Software Engineering for CSAI

Fall semester, 2022/2023

- Group Prolog -

Aryan Verma
Benjamin Tomas Matthews
Dominic Armand Stamatoiu
Mathijs Sebregts
Ossama Al Hassan
Stefan Tei Năstasie

Problem Definition

Agriculture has been humanity's primary source of food supply for the past 10 millennia, and that is unlikely to change within the foreseeable future. In order to facilitate the growth of farms, which have become entire ecosystems, we must take into account even the smallest of details, which can be usually found in nature. Our team wants to provide a tool that allows farmers to identify and track the different species of birds and insects surrounding and inhabiting their farms.

Maintaining a balanced ecosystem is important for high production rates and because animal pollination is one of the most important elements for modern-day agriculture, the lack of a balanced ecosystem can decrease crop production by up to a staggering 71%, as seen in the study by Bartomeus et al.

According to Olimpi, et al.(2022), "farmland birds can suppress insect pests, but may also consume beneficial insects, damage crops and potentially carry foodborne pathogens", so a clear distinction between bird species will be of use. Moreover, a problem that is present nowadays in farming is the overuse of pesticides. The insects that are threatened by the use of pesticides also include the ones that are beneficial to the crops, mainly pollinator insects (Raven & Wagner, 2021).

Problem Breakdown:

- Many farms are not optimized for the potential yield they could offer or food waste is not prevented efficiently
- Farmers have little to no understanding of insect and bird populations that are part of their farm ecosystems
- Farmers cannot adapt their farming techniques and tools efficiently
- The ecosystems involving farms and their surroundings are disrupted

Idea

Our team is planning to build an intelligent system using machine learning that differentiates and counts the population of different insects and bird populations by using audio and image data as input. This solution is primarily aimed at helping farmers, but would also be available to the general population. The system will provide the user with the choice of uploading either a sound recording of a bird or an image of an insect (a beetle to be specific). After that, the system proceeds to first detect whether a recording or picture is valid, meaning that there is in fact a bird/insect present, and either proceeds further or warns the user that there hasn't been any positive detection. If the detection phase has ended successfully, the system then proceeds to classify the given species and display the output result. We are planning to build an easy-to-use intelligent system, with a simple interface, using deep learning that is able to perform multiple operations in accordance with a previously specified hierarchy. During the final phase of development, an A/B testing method will be used to test multiple versions of the system and assess its performance, while measuring the user reaction to the narrative of a customer-based product.

Objectives

For the project to be deemed successful it needs to detect and classify both insect and bird populations, given video/image and sound recordings. The effectiveness of the tool will not only be determined by how well the AI model has been trained and how accurately it can predict, but also by how well it can perform the initial detection phase. Although the main objective is to allow the user to identify insect and bird populations accurately, timely responses and accessibility matter too, meaning that predictions have to be made fairly fast and the interface should be made easily interpretable to keep the user engaged and encouraged to come back and reuse the software.

- Organisational objective - With the help of this tool, farmers will be able to adapt their agricultural approaches and thus, increasing their productivity while decreasing food waste.
- Leading objective - Farmers should gain insights and knowledge repeatedly whenever they feel the need to.
- User-oriented objectives - The tool should deliver results of the predictions in a timely manner while being easily interpretable
- Component Objectives - Every element must be connected to the system, while at the same time functioning as intended without any errors or bugs

Environment/software

When setting up the environment for our AI models, Anaconda for Windows and JupyterLab were chosen because of the easy access to GPU and the familiarity we all have when working with Jupyter notebooks. At first, a GPU machine was used to train the models because there was one available in our group (Nvidia RTX 3080). Later we realized that training on a less powerful machine was possible. This was preferred because of the fact that models could then be trained in parallel.

Data

1. **Bird sound classification**

a. **Volume**: The dataset that was used in order to train the model had to be pre-processed. It came as a compressed zip file of 12.1 GB. This zip file contained 4131 ‘flac’ files, each containing a sound recording of a bird.

b. **Velocity**: The 4131 ‘flac’ files were processed by the feature extraction unit in just over 6 minutes, at a speed of roughly 13 files a second. The files were individually loaded and processed by the unit.

c. **Variety**: The dataset consists of 4131 sound recordings and a metadata file that contains the labels and tags necessary for interpreting the audio files.

d. **Veracity**: The metafile for the bird's sound dataset contains ~500 entries that have no match in the sound file folder. This means that we are left with 4131 files out of 4,619 entries.

2. **Insect image classification**

a. **Volume**: The data that was collected in order to be processed and used in training the model has the size of 2.5 GB. It was firstly compressed into a zip file of 2.5 GB, but when extracted, its size remained the same. The data is stored in 63,364 files, each file containing a picture of an insect.

b. **Velocity**: Downloading and extracting the data from the zip file took about 45 minutes. Moreover, the data was loaded using batches of size 32, resized, and split into training and validation. The whole process took about 5 minutes.

c. **Variety**: The data is made up of only files of type jpg. There are 291 different species of insects.

d. **Veracity**: The data was collected by a researcher who took image crops of specimens from beetle drawers. Whether the data is reliable or not is hard to determine because of the lack of metadata that describes the pictures. It does however seem to be consistent, meaning that all pictures within the same folder seem to contain the same species of beetle.

System

For the first attempt at this classification task, we decided to create our own models, just to experiment with the data. For the bird classification assignment using sound data, we used a dense neural network (DNN) with 3 hidden layers and 256, 512, and 256 nodes for each layer with a total of 404,992 trainable parameters. The training process was executed using 256*i epochs (where 'i' ranged between 1 and 5) and a batch size of 100. The result of the training process was that the model was overfitting by a large margin. The accuracy on the training set varied around 95 and 98.5% while the accuracy on the test and validation sets was stuck between 20 and 30%. This issue will be addressed by either changing the model entirely or fine-tuning the parameters as well as by experimenting with different ways of data extraction.

Software Requirements Specification

1. Introduction

This document is intended to provide readers with a detailed overview of the requirements that are connected to BUzzCrowTulip's project for the development of a tool to help users detect and classify birds and insects. The requirements are outlined in three sections: User requirements, System requirements, and AI-specific requirements, with a brief explanation of what these requirements entail prior to each respective section.

1.1 Purpose

The purpose of the system is to monitor the balance between different types of insects and birds. In order to optimize crop growth, as well as to maintain a healthy ecosystem, the right balance between different types of insects and birds should be maintained. Before executing an action for the maintenance of the balance, a clear overview of biodiversity would be required. In case there is no clear overview, there lies the increased likelihood of the occurrence of problems such as the overuse of pesticides, damage from infestations, etc. As our software can accurately detect and categorize different insect and bird species, it would allow for an insight into the populations; Thereby aiding the user in achieving the aforementioned clear overview.

1.2 Intended audience

The software requirements document created for this project is intended for farmers, our primary group, along with several secondary groups including developers, project managers, and testers that work for BUzzCrowTulip. Its goal is to help readers understand how the software works as well as what requirements need to

be met in order to identify and track the different species of birds and insects that are captured and uploaded into the software.

1.3 Additional information

This document is subject to change based on future adaptations made to the software as well as emerging technical innovations that would alter the requirements that need to be met.

2. User requirements

- **A.** User start screen shall have the option to upload insect or bird data in common image and audio formats
- **B.** The user screen may provide the current status of the system (is it processing the data, or does it bump into an error)
- **C.** The system must be accessible through a web browser (Safari, Chrome, Firefox)
- **D.** The picture/audio file upload must happen in no more than 10 seconds (given a stable internet connection)
- **E.** The certainty score must be displayed within 15 seconds requiring solid connection speeds to the database and model
- **F.** The User has the option to remove their data from the software application
- **G.** The User has the option to enable and disable the system's access to recording devices
- **H.** User consent must be given in the case that the user wishes to submit data for model improvement.

3. System requirements

3.1 Functional requirements

- **A.** The system should be able to distinguish whether the sound detected is a bird or something other than a bird.
- **B.** System should be able to distinguish whether the image that is being analyzed contains an insect or not.
- **C.** Algorithm should be able to determine what bird species can be heard in the recorded sound.
- **D.** Algorithm should be able to determine which insects are seen in the image.
- **E.** User screens should load in no longer than 2 seconds using efficient cloud computing
- **F.** The system can display the results of the classification in natural language to the user.
- **G.** The system can display confidence percentages for each classification

3.2 Non-functional requirements

These non-functional requirements are listed in order respective to the functional requirements. This means that the non-functional requirements from 1 correspond to the functional requirements of 1 in the section above.

A. A microphone and classification algorithm in Python is required as audio recordings could be compromised by environmental noise. For bird data, only audio recordings in FLAC or WAV will be accepted.

B. A video recorder and classification algorithm in python are required. An external factor that could cause problems with classification could be when two insects are recorded at the same time. For insect data, only image files in JPG will be accepted.

C. A classification algorithm is required.

D. Same as for 3, a classification algorithm is required.

E. An interface that enables the user to classify a bird or insect. The interface should also give clear feedback about the classification.

F. The system can compute how likely each classification is to be true (how confident the system is in the classification)

4. AI-specific requirements

A. The system should be able to achieve more than 80% accuracy as it is built using a convolutional neural network which is not only trained but also optionally updated using the user's recordings and feedback.

B. As the access to, and usage of, an AI model can be relatively daunting, a good GUI is needed to guide the user through this process

C. The user should be updated often as to the internal state of the system for similar reasons as mentioned in the previous point.

D. The designed system is meant to be highly accurate at classification. As such the AI model should generally outperform average human classification attempts.

E. As with other parts of the system, the AI model will require an internet connection to be accessed, used, and updated.

AGILE methodology and version control

1. Progress & Updates (task 4)

A. Overall progress - We continued working on the project tasks and focused on improving the models. Also, we made sure to define and follow up on all the requirements needed.

B. Phone application - We tried to create an Android app that could use the models. We used Android Studio and made a basic app by using the Kotlin programming language. We soon realized that this approach is not feasible due to the lack of our team’s experience in developing phone apps and the lack of time.

C. Web application - We worked on building a simple web application that can request uploads for both images and sound files. We believe that this approach is more appropriate for our project because it better aligns with the experience we have as a team and the ease of use it offers.

D. Github repository - We created a GitHub repository (https://github.com/GroupProlog/Bird-Insect_project). We defined three branches. One main branch and one for both of the models, the bird model, and the insect model. Two of the three

branches focus only on the bird model or the insect model, while the third branch puts both models together and adds the web app, front end information. The additional two branches were created for the sake of clarity and for being able to experiment on the models separately.

2. SCRUM masters and AGILE methodology

During the two weeks, Stefan and Ben were the scrum masters. We tried to follow the AGILE methodology for our project, however, due to unforeseen circumstances, our meetings were canceled a few times during the last two weeks. We did however work individually without setting up sprints and made significant progress. We then started working towards organizing sprints and dedicating more attention to our Trello board (https://trello.com/b/ixaaCEVx/task-management).

5. Testing

Unit tests

Bird Model:
For the bird model, the following unit tests were conducted by using pytest and creating a separate test.py file. The following unit tests were included in the pytest file:

- Does the unit, that generates a species list from the metadata and saves it as a .txt file in the correct format, work?
  - This test was selected, as creation of a unique species list for the dataset we are using is necessary for more accurate results from the pretrained model.
  - The input was a sample of a newly created list, which was then tested to see if it was formatted correctly. This test was passed.

- Does our data processing codeblock extract the right amount of features from the sound files?
  - This test was selected, as the model training breaks if an incorrect amount of features is extracted from a file. This would lead to incorrect input into the neural network.
  - The input into the test is newly processed data. The test then checks if the correct amount of features are extracted. This test was passed.

- Does our clustering code remove the correct amount of data from the arrays?
  - This test is important, as we quickly realized that some labels had too few files associated with them. As such the model would be extremely inaccurate.
  - The trimming function removes files and labels from the model depending on how many other files share the same label. This was tested by seeing if the correct amount of files are trimmed. This test was passed.

Similarly to the tests implemented above, the unit tests below were conducted by using pytest and creating a separate test.py file.

- Does the detection phase work as intended?
  - We needed to check whether the detection algorithms generated a binary output because we wanted to avoid the system classifying non-insects or non-birds as one of the two. We checked this by testing the

algorithm using the initial datasets against separate datasets that excluded bird sound recordings and beetle images. The test was passed, with an accuracy of ~85%.

- Are the models saved properly and are the ‘h5’ files functional when hosted?
  - Since we had to save 4 models, it was crucial that every single one of them was loaded and saved properly so it can be hosted on the cloud service.
  - We checked if the same input would generate the same output on both versions (loaded and local) for each model. The test was passed.

Integration tests

- Is the file upload is successful
  - We needed to test if the file uploaded by the user was registered and saved by the system. We ran a simple test that would output the file uploaded. The test was passed.
- The system processes the upload (sound files) as intended
  - After the upload was successful, we needed to check if the sound files were being preprocessed as intended. We checked this by running a local version of the preprocessing stage and comparing it to the output generated by the system. The test was passed.
- Does the bird classification algorithm act as a prerequisite for the classification algorithm?
  - This test was important because it was our only way of testing the connection between the detection and classification algorithms.
  - The models were tested as a whole to see if the classification algorithms would produce any output even if the detection algorithm failed. The test was passed.

6. Scaling

Instances

- Our models for both insect and bird detection, and our model for insect classification, are being hosted on AWS. To do this we are using a default 4 elastic computing (ec2) instances. 2 Instances that will host the bird related models, and 2 instances that will host the insect models.
- These instances are set up to receive http requests over port 8080, which is important for the load balancer as this will allow it to communicate with them.
- The instances are all set up on the same availability zone, us-east-1a
- Importantly, the instances share a key pair, making accessing them individually for configuration easy.

Load balancer

- We are going to be using an application load balancer with 2 listeners for the project. The listeners are set up on different ports, and will direct traffic to either Instances in the Bird model target group or insect model target group.

- As an added measure, the listeners will default to an instance that is loaded with both models and a different python script that can handle either input. As such, in case there is an error with the Listeners rules which doesn't allow forwarding to the correct target group, the request should still be handled properly.

Scaling policies

- Although the instances default to 2 of each model, the auto-scaling policies allow for up to 5 identical instances of each model to be running at any given time. Furthermore, at least 2 instances for each model will be running at all times.
- The scaling policies will automatically scale up and down between 2 and 5 instances of each type based on average instance CPU usage, with a 5 minute time limit on usage over 50% leading to the creation of more instances and 5 minutes under leading to the termination of instances.
- The policies will aim to have 3 of each instance type running if possible.

Teamwork description

At the start of the project, we had assigned roles to everyone on the team but also agreed that these roles would be flexible and interchangeable. Aryan and Mathijs' roles were to write the reports but soon transitioned to writing code, organizing the GitHub repository, and doing research. Osama and Benjamin were the main people responsible for pre-processing the datasets, creating the AI models, and other coding tasks. They stuck to their roles and provided the groundwork for the back-end side of the project. Dominic was responsible for doing research and helping out with reports, but later also contributed to the front-end side of the project, Stefan was responsible for organizing the meetings and scrums, keeping track of progress, keeping the team working together, and helping with whatever needed to be done.

From the beginning, we understood that this project required constant participation, work, and perseverance, which is why Stefan was in charge of keeping the team together by sending texts, calling team members to bring them up to speed if they missed a meeting, assigning tasks to team members, and everything else required for completing every step of the project on time. At the same time, everyone was successful in completing their assigned tasks which kept the team motivated and ensured a good working atmosphere and team chemistry. Finally, using the AGILE methodology and setting up SCRUMS proved to be effective tools for our team and helped us repeatedly throught the project.

The most recurrent issue we have encountered was that of lack of experience. Most of us only had experience with working on simple AI models and nothing else. Starting from scratch with front-end development, deployment on cloud services, connecting and sharing all our code was a challenge for every member of the team, but it was definitely a learning experience that we all enjoyed. There were other setbacks, such as poor or lack of communication between team members, scheduling problems that prevented us from meeting at the same time consistently, and personal issues that made us take some time off the project which meant some people had to do more work during some periods. Nonetheless, we believe that we overcame most of the challenges fairly quickly and in a proper manner by communicating, helping each other in times of need, and making an extra effort to learn and apply concepts, tools, and frameworks needed for the project and the success of the team.

We believe that this project served as a major learning experience for all of us, not only in terms of software engineering skills but also in terms of teamwork, social and organizational skills. We all made an effort to get our tasks done to the best of our abilities, and although there were periods when deadlines and tasks proved difficult and stressful, we were able to work together and prevail.

References

Agricultural intensification and climate change are rapidly ... - PNAS. (n.d.). Retrieved November 25, 2022, from https://www.pnas.org/doi/10.1073/pnas.2002548117

Semi‐natural habitat surrounding farms promotes ... - besjournals. (n.d.). Retrieved November 25, 2022, from https://besjournals.onlinelibrary.wiley.com/doi/10.1111/1365-2664.14124