successfully accessed, the application must be able to make Flask API calls to access the algorithms that calculate acoustic indices.
  - (1) Once predictions are completed, the application will be able to continue its execution.

2.1.4 ***F.R.4: The application will display the results of the analysis in multiple ways***

The ability to see the results of the analysis is a very important aspect of the application. Without visualizations the application would not be useful to our clients and their users.

(1) The application must be able to access the results provided by the machine learning model.

(2) The application must be able to access the results provided by the machine learning model.
- (1) The application will then be able to create multiple styles of visualizations through different Python libraries.
  - (1) The visualization must be able to display a spectrogram of the audio file.
    - (1) The spectrogram must be able to show the audio file chunked into 1 second clips.
    - (2) The spectrogram must be able to show the chunks in which individually labeled sound components are found in the audio file.

        (1) If a sound component is found in more than one concurrent chunk, the spectrogram must show that the sound overlaps chunks.
        (2) The spectrogram must be able to show the chunks in which Biophony is located.
        (3) The spectrogram must be able to show the chunks in which Geophony is located.
        (4) The spectrogram must be able to show the chunks in which Anthrophony is located.
      (3) The visualization must be able to display a pie chart showing the percentage of Biophony present in the audio file.
      (4) The visualization must be able to display a pie chart showing the percentage of Geophony present in the audio file.
      (5) The visualization must be able to display a pie chart showing the percentage of Anthrophony present in the audio file.
      (6) The visualization must be able to display a list of all the sound components found in a file.
        (1) The list of sound components must be able to display the total number of seconds each sound component was present in the audio file.
      (7) The visualization must be able to show a table showing the results of each acoustic indices calculation.
        (1) The table of results must be able to show the results of machine learning classification for comparison.
      (8) If multiple files are analyzed, the application must be able to wait for the users input to produce results.
        (1) Once the display of the results is complete, the application will then continue its functionality and execution.

2.1.5 ***F.R.5: The application will be able to export the results of the analysis.***

(1) The application will be able to export the results of the analysis.
  (1) The application will be able to start an export data process when a user provides input.
    (1) The application will be able to create a .CSV file.
      (1) The application will be able to add data results to the .CSV file.
        (1) The application will be able to add the results of the individual sound component classification task.
        (2) The application will be able to add the results of the Geophony found in a soundscape file.
        (3) The application will be able to add the results of the Biophony found in a soundscape file.
        (4) The application will be able to add the results of the Anthrophony found in a soundscape file.
        (5) The application will be able to add the results of the Acoustic Indices algorithms.
        (6) The application will be able to post the .CSV file onto the website.
          (1) The application will be able to allow the user to download the completed .CSV file.

2.1.6 F.R.6: The application will be able to extend to an offline version for fieldwork

(1) The application will be available in an offline fieldwork format.
  (1) The offline application will produce a batch job script when files are analyzed by the user.
    (1) The resulting batch job script will be available for use with any HPC cluster.
  (2) The offline application will ingest audio files.
    (1) The offline application will specifically require audio files in the .WAV format.
    (2) The offline application will use a standard file explorer to search the user's files.
    (3) The application will be able to use a "Drag and Drop" functionality to upload user files, for ease of use.
    (4) The application will be able to ingest a single audio file.
    (5) The application will be able to ingest multiple audio files.
    (6) The application will be able to point to a single folder containing audio files.
      (1) The application will run the entirety of *F.R.2*.
      (2) The application will run the entirety of *F.R.3*.

2.2 Non-Functional Requirements

The above functional requirements are geared towards what the application will do and specifically how it will function. The Non-Functional requirements are all about how the application will perform, in terms of execution speed. Through meetings with our clients, we have defined the following two non-functional requirements that our application should meet:

- **N-F.R.1:** The application upload of 1 minute audio files should take no longer than 5 minutes.
- **N-F.R.2:** The application should take less than 3 Seconds to complete a full analysis of a 1 minute audio file.
- **N-F.R.3:** The Machine Learning model should make predictions of at least 65% accuracy.

Once we defined the above non-functional requirements, we then wanted to define how the application will be running and it performance at each step of its execution. The following sections are committed to the definition of how the applications performance will run off of the above non-functional requirements:

2.2.1 *N-F.R.1:* The application upload of 1 minute audio files should take no longer than 5 minutes

(1) The application must be able to open a standard file explorer on the user's computer within under 1 second.
(2) The application must be able to receive the information of a "Drag and Drop" file within under 1 second.
  (1) Once a file explorer is open and the desired file is selected, the application must be able to start the uploading process upon clicking âopenâ on the file explorer. The upload should initiate with no delay.

(1) The uploading process of a 1 minute audio file should take no longer than 5 minutes total.

2.2.2 ***N-F.R.2:*** *The application should take less than 3 Seconds to complete a full analysis of a 1 minute audio file.*

(1) Once the uploading process has been completed, the analysis process should immediately take place after the user has clicked the âAnalyzeâ button.
   (1) Once the button fires, the application will make a call to the Flask API which will contain the machine learning model. The Flask API call should take no longer than 1 second.
      (1) Once called, the machine learning model will begin analyzing the selected audio file immediately.
         (1) The machine learning model will be able to produce results for a 1 minute audio file within 3 seconds.

2.2.3 ***N-F.R.3:*** *The Machine Learning model should make predictions of at least 65% accuracy.*

(1) While the machine learning model is in the process of identifying specific sounds within an audio file, the model should meet an accuracy of 65
   (1) The model must be able to provide consistency of accuracy when identifying evident sound events within an audio file.

The three above non-functional requirements mainly show concern for the speed at which the application will complete a full analysis, along with the accuracy of the machine learning model associated with it. Other than the true performance of the machine learning model or uploading speed, our clients have not specified any other performance requirements that we must meet.

2.3 Environmental Constraints

The Environmental constraints are slightly different from the above Functional and Non-Functional requirements, but can still be seen as a requirement for the product. The environmental constraints may define a specific set of tools which the product must be produced with. Through our meetings, our team has only defined the following environmental constraint:

- **E.C.1:** Machine Learning Algorithm must be accurate on data from Sonoma County, CA.

Our clients work currently only consists of working on data that comes specifically from Sonoma County in California. Through this constraint, we have determined that we cannot guarantee that our machine learning model will be able to provide a consistent accuracy when the model is used with data from different regions. Therefore, we have decided to deem our only environmental constraint to be the data that our client is providing us with, being specifically from Sonoma County, CA.

3 POTENTIAL RISKS

When developing products for clients, the developers must always consider the major risks that can occur when developing and releasing the product. During our requirements acquisition and technological feasibility studies, we have identified two risks to this project's success and overall impact. These were documented as we went through both processes and conducted general research including reviewing Software Engineering. It is important to outline the risks and methods in order to overcome them, because they can ultimately lead to the project not functioning enough or completely failing. Our risks are high-level or medium-level, which and can encompass many different, low-level issues.

Risk 1: Inaccuracies — Likelihood **HIGH** — Severity **HIGH**

The first and biggest risk is accuracy. This is likely to happen, because obtaining sufficient high-quality training data is our most time-consuming task. We spend a lot of time looking for useful data to train the model separately, but high-quality data with accurate labels is not easy to find. The severity is also high, because if our application is not accurate enough, then it cannot functionally give correct results. To this end, our mitigation plan is to use another model other than neural networks, which has shown to have a higher accuracy for smaller training data sets, or we must use other ways to collect more data, such as using data augmentation. Now that we know that a group used neural networks to obtain 65% accuracy on the open source dataset we are using now, we will find a better model for higher accuracy.

Risk 2: Privacy — Likelihood **MEDIUM** — Severity **LOW**

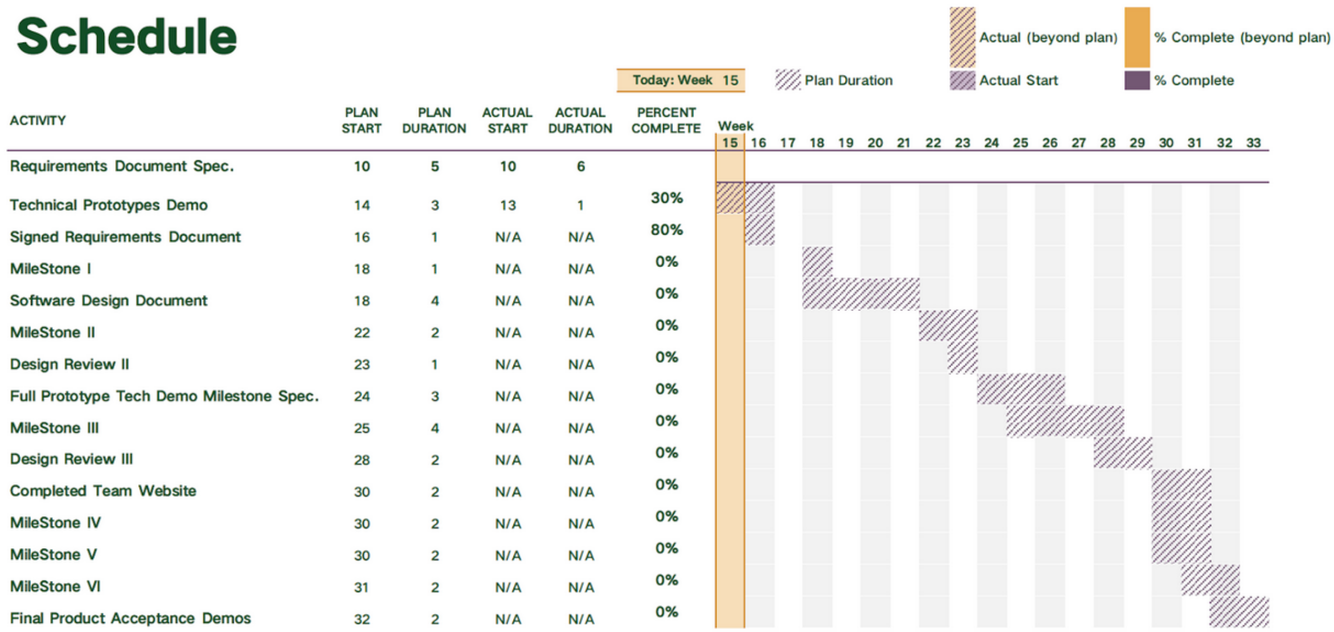
The second risk is privacy. We think that since we cannot get enough training data, we can manually check the data uploaded by the user and add it to the training data, which leads to the possibility that it is medium, because we may store the user's data in our Database for future training and validation. This may cause some legal issues, but we have a mitigation plan that allow users opt-in to submit their data, so the severity is low.

4 PROJECT PLAN

The project plan is an essential part of every software project as its primary purpose is to notice when the project is veering off-course as early on as possible, which allows for ample time to readjust the project and the final product.

Now that we are at the end of the semester, we have completed each task in advance or on time. And we have started tech demo ahead of plan right now. During each mentor meeting, we are showing our progress towards a finalized tech demo. We are also currently implementing multiple different machine learning algorithms to determine which will give our solution the highest accuracy.

To develop our product, we plan to implement at least a core feature every two weeks for the website. We will show our clients our app each time we update it. This will allow the client ample time to review the website and request any changes and report any bugs during development. Below, in Figure 3, we outline our schedule. The orange line shows where we currently are. We will have a total of six milestones up until the end of next April, which are six major functional requirements. When April is over, we should have all major features of the app completed and the product will enter into an actual environment as much as possible for the client to help determine the existing flaws with the product. We will also have testing to do for getting the product fully ready to be adopted by the client. At the end of the final Acceptance Testing stage, the product is delivered to the client and they will determine if they will accept the solution or not.



ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Requirements Document Spec.	10	5	10	6	
Technical Prototypes Demo	14	3	13	1	30%
Signed Requirements Document	16	1	N/A	N/A	80%
MileStone I	18	1	N/A	N/A	0%
Software Design Document	18	4	N/A	N/A	0%
MileStone II	22	2	N/A	N/A	0%
Design Review II	23	1	N/A	N/A	0%
Full Prototype Tech Demo Milestone Spec.	24	3	N/A	N/A	0%
MileStone III	25	4	N/A	N/A	0%
Design Review III	28	2	N/A	N/A	0%
Completed Team Website	30	2	N/A	N/A	0%
MileStone IV	30	2	N/A	N/A	0%
MileStone V	30	2	N/A	N/A	0%
MileStone VI	31	2	N/A	N/A	0%
Final Product Acceptance Demos	32	2	N/A	N/A	0%

Fig. 3. Visualization of IntelliChirp's Current Schedule

Next semester, we'll have several tasks to complete, and based on our performance of this semester, we're confident with those tasks and this project.

5 Conclusion

The impact of human involvement on ecosystems has major consequences. More than one million plant and animal sciences are going extinct, with the majority happening within the last few decades (IPBES). There is an ever growing need to properly monitor the biodiversity in ecosystems, as well as the factors that impact biodiversity. Our clients Colin Quinn and Patrick Burns work with the science-based group Soundscapes2Landscapes to provide a more effective and efficient way to monitor biodiversity. They have tasked our team with building an automatic way to identify specific, individual sounds present in a soundscape recording.

The problem that we are trying to solve is the time-consuming manual identification process of audio components in recordings from various sites in Sonoma County, California. Our solution is to develop a user-friendly web application that hosts a machine learning model to automatically classify these audio components. This will allow volunteers as well as researchers to efficiently classify the components of their recorded soundscape files. As a stretch goal, we will also be implementing an offline fieldwork application for use on a laptop or Raspberry Pi inorder for researchers in the field to also use our application. In order to get one step closer in making this solution a reality, we have put together this technological feasibility document.

This document's goal is to outline all the requirements our system will need in order to properly work with all of the complex components needed, and to effectively classify sounds present in a soundscape file. Our team has outlined the requirements needed to successfully complete: the ability to upload audio files, the ability to analyze audio files, to ability to see visualized results of the analysis, the ability to receive these results in a timely manner, and the ability to export the results. With all of these requirements complete we believe our application will deliver an effective solution to the problem. Based on our teams progress through our Technology Demo and Technological Feasibility documents, we believe that a successful solution is possible, and will provide researchers with the ability to more effectively manage the biodiversity in ecosystems.

6 Glossary

Acoustic Indices - Statistics that measure the distribution of acoustic energy and information in a soundscape recording.

Anthrophony - Describes the collective sound that humans create in a soundscape recording (i.e. Car engines, Talking, Construction noise).

Biodiversity - The variety of living organisms on Earth, how they relate to each other, their ecological function, and genetic diversity.

Biophony - Describes the collective sound that animals create in a given environment (i.e. Birds, Crickets, Amphibians).

Geophony - Describes the collective sound that naturally occurring non-biological sounds create in a soundscape recording (i.e. Wind, Rain, River).

Data Augmentation - A strategy for increasing diversity in a data set without gathering new data.

Machine Learning - A mathematical method that automates data analysis by using a computer system that learns based off of patterns in data.

References

[1] ”Soundscapes to Landscapes,” *Soundscapes to Landscapes*. [Online]. Available: https://soundscapes2landscapes.org/. [Accessed: 02-Dec-2019].