Final Report
by FELEREST

Contents

1 Introduction

In this final report, the team members of Felerest present to the reader the final product. From the start, the mission was to design and construct a 'Smart Connected Cat Feeding & Monitoring System' which would eliminate the need for intelligently feeding cats either in private or public areas. This report examines each step taken in this path to achieve this goal.

Unfortunately, at the start of 2020 an unseen worldwide pandemic rose. In order for the Felerest team to adapt to the new conditions that the pandemic brought, Felerest's mission and vision had to be revisited. Until the start of the pandemic, crucial advancements were made regarding the physical system. There were only a couple of weeks left for the entire product to be completely available for market. To extend the meaning of this, the current status of the project before the lock-down should be examined. Let's examine this in 4 sections; computer vision, web design, electronics and mechanics. Felerest's computer vision engineers were working to finish the identification subsystem and discussing possible ways of optimizing it. The web design department was approximately one thirds done; the website platform was ready, some basic features like cat name, smart system location, and battery power were added. There were couple of fundamental and additional features missing and the web design part was to be integrated with other subsystems. In the electronics department everything was going accordingly, the engineers had just finished redesigning and implementing the electronics board. This board was tested and approved, the only step left was to implement it inside the system box. In the mechanics department, every 3D drawing was ready for the entire system, the outer box was to be outsourced to professional laser cutters, the inside (food flow mechanism and the reservoir) had just been implemented and was ready. The only step was to obtain the laser-cut boards, form the outer box of system and put everything together both in mechanics and electronics. The sudden pandemic left each department scattered around Turkey. Team members had different parts of the project and it was impossible to continue any physical activity without being disrespectful to social distancing. However, the Felerest team had to find a way around it so they did! The solution was to change the product itself. Since no further improvements to electronics and mechanics could be done, a team meeting was called and a new work division was agreed upon. The product itself changed from being a physical system to being a software application. The final product is the computer vision and web design algorithms. The customers are free to implement a homemade physical version of a cat feeding system or a store bought one and a camera should be installed. Later, the Felerest team will remotely install the software packages to take input from this camera and track dietary plans of up to 50 cats. Now, Felerest offers its customers knowledge in the form of software to shape the future of cat feeding and monitoring systems.

This report will cover everything from the start until the very final product. It is composed of three main parts. The first part is pre-pandemic, 'Current Status'. This part covers everything until the start of the pandemic as its name suggests. In this part, first we examine the whereabouts of each team member and their accessibility to technical resources. Later, a brief top-to-down discussion is given of the entire system. This is followed by examining the logistics of subsystems. This is a direct and thorough way of showing how the team member's access to hardware components have been affected. Lastly, the status of the subsystem is examined in full detail. Every subsystem ranging from classification to mechanics is described in this section. The reader will be able to fully implement the system by looking at these instructions.

In this part, the requirements of each subsystem is revisited, a general description of the underlying technology and the techniques used to successfully implement it is given, and the tests and its results conducted to determine the quality (or the performance) of these subsystems is present. The second part covers post-pandemic, 'Change of Plans'. This section is comprised of 6 smaller sections. The first discusses the remotely completed subsystems. These are only the algorithmic modules which were not finished before the pandemic that can be finished by individual work without necessity for hardware equipment. The second discussion is regarding the modules which are not feasible to complete under the pandemic conditions. The third discussion and fourth discussions are ideas on creative alternative solutions and how the remote integration is done. The work until this point will be demonstrated with a 5 minute demo video which will encapsulate the essence of the project and be a way for presentation the test results and case studies. Lastly, this section will include a detailed work breakdown of the Felerest team indicating role of each individual in the team. Before concluding the project, a discussion on the cost breakdown, deliverables and results will be done. The deliverables are the modules or any component that can be transferred to a future team which can finish the project as planned initially. This final report's intent is to also serve any future team with guidelines and materials.

2 Current Status

a) Team Members

- **Asude:** Asude is located in Ankara. She has no problems reaching technical resources such as internet, computer and some useful tools for programming or simulations.
- **Doga:** Doğa is in Izmir, staying with his family. He can reach technical resources such as internet and computer. The inner design is with him, however he has no other equipment to improve the system.
- **Eser:** Eser is in Ankara, staying with his family. He has no problem for connecting the internet and computer. He has only Raspberry Pi 3 in his house as a technical equipment.
- **Furkan:** Furkan is in Istanbul with access to the tools for developing the user interface.
- **Utku:** Utku is located in Ankara, staying with his family. He has no problems reaching technical resources such as internet, computer. However, he has no technical equipment in his house.

b) System Description

1. Overall System

The system consists of 3 main sub-modules: electronics, mechanics and computer vision.

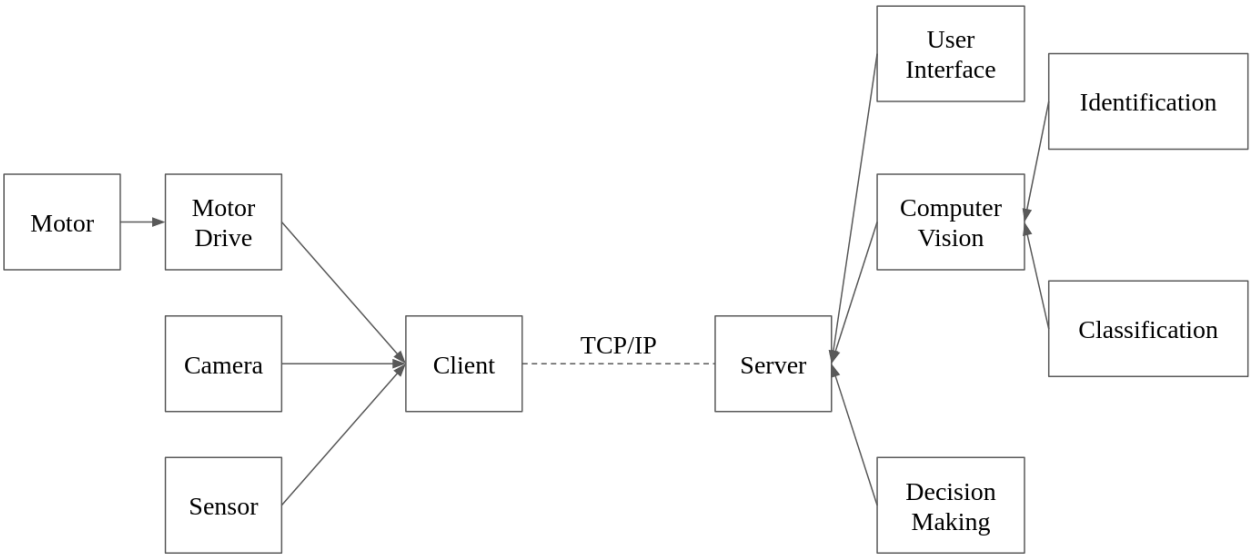


Figure 1: Overall System Parts

The system is best demonstrated with Figure 1. There are mainly 2 parts, server and client respectively, in addition to the connection between them. The connection is implemented by a computer network over the internet, and these parts have special sub-modules and missions assigned to them. System details will be explained in the next sections.

Information flow and decision making are the two crucial parts of the system. Figure 2 shows how information flows and decision is made by the system. Note that camera, Raspberry Pi, and Final Action are at client side where Classifier, Identifier, Decision Maker are at server side. Camera captures the image and it transfers the image the client, Raspberry Pi where the image is then send through TCP/IP connection to the server side. Server processes this information and takes action based on the system requirements. The image processor actually consists of two independent phases, classification and identification. Data is processed in classification program which is then sent to identification in case of a cat detected. The cat image is then identified in identification stage with image descriptors. This way, the cat name, or id, is determined and it is returned to the decision control module. Decision control module plays a role as a a top module in this manner.

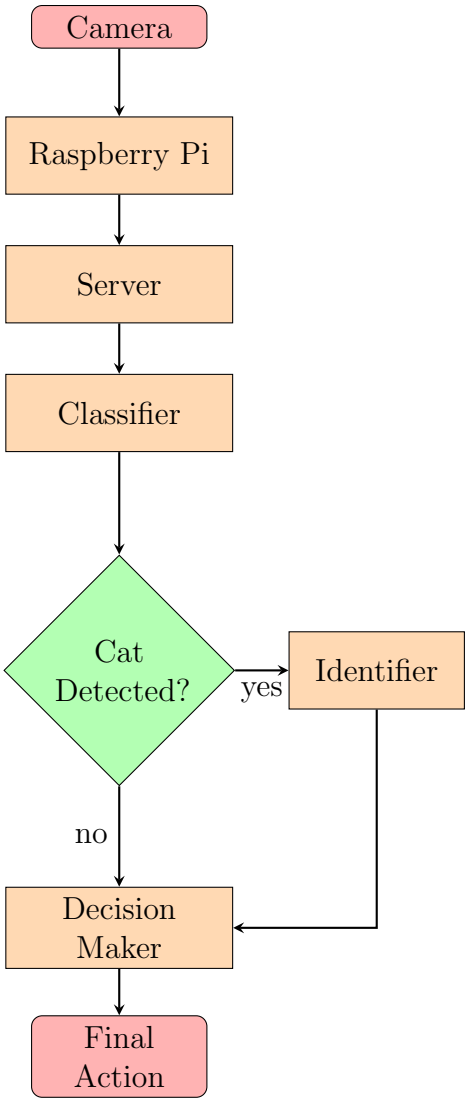


Figure 2: Information Flow Chart

2. Compatibility Analysis

2..1 Compatibility Analysis of Identification

Identification can be easily integrated into further versions since they work in a black-box manner. Moreover, virtual environments allow almost all modern CPU architectures to execute the code independent of hardware / operating system and software installed on the server system. The identifier object is also very flexible that can introduce many use areas with proper integration.

2..2 Compatibility Analysis of Server-Client Model

Clients can be replaced with similar hardware that supports specific properties such as camera, wifi connection, 256 MB at least ram, and supported CPU. The system is compatible with future versions and alternative hardware.

2..3 Compatibility Analysis of Electronics

Before the Covid-19 pandemic, all units excluding the light bulb are connected to the other subsystems, and tests are completed. However, since the outer box is not completed, the weight sensor's calibration is not completed. The other units operate at the required performance, and no error is observed during the test of the complete system without the outer box. Hence, the electronics subsystem is compatible with all other subsystems and it is available for implementation with any box with an micro-USB outlet for charging.

2..4 Compatibility Analysis of Mechanics

In this section the compatibility analysis between mechanics and other subsystems. Due to the Covid-19 pandemic, the entire mechanical design is not realized.However, the interior design is nearly finished.

The PWM signal generated in order to drive the motor can be seen from Fig 25. There is no ripple in the voltage values. It means that the motor can rotate the cylinder (Fig 17) without wobble.

Although the outer design is not realized it is designed such that all the electronic devices would fit in. The electronic devices are collected in a card that has dimensions:

$$15cmX20cm = 300cm^2 \tag{1}$$

From eqn[1] and Fig 19 it can be seen that electronic devices would fit into the second compartment if it is mounted on the front wall.

c) Logistics about subsystems

Due to the unexpected school break some critical equipment was locked in the cabinet. In the first days some of the equipment was taken however, it was not possible to rescue all. The list of the unreachable equipments can be seen below.

- Saw
- Drill
- Screwdrivers, side chisel etc.
- Voltmeter
- Oscilloscope
- Cables
- Plastics for the design of outer box.

The list of the reached equipments can be seen below.

- The inner skeleton of the mechanical design
- Raspberry pi
- Arduino

d) Status of subsystems

The current status of each subsystem is presented that has been completed until the end of the regular education period. Recall, in the critical design report, few tests were missing and some future plans weren't fully implemented. In this report, advancements done since the critical report are included. It revisits the requirements of each subsystem, gives a general description of how the subsystems work, how they were implemented, and an insight into the underlying

technology where applicable and necessary. It includes all tests performed on evaluating the subsystems in an explanatory fashion highlighting the key points such as the test's importance and how it was conducted.

1. Classification

1..1 Requirements

- Classify between cats and dogs.
- Detect the number of cats present in a given time.
- Have reliable confidence (50%) for each classification.
- Have accuracy of at least 90%.

1..2 Subsystem Description

The classification is achieved by using a Convolutional Neural Network (CNN). Neural networks are universal approximators according to the universal approximation theorem [2]. Therefore, it is clear that, any feature can be taken into account with neural networks. Since our problem was a simpler one, determining between dogs and cats, it was foreseen that the network would function with high accuracy. Also, powerful results obtained from the CNNs in recent years [4] are an indication of the efficiency and accuracy of this tool. For the interested reader, a basic neural network architecture can be seen in figure 3 which is adapted to our application. A convolutional neural network is a specific version of deep neural networks which is a specific version of neural networks which comes from the machine learning family. If we were to dig in deeper for the explanation of convolutional neural networks, figure 4 would be a simple representation which is adapted to our application. This is slightly more complex than figure 3, because it is more layered and produces a better outcome. However, in principle, its function is the same. The network was trained, validated and tested before it was ready for real-life application. Our trained network was obtained from an open source repository, YOLO: Real-Time Object Detection. The one used in our project was YOLOv3, it has a better backbone classifier than previous versions and includes scale predictions which in turn increases the performance and improves training [3]. Since this was a pre-trained model, the detection of cats and dogs were done by blocking out other classes (The original network can detect up to 80 classes.) and the detection threshold was adjusted to fit our use.

Additionally, there are few parameters associated to the performance of the system which need to be defined. Before doing so we need to make another definition. There are 4 cases of outcomes and they are as follows:

1. **True Positive Outcomes (TP):** This case arises when both the real output and the model's prediction is malignant.
2. **False Positive Outcomes (FP):** The real output is benign but the model's output is malignant. This is the most undesired case. For example, a dog was present but the model predicted a cat.
3. **False Negative Outcomes (FN):** The real outcome is malignant, prediction is benign. For example, a cat was present but our system thought it was a dog.

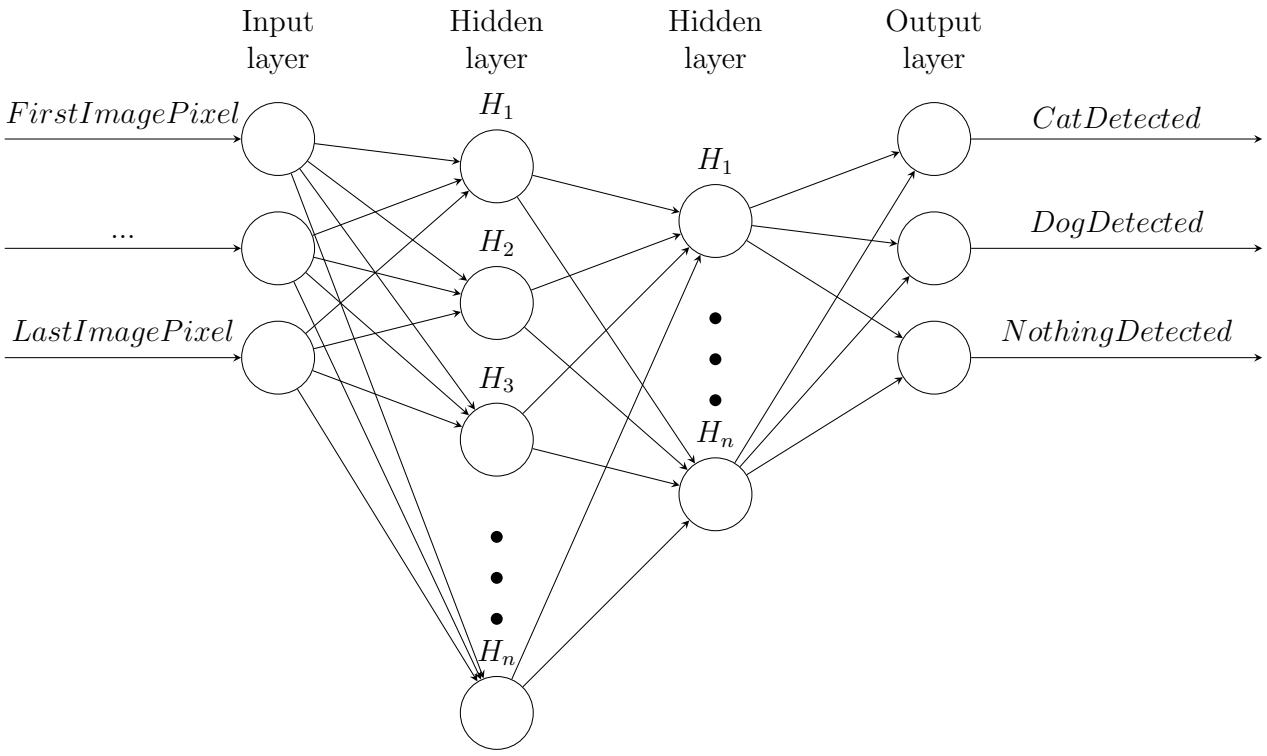


Figure 3: Proposed Neural Network Architecture

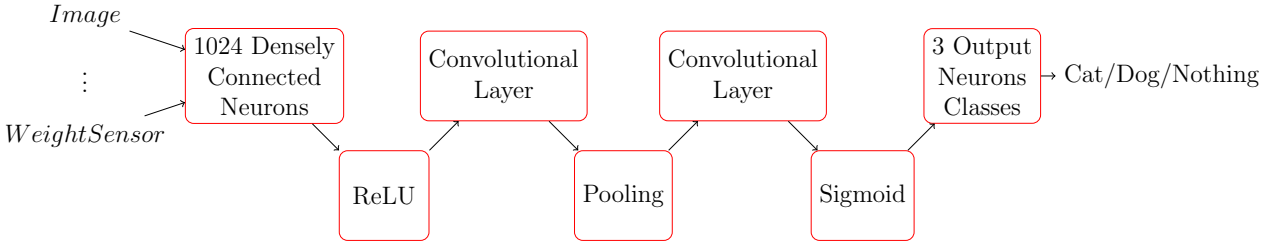


Figure 4: Simple Neural Network Architecture Design

4. **True Negative Outcomes (TN):** Both reality and prediction are benign. A dog was present and a dog was detected.

Malignant cases are the positive class, for our application it corresponds to cats. Benign case is the negative class, for our case it corresponds to dogs.

Now, we can define the parameters of the system with these definitions in mind.

- **Classification Accuracy:**

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} * 100 = \frac{TP + TN}{TP + TN + FP + FN} * 100$$

- **Classification Error:**

$$Accuracy = \frac{Number\ of\ incorrect\ predictions}{Total\ number\ of\ predictions} * 100 = \frac{TN + FP}{TP + TN + FP + FN} * 100$$

- **Confidence:** Confidence is determined within the model. It's not easy to show with a formula because it's unknown what each layer does exactly. However, we can explain it in an intuitive manner. We have two classes at the output (cats and dogs), this corresponds to two nodes in the output layer. Each node has a value at the output in the range [0,1]. If a node exceed the confidence threshold we've defined (for example 70%) than that node determines the object by assigning it to that class. If this threshold is not exceeded by any node, then the output is 'Nothing is detected'. Now, let's try to define it in mathematical terms. A neural networks approximates the following function for each layer in the network. The following is the activation values of layer 2.

$$\begin{bmatrix} a_1^2 \\ a_2^2 \\ . \\ . \\ . \\ a_k^2 \end{bmatrix} = g(\begin{bmatrix} w_{10}^1 & w_{11}^1 & . & . & w_{1k}^1 \\ w_{20}^1 & w_{21}^1 & . & . & w_{2k}^1 \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ w_{n0}^1 & w_{n1}^1 & . & . & w_{nk}^1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ . \\ . \\ . \\ x_k \end{bmatrix})$$

where g is the activation function, the subscript ij denotes unit i in layer j, w is the weights mapping layer j to j+1, and j=0 corresponds to the bias unit. The following is for the first node of the third layer.

$$f(x) = a_1^3 = g(\begin{bmatrix} w_{10}^2 & w_{11}^2 & . & . & w_{1k}^2 \\ w_{20}^2 & w_{21}^2 & . & . & w_{2k}^2 \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ w_{n0}^2 & w_{n1}^2 & . & . & w_{nk}^2 \end{bmatrix} \begin{bmatrix} a_0^2 \\ a_1^2 \\ . \\ . \\ . \\ a_k^2 \end{bmatrix})$$

Continuing these calculations, the output is obtained as,

$$f_n(W_n \times f_{n-1}(...(f_1(Image_{input}))))$$

where W_n is the weight matrix and the neural network is n layers. The confidence is the output of f_n for each node. In our network architecture, the last layer uses the sigmoid activation function. This corresponds to the following equation, also given in figure 5.

$$\sigma = \frac{1}{1 + \exp^{-x}}$$

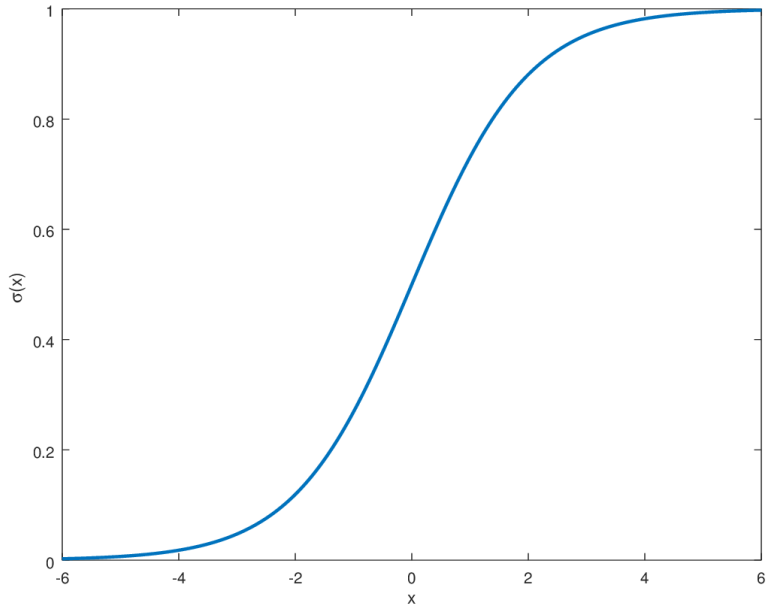


Figure 5: The sigmoid function

Now, the algorithm will be investigated and explained as a general overview. In the initialization process, first the weights of the system are obtained and later the frame is passed through the neural network to classify whether any objects belong to the 80 different classes or not. As mentioned before, we block other classes and only take outputs from cats and dogs. Therefore, the list containing the detected classes are compared with the desired classes which are cats and dogs. If there is a dog present, the system returns the string 'dog' no matter the presence of a cat. If there are cats present, the system returns the string 'cat'. If the debug mode is activated, the algorithm puts every classified object in a box with the confidence value written on top. It also writes the time that has passed to do this procedure to every frame, in other words, process time of each frame is shown. In addition to this, the algorithm contains logging info lines which log what the system is currently doing to a separate file at all times. This way, it can easily be defined what the network is doing wrong so the problem is easily detected, fixed and improved. The general flow structure of the algorithm is given in figure 6

If we are to itemize the basic principles of this algorithm and discuss the extreme cases the following would hold:

- If there is a dog present the algorithm detects it independent of number of dogs and cats present at the same time.
- If there are only cats present; first it detects the presence of a cat, second it detects how many cats are present at the same time.
- If there is nothing present (no cats and no dogs) then nothing is done.

This itemization covers the backbone of the classification module where the input is the image and there are 3 states that carry two outputs each.

Example outputs are as follows:

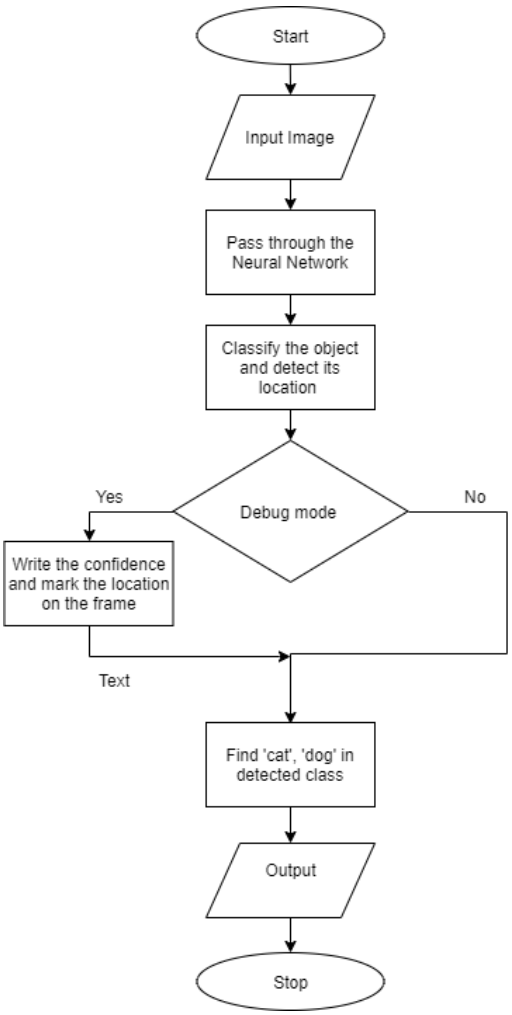


Figure 6: Algorithm Flowchart of the Classification Subsystem

- O = (‘cat’, ‘3’)
- O = (‘dog’, ‘NA’)
- O = (‘NA’, ‘NA’)

The second variable in each outcome is only important for the case where a cat is detected in the image. For other cases, where there is a dog or nothing is present at all, than the quantity doesn’t have an importance therefore it is ’NA’ or any other random number.

1..3 Test Results and Discussion

In this part, the conducted tests will be discussed in detail alongside planned future tests. Information regarding each test will be given in the following order: ‘What is the test?’, ‘Why is it important?’, ‘How was the test conducted?’, ‘The meaning of the results and its significance’.

1. **Test: How confident are we on detecting the correct class for cats?**
    - **What is it?** This was explained earlier.

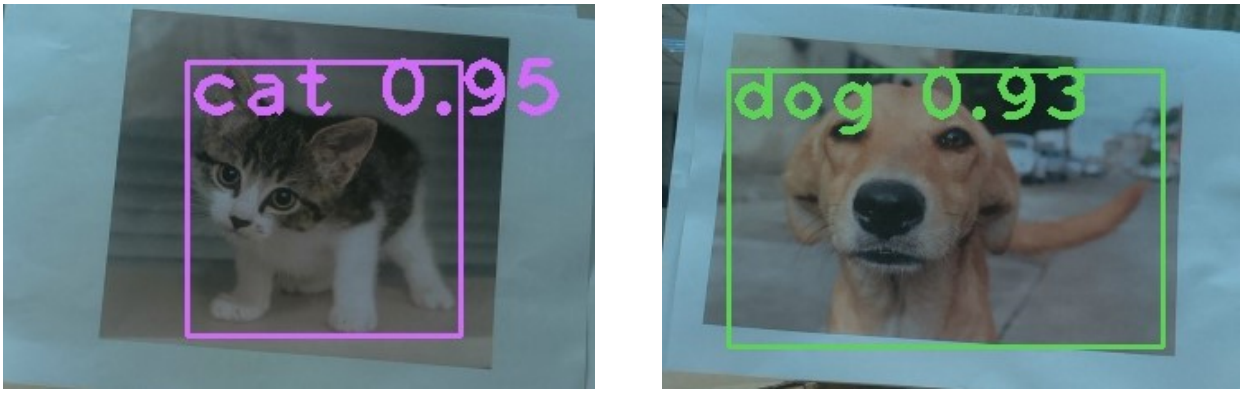


Figure 7: A cat and dog being detected with confidence of 95% and 93%, respectively.

- **Why is it important?** This test is important in determining the confidence threshold for the algorithm. This way it is possible to correctly eliminate cases where confidence is low. We are then left with correctly detected cats with high probability and no cat is left that is below this threshold.
- **How was it conducted?** In this test we investigated photos with single cats present. We collected the confidence for these images and found the estimate confidence. The photos were taken with a camera and they were of real cats. Unfortunately the cats present were 3 different types due to the lack of finding willing real cats to test with. These cats' photos were taken in different angles and positions.
- **Results** There were 51 photos in total and the average confidence of cats detected in these photos were approximately 84%. It should be noted that this is a successful and high result. The threshold when for these photos were around 50%, meaning that any cats detected with a lower confidence than 50% were omitted and regarded as a failure. The average result being 84% implies that 50% was a successful result since it exceeds it largely. Figure 7 is a great example of this. Please take regard to the image quality in this case. As can be observed the cat is very dark in this image and the contrast is low. The cat is also very close to the camera and only a small portion of the nose and left ear can be easily observed even with a human eye. This demonstrates the performance of the classification algorithm.

2. **Test: How current is the false positive cases?**
   - **What is it?** False positives are the worst cases where a negative situation is detected as a positive situation. In this module if dogs are detected as cats this is the most undesirable scenario. So, we want to run images of dogs and see if these dogs are detected as cats. If so, we want to detect the likelihood of false positive cases occurring. This might give us a specific cause where we can limit the code to avoid false positives.
   - **Why is it important?** This is important because then dogs are fed with cat food which increases the likelihood of the dog revisiting the feeding system for food. We need to detect false positive cases so a solution can be found to prevent this case.
   - **How was it conducted?** Since finding real dogs are harder than finding cats we

had to test with only images of dogs. The important restriction to be kept in mind is that the photos found online are usually photo-shopped and doesn't give a good representation of reality.

- **Results** There were 6 different dog photos, by capturing different orientations and distances of these photos with the camera, we analyzed 24 cases. Out of these cases there was only a single case where a dog was detected as a cat. We believe that in real life cases this will be reduced even further to perhaps 1 false positive in a 100 dogs. Since there are multiple frames sent per second and a final result is obtained from these images this result will not cause a problem. Because in the final result the dog will be detected eventually in one of the frames taken over a small interval of time and food will not be distributed.

3. **Test: What is the accuracy of our classification system?**
   - **What is it?** Definition of accuracy was defined earlier.
   - **Why is it important?** It gives a measure of the performance of the network. It also defines the previous test in a more numerically compact manner.
   - **How was it conducted?** By following the data in 1 and the accuracy function defined earlier, we extracted the result.
   - **Results** We found that the system accuracy was **80.27%** and the system error was **11.27%**. It should be noted that unclassified cases were also included in the denominator of the accuracy and error functions. These functions are originally defined for binary cases. In our situation, the 'undefined case' is not a class but it is a possibility so it should be included in the function as we did.

The data for the previous 2 tests are summarized in table 1. It should be noted that, tests are done in extreme conditions where light is very little, and view angle is narrowed by intent. The 'correct' column indicates that the corresponding class has been identified correctly. 'False' indicates if the class present in the photo was detected as another class, for example; a cat was present and a dog was detected. 'Unclassified' corresponds to a class being present and the algorithm not detecting anything at all, in other words not realizing the presence of a cat or a dog. 'TOTAL' is the number of photos taken with the camera and the test was ran over these photos. There were 112 cat photos present with real cats and photos of cats. This test data-set includes multiple photos of the same cat with different backgrounds, light settings, contrasts, orientations, positions and such.

	Correct	**False**	**Unclassified**	**Total**
Cat	91	15	6	112
Dog	23	1	0	24

Table 1: Table of test results for cat and dog classes

In result, the cases that are either unclassified or falsely detected does not have significance. This is because while the camera is ON, the algorithm only processes frames taken from each time interval. This time interval is determined by the speed of the algorithm (both the preprocessing an the time it takes for the algorithm to process the frame) and

to some extent the capability of the camera. Since there will be x many frames processed in a second there will be higher likelihood of correctly classifying the object. In other words, a single frame that is wrongly classified or unclassified will be ignored and have no importance. In reality, this indicates that accuracy might be slightly higher. To be certain that this won't be an issue and to have an idea about the processing speed, it is best to do some tests.

4. **Test: The processing speed of the classification algorithm**

   - **What is it?** This test will determine how many second it takes to run the classification algorithm on a single image. It will give us the rate.
   - **Why is it important?** As mentioned before, this test is important for the next step of the computer vision part. After classification it should be made certain by somehow mediating over a selected interval of time. This part of the process can even be determined perhaps after the identification part and before commanding the mechanic part for food flow. This way the accuracy is surely increased.
   - **How was it conducted?** For this test, we placed time stamps throughout the code to first determine the most time consuming part of the code. After doing this we evaluated the time consuming sections of the code to decrease the time further. After repeating this process a several time, we outputted the total time spent over a single frame. An important note to be made here is, the initialization part of the code was omitted. The initialization is done only once in the beginning when the camera is turned ON or the system is started over. For example, these parameters are weights that need to be downloaded once in the very beginning. Basically a single time stamp was placed in the code after the initialization part and at the very end where the code is executed or started over. Then the time elapse was collected by subtracting the result of these two time stamps.
   - **Results** The rate was found as **3 frames per second**. In other words a single frame took nearly 0.3 second to process. As mentioned earlier, at the beginning the time for each frame to be processed was longer (more than 0.5 seconds) which sometimes meant not more than a single frame was processed in a second. By adapting the code for better performance, the achieved final time elapse was significantly smaller.

   It should be noted that the time required for single frame's processing is a varying parameter. The detection speed changes under conditions where there is light exposure, many object, color variations, movement, contrast changes, fast moving objects that cause blurring in pixels and such.

5. **Test: The distance from the camera for successful classification**

   - **What is it?** This test is to have an idea about the camera's distance range and the classifications capabilities. At the end we will be able to determine how successful we are in classifying close range objects and objects that are far away.
   - **Why is it important?** This is important for two reasons. One is for the standby state of the machine. For low power consumption the camera will be off at idle, when there are no pets approaching the box. However, there will be a sensor which detects movement. When a movement is detected the camera will turn on and

start recording. The information of farthest detection is important such that the movement sensor will have a similar distance range. For compatibility ans successful operation this is crucial. The second reason is for the developers. This knowledge is important so the requirements are satisfied. A short range is undesired since detection and classification becomes difficult. This might cause cases where the cats are present but its undetected because ts out of range. Therefore, the range should be somewhat large.

- **How was it conducted?** An image of a cat is held in front of the camera at a very close proximity and waited for processing, then the image is moved further away from the camera inch by inch. At a distance where classification is unsuccessful is marked and measured. A similar experiment is done to detect the the lower distance boundary of detection. Of course the images are chosen from the data-set with high confidences such that the reason for non-detection is not dues to that specific cat photo and the test is conducted under close to ideal conditions.
- **Results** Detection distance is **5-150 cm for cats** and **10-180cm for dogs**. This range was found to be sufficient for robustness and high performance.

The entire data set is given in this link. Some of the tests were done on images obtained from google randomly. These images were printed on A4 paper in large scale and held in front of the camera at various angles from various distances. Because of the enlarged photos, low quality of the printer and the images itself there were some uncompensated noise factors. Some were done on real cats as indicated.

2. Identification

2..1 Requirements

System should identify cats and feed them accordingly with a high accuracy. The requirements are listed specifically below.

- The system shall identify cats with accuracy greater than 0.9. Accuracy is defined as the correctly identified cat samples over all samples.
- The system shall recognize new cats event and add them to database. Maximum allowed shots for the new cat is 2, and no human supervision should be included.
- The system should work with a reasonable time response. Up to 10 seconds are acceptable.
- A single device shall occupy space less than 1GB on the server for database.
- The system should self-optimize in case of any need.
- The database shall be self-adaptive. Adding - removing cats are always supported.
- The system shall be capable of handling 15 active cats at the same time under these limits.
- Arbitrarily many cats should be stored in the database.

2..2 Subsystem description

Feature descriptors are used in identification. SIFT and ORB implementations are used and feature vectors are created for the interest points that are calculated based on known algorithms SIFT and FAST. For commercial purposes, ORB is decided to be used; however, SIFT is used in research of the product because of its slightly better performance. Note that SIFT is patented which is why ORB is going to be used in the mass production software of the product.

Identifier module requires an image with strict bound. Cropped image is directly sent to the Identifier module from Classifier module. This strict bound both prevents excessive data coming from background and smaller set of vectors to be processed. Since the angle that cats coming to eat the food is deterministic and more or less the same, background details are mostly eliminated, and an image consists only of cat head. This way, computation burden and noise are minimized.

2..3 Test Results and Discussion

The test methodology is based on accuracy tests with two data sets. Online images are mostly consisting of random cat images that are not identified; or a specific cat only appears in the pictures. Therefore, a data set is created based on the cat images shown on the computer screen and shots taken from the screen as well as Facebook group volunteers that give samples for different cats with different names. Consequently, there exist two separate data sets. The first one is set of images for the same cat on the screen, and a few number of classes (cat identities), but a lot of samples. The second one is data set with various classes of cats but with very limited data.

3 different cat images found on the internet. These cat images are displayed on the computer screen and total of 100 shots were created for each cat image that is displayed. Among these 100 images there are only 60-80 possible samples which include cat, or classified correctly. Since camera gets blurry when moving and sometimes it points to different directions, it was not always possible to take pictures of cats. Therefore, only 40 - 60 sample is generated for the cat image displayed on the screen. It should be noted, to show background effect, different backgrounds, different display positions and different displays are used to display image.

Cat windows inside these images are extracted using the classification code and database creation script. 5 samples per class are separated for test set. Note that since there are no hyper-parameters currently optimized, there is no validation set selected and only train and test sets became in use.

The other database consists of 17 different classes and 4-7 samples for each class. This is a good example of few data identification. The data set is divided to train and test sets, where test set consists of 1 sample of each class, note that data is very limited. This seems a biased approach; however, system is designed for the conditions of excessively limited data. Tests are done on both data sets for different approaches and the hybrid approach at the end, and the results are given in tabular and graphical forms.

Confusion matrix representation is given in figure 8 and 9 for two different data sets explained above. Moreover, technical information on database provided in figures 10 and 11 for this data sets. Note that in case of very little data, there are hardly SIFT vectors found. Test results show that, with correct camera position, and enough data accuracy can easily can go around 1.0

Confusion Matrix					
		Predicted			
		Cat 1	Cat 2	Cat 3	Cat 9
Ground Truth	Cat 1	5	0	0	0
	Cat 2	0	5	0	0
	Cat 3	0	0	5	0
	Cat 9	0	0	0	5

Figure 8: Confusion Matrix for Set 1 - Test 1

Confusion Matrix																
		Predicted														
		Selin	Seda	Melis	Haza	Ezgi	Esra	Esr	Ekin	Dilara	Ceren	Bahar	Ase	Cem	Deniz	Ezg
Ground Truth	Selin	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Seda	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	Melis	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	Haza	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	Ezgi	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
	Esra	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
	Esr	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	Ekin	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
	Dilara	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
	Ceren	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	Bahar	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	Ase	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
	Cem	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	Deniz	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
	Ezg	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Figure 9: Confusion Matrix for Set 2 - Test 2

```
Number of classes in total : 3
Number of vectors per class :
        bir                             : 347024
        dokuz                           : 662830
        uc                              : 52026
        ------------------------------- : ____+
        Total                           : 1061880
```

Figure 10: Database Information for Test 1


```
Number of classes in total : 17
Number of vectors per class :
        ase                          :   164
        bahar                        :   154
        cem                          :    10
        ceren                        :   118
        deniz                        :     2
        dilara                       :    18
        duygu                        :    24
        ekin                         :    24
        esr                          :   156
        esra                         :  4130
        ezg                          :  1082
        ezgi                         :    18
        haza                         :     0
        hazal                        :     0
        melis                        :    46
        seda                         :    40
        selin                        :   148
        ---------------------------- : ____+
        Total                        :  6134
```

Figure 11: Database Information for Test 2

```
Number of classes in total : 3
Number of vectors per class :
        bir                          :   308
        dokuz                        :   146
        uc                           :   136
        ---------------------------- : ____+
        Total                        :   590
```

Figure 12: Database Information for Test 3

```
Detailed accuracy report :
        Cat name selin with accuracy   : 1.000000
        Cat name seda with accuracy    : 1.000000
        Cat name melis with accuracy   : 1.000000
        Cat name haza with accuracy    : 1.000000
        Cat name ezgi with accuracy    : 1.000000
        Cat name ezg with accuracy     : 1.000000
        Cat name esra with accuracy    : 1.000000
        Cat name esr with accuracy     : 1.000000
        Cat name ekin with accuracy    : 1.000000
        Cat name dilara with accuracy  : 1.000000
        Cat name deniz with accuracy   : 1.000000
        Cat name ceren with accuracy   : 1.000000
        Cat name cem with accuracy     : 1.000000
        Cat name bahar with accuracy   : 1.000000
        Cat name ase with accuracy     : 1.000000
        Cat name bir with accuracy     : 0.978261
        Cat name dokuz with accuracy   : 0.932584
        Cat name uc with accuracy      : 1.000000
Calculated accuracy is 0.9597701149425287
```

Figure 13: Accuracy Tests with New Cats to the System

whereas different cameras with different angles make it to reduce even 0.3125 which is the case in figure 9. However, for practical application, camera angle and viewpoint is strictly defined and these problems are safely ignored.

Above discussion distinguishes the behaviour on two separate sets. However, there is no single control variable to measure the actual performance. Since these data sets differ in both size and capture techniques, another data set is created which is small in train images but data collected with the same camera in different environments. This data set is tested for single shots. Single shot of a cat is taken from the camera and remaining images are put as test set. In other words, for 3 different classes, there is only 1 train sample, and 40-60 test samples. The results are pretty good in this test, accuracy is around 0.96 which is given in figure 13. In addition, new cats which are the cats the system did not see before are identified as 'None' meaning it is a new cat. The accuracy is exactly 1 which is because a threshold is dynamically adjusted for the train samples with a little loss in accuracy of the known cats, only around 0.02 - 0.07 as can be seen in figure 13.

Test results show that the system is capable of handling the cat identities with a higher accuracy. Despite the little data sources, more than 300 images and 20 classes are used in the test procedure. Identification times, model train, database creation time are given below for operative perspective.

- **Database creation (164 images) (54 per class) :** 6 hours 30 minutes 33 seconds
- **Database creation (74 images - 4.3 per class) :** 12 minutes 45 seconds
- **Database creation (3 images - 1 per class) :** 4.65 seconds
- **Image match (average) :** 0.4 seconds

The tests are done on a regular old personal computer without any acceleration or turbo technology. Single core - single thread is used in the process for better accuracy.

```
Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   43 bits physical, 48 bits virtual
CPU(s):                          8
On-line CPU(s) list:             0-7
Thread(s) per core:              2
Core(s) per socket:              4
Socket(s):                       1
NUMA node(s):                    1
Vendor ID:                       AuthenticAMD
CPU family:                      23
Model:                           24
Model name:                      AMD Ryzen 5 3500U
                                 with Radeon Vega Mobile Gfx
```

3. User Interface

3..1 Requirements

This module deals with the delivery of information regarding the device and its cats to the owner (user) of the device via the Internet. Following are the required features:

- Device's food supply level
- Device's battery level during charging
- Device's battery level during operation
- Device's feeding log
- Cats' profiles

An explanation and reasoning of the requirements is due:

Food Supply: Food supply level information helps the user decide when to refill the food tank.

Battery Level: This is essential for any rechargeable product.

Feeding log: Each device has a feeding log of all cats that have been fed from it since its setup.

Cat profiles: These profiles are initially handled by the Identification module. Each profile includes a name, an image and an individual feeding log. Furthermore, the name attribute is initially 'name' and can be edited later on by the user of the device. Nonetheless, the user cannot edit his/her cats' images: the images are provided by the device's camera. Feeding log is also not to be edited.

3..2 Subsystem Description

For the interface, a Python web framework called Flask is used. There were a few reasons behind this choice:

- Our user interface and web designer, Furkan, had more experience in Flask than in other frameworks.
- Since Flask uses Python in the background, CPU time is less than other platforms such as Nodejs.
- There is no need for an additional server, since a command as simple as 'flask run' can let us run our website on our computers.
- Because the communicated modules are mostly developed in Python, the communication is seamless.

The user interface, prototype of which is currently hosted on
felerest.pythonanywhere.com, consists of 3 main pages: devices page, cats page, settings page, all of which can be accessed only with the user's password.

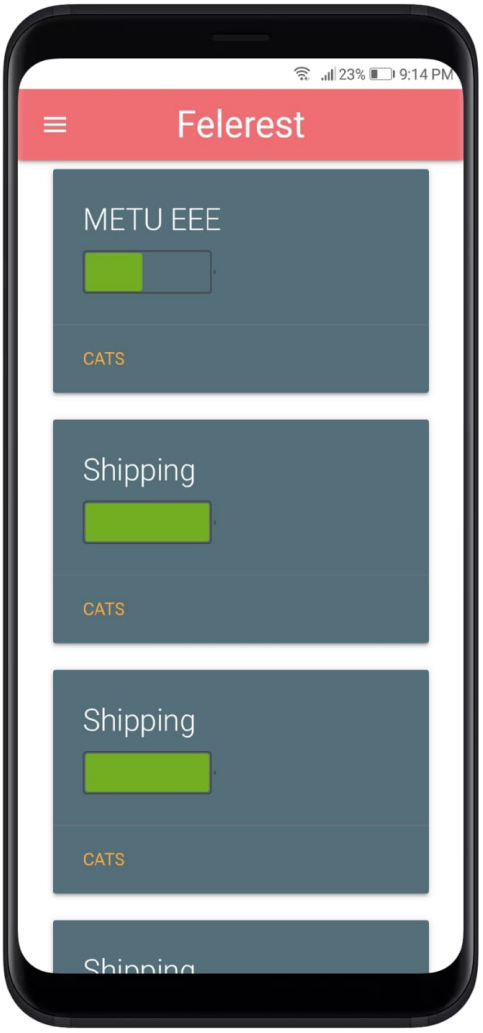
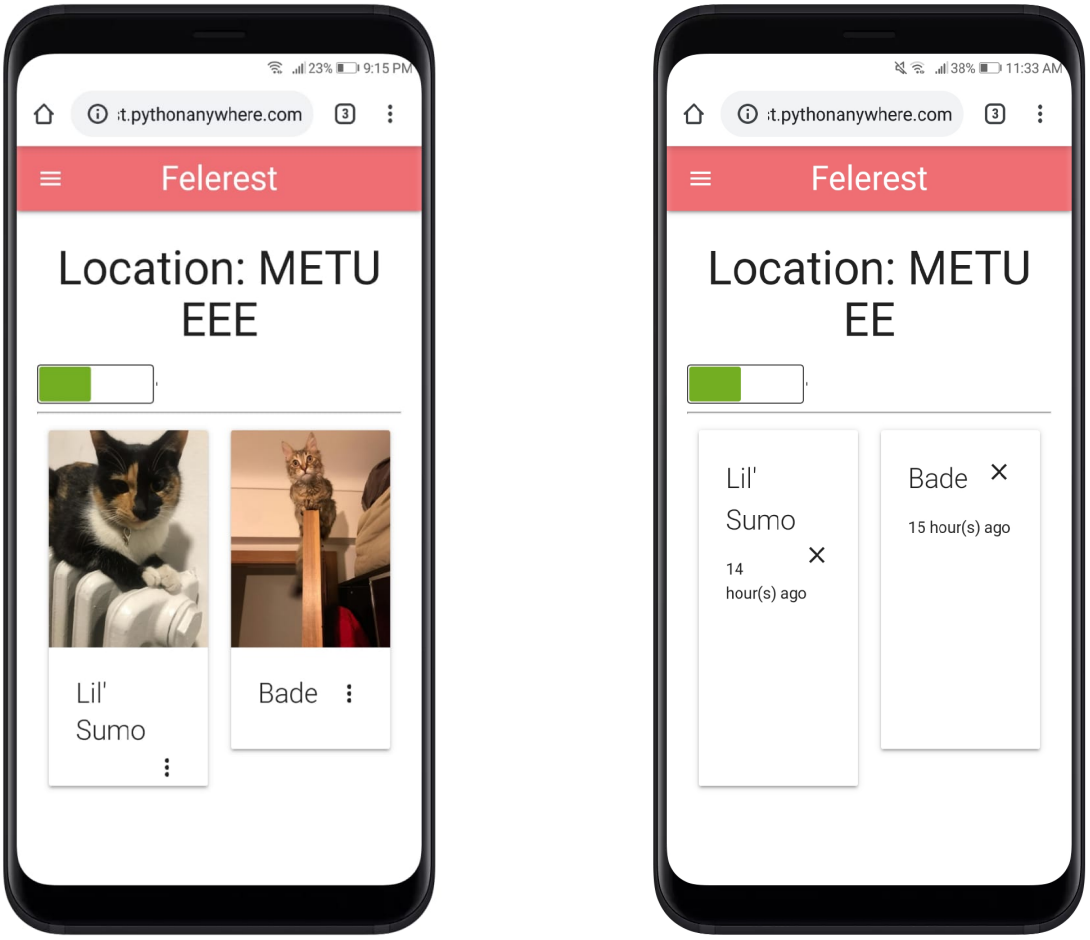


Figure 14: User Devices Page

The devices page (Figure 14) displays all devices that the user owns as cards in a grid. Each card shows that particular device's location, battery level, and food supply level. The food supply level indicator will be added eventually. There is also a link at the bottom of each card that takes the user to that device's cats page. There will also be another link that displays the entire feeding log of the device.

Cards in the figure labeled Shipping are just dummy devices to test the behavior of the grid with multiple cards.

The cats page (Figure 15) displays all cats that belongs to a device. On the page, the location, battery level, and food supply level of the device is displayed again for convenience of the user. Moreover, each cat has its own card that displays a name and an image. When the user clicks on a cat's card, the last feeding time of the cat is revealed, along with a link to the cat's feeding log. Currently, only the last feeding time is implemented.



(a) Device Cats Non-revealed

(b) Device Cats Revealed

Figure 15: Device Cats Page

The last essential page, the settings page, can be seen in Figure 16. Here the user can change the locations of devices and names of cats that are affiliated with him/her. The changes that are done here are implemented instantly. Locations/names 'METU EEE', 'Lil' Sumo', and 'Bade' have all been given as inputs to this page.

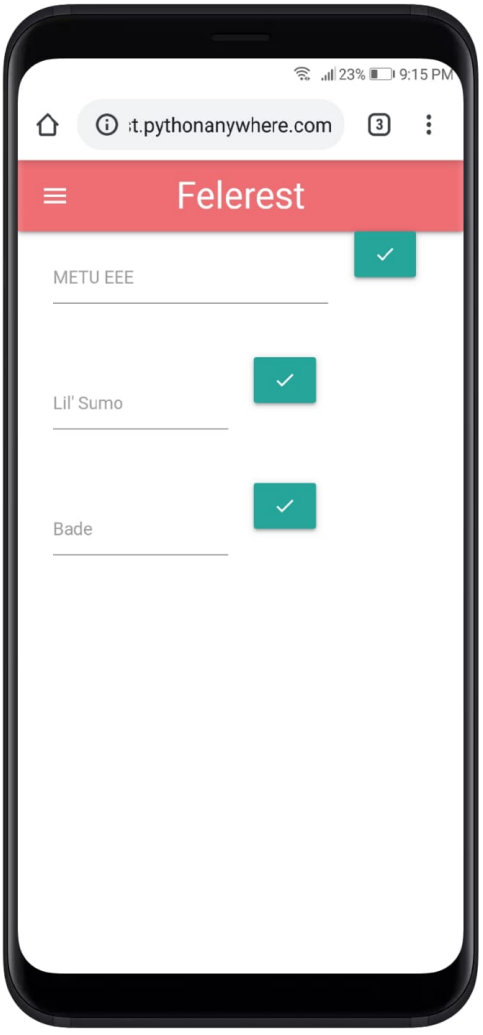


Figure 16: User Account (Settings) Page

4. Mechanics

4..1 Requirements

- Total system should be lifted by an ordinary person.
- Volume of the box should be enough for all electronic devices to fit in.
- Box should be durable for harsh outdoor conditions.
- Food reservoir should be provide at least 100 meals for an ordinary cat.
- There should be maximum %10 waste of food.
- Angle and the position of the camera must be adjusted properly in order to identify cats and dogs from 1.5 meter.

4..2 Subsystem Descriptions

The mechanical design is composed of two sub-parts.

- Interior Design
- Exterior Design

Interior design is implemented whereas the exterior design is not finished completely due to the Corona pandemic.

1. ***Interior Design***

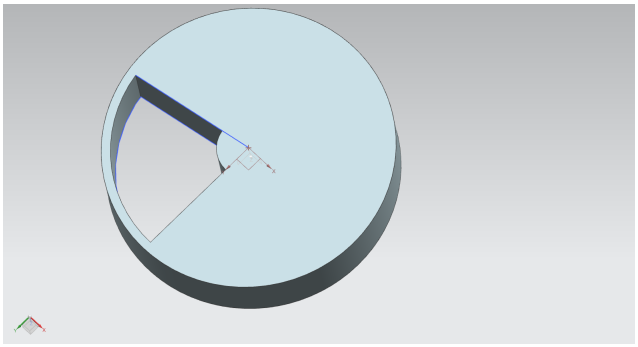
   In this project it is expected to give a different amount of food to different cats. To solve this problem an interior design is planned. Interior design is composed of the food mechanism part. The food mechanism part is designed such that the food given to the cats is completely controllable in the sense that the mechanism can give the desired amount of food in a %10 error rate. The apparatus used for this purpose can be seen from Fig 17. To control the food amount a cylinder shaped apparatus is used. The apparatus has three purposes.

   - To stop the food flow
   - To control the amount of food given
   - To allow the food flow.

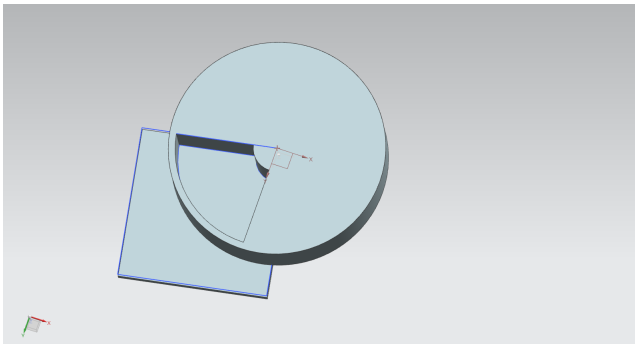
   From Fig 17 it can be seen that the cylinder has two states namely Off and On state.

   In the OFF state the empty side of the cylinder is blocked with an obstacle such that there isn't any food flow from the reservoir to the out of the device. However in this position the food is flowing from the reservoir through the empty region of the cylinder. The flowed food is stored in this empty region, waiting for a cat to arrive. The following paragraph discusses the necessary amount of food storage needed for a single cat.

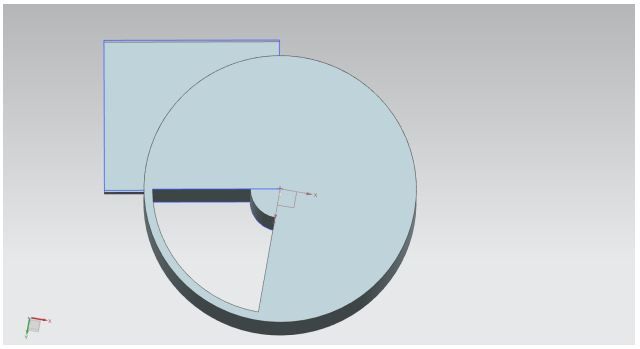
   From the literature it is found that for an ordinary cat which has weight approximately 4 kg is advised to eat 60 gr of food per day.[1] Our system is designed such that it will feed the same cat twice in a day. Therefore for one meal, a single cat will eat 30 gr of food. In order to sustain this amount with a minimized error rate, the empty side of the



(a) Cylinder



(b) Cylinder in Off state



(c) Cylinder in On state

Figure 17: The cylinder shaped apparatus for food mechanism

cylinder is selected to have 391 cm^3 eqn[3]. This value is selected since it is expected to provide an average of 15 gr of food in each rotation. This means that for an ordinary cat the system has to rotate two times to sustain 30 gr food. The experimental results for the poured food amount can be seen under Test Results and Discussion Part. In the Test Results and Discussion Part it is observed that the average poured amount is 17 gr. Therefore it can be said that still, to sustain 30 gr of food one has to rotate the cylinder two times.

In the ON state (Fig 17 (c)) the empty side of the cylinder is no longer blocked by an obstacle, therefore it corresponds to the state where we give food to the cat. When the cat arrives the classification and identification sub units detect and categorize the cat. After the categorization process the amount of food that should be given is known. In order to sustain this amount, the cylinder is rotated by using a servo motor namely MG996R (see electronics part for a detailed explanation about the motor). From the last discussion the cylinder is expected to pour 17 gr of food in each rotation, therefore to sustain this amount for an ordinary 5 kg cat, who eats around 22.5 gr to 40 gr in every meal[1], an average of two rotation is needed. The time required for one rotation is 1.5 sec which extra 3 seconds are needed for refilling the empty region of the cylinder, the total time required for sustain 34 gr food is:

$$T = 1.5 + 3 + 1.5 = 6sec \tag{2}$$

The decrease of the required time is possible, however, it will result in additional errors such as, food spreading and unstable operation, therefore this time value is selected for

optimal purposes. It is the best solution for the trade-off between the stability of the circuit and the lowest time for the operation.

The technical drawing for food mechanism can be seen from Fig 18

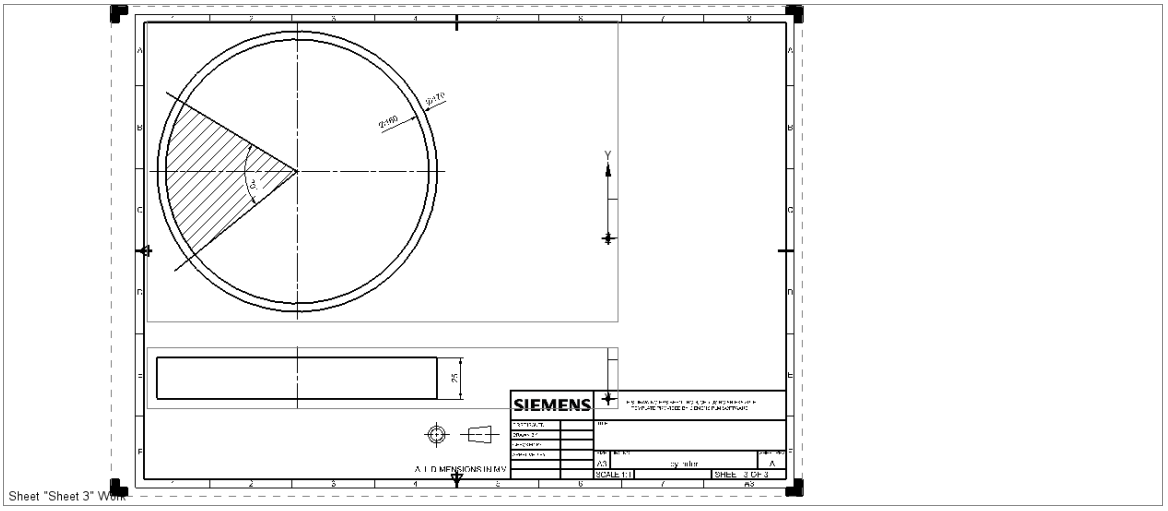


Figure 18: Technical Drawing for Food Mechanism

The empty region is the region which is marked with a dashed line. From the Fig 18 it can be seen that the volume of the empty region is:

$$pi * (160mm/2)^2 * 25mm * 70/360 = 97.7cm^3 \tag{3}$$

The base of the food reservoir has a 15 cm radius. Therefore the cylinder-shaped apparatus is designed such that it has a 17 cm outer radius (Fig 18), in order to minimize the undesired food flow from the ends of the cylinder. The results of the tests conducted can be seen under Test Results and Discussion. The food mechanism is realized using 3D printers.

From the discussion based on the daily food amount of cats, the mass and the volume of the reservoir can be calculated. From the requirements parts it is suggested that food reservoir should provide 100 meals for an ordinary cat. From [1], the mass of a one meal is 30 gr. Thus the minimum mass of the reservoir becomes:

$$100 * 30gr = 3kg \tag{4}$$

The mass of the empty reservoir is 0.4 kg. Therefore the mass of the reservoir in total becomes:

$$3kg + 0.4kg = 3.4kg \tag{5}$$

The mass of the servo motor and food mechanism is 1 kg and 0.2 kg, where in total the mass of the inner design becomes:

$$3.4kg + 1kg + 0.2kg = 4.6kg \tag{6}$$

2. ***Exterior Design***

Exterior design is the part that consists of the outer box. Due to the Covid-19 pandemic the realization of the outer design is not possible, however, the 3D drawings and theoretical calculations done. In this section one can found the discussion about how to implement the outer box, it's 3D drawing, the technical drawing of the outer box, and the theoretical calculations related to the dimensions of the outer box.

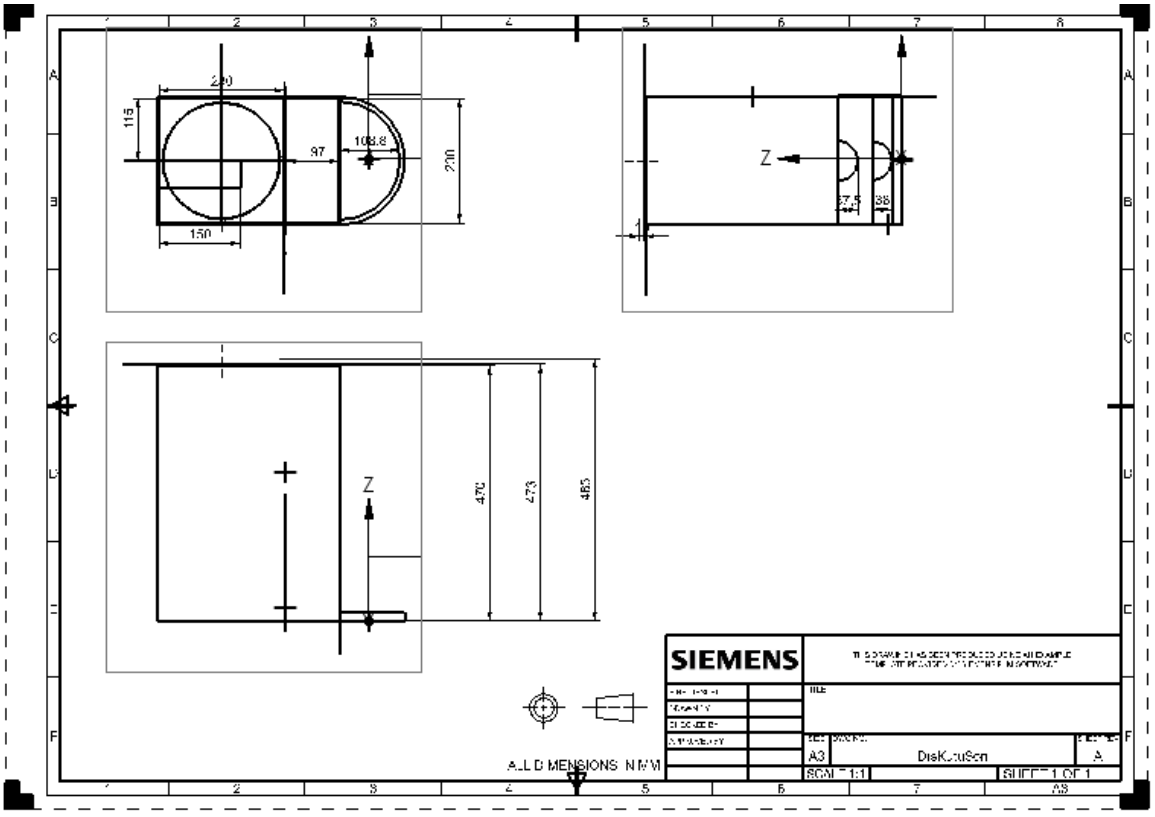


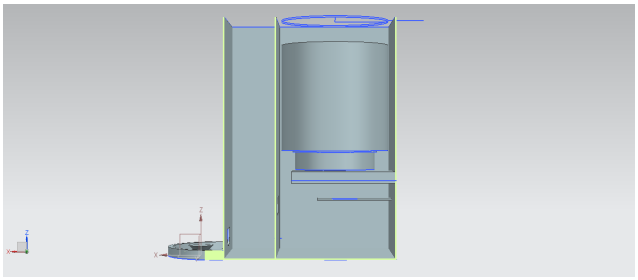
Figure 19: Technical Drawing for Box

Fig 19 represents the technical drawing of the outer box. All dimensions are in mm scale. The outer box has a two compartment, namely the first and second compartments. The first compartment is the leftmost square in the base view whereas the second compartment is the middle rectangular portion (Fig 19). The box is designed such that, the reservoir, food mechanism, and servo motor will be placed into the first compartment, whereas electronic devices will be placed into the second compartment.

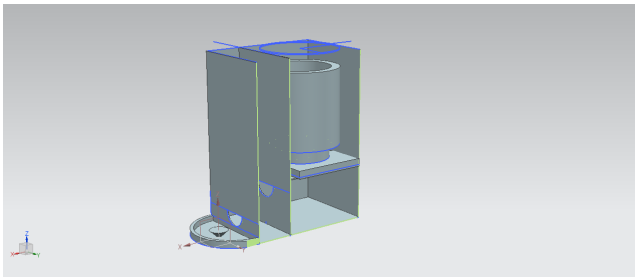
In order to get a clear understanding about the design the reader can look Fig 20.

The dimensions of the first compartment are 23X23 cm^2, where the top radius of the reservoir is 21.5 cm. By placing the reservoir into the middle the first compartment one gets a 0.75 cm gap in each direction (eqn[7]).

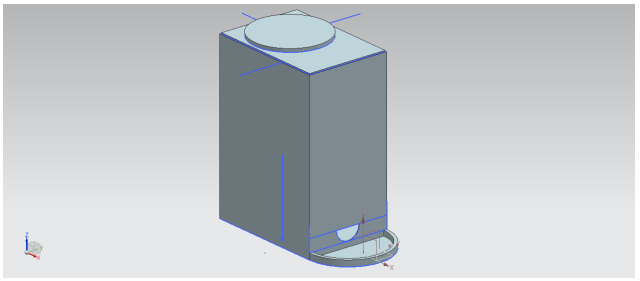
$$Gap = (23cm - 21.5cm) = 0.75cm \tag{7}$$



(a) The inside of the box 1



(b) The inside of the box 2



(c) The outer design

Figure 20: The 3D drawing of the outer Design

This empty region is needed in order to control the stress on the food reservoir due to screw that will connect the wall of the box and reservoir.

To prevent the cats to reach the food reservoir, the reservoir is placed at the back of the box. By arranging the placement like this, it is planned that the cat cannot reach the reservoir since the food mechanism is 20.5 cm away from the cats. (Fig 19) In addition to that food reservoir and the outer receptacle is connected with a closed half-cylinder shaped apparatus (not mentioned showed in schematics due to simplicity), which lies through the two half-circled hole (Fig 20).

The height of the box is 473mm.(Fig 19) This value is selected in order to fit all electronics and mechanical components into the device. From Fig 20 it can be observed that there exists an extra gap from the bottom of the box and food mechanism. This extra gap is necessary to create a high slope for the food path. The gap is 27.4 cm from the bottom of the box which creates a high enough slope. (eqn[8]) In order to calculate the slope one can think the motion is happening on the plane since there isn't any motion on the y-direction. Then the slope is 0.73(eqn[8]).

Δ x = The Distance of Food Mechanism from the front surface of the second compartment

Δ z = The Distance of Food Mechanism from the bottom of the box

$$m = \Delta z/\Delta x = 274mm/(108.8 + 97 + 215/2)mm = 0.73 \tag{8}$$

In the second compartment the electronic devices will be mounted on the surfaces. The hole on the front surface (Fig 20) will be filled with the half-cylinder shaped food path, therefore the access from hole to electronic equipment's are not possible. This guarantees that cats cannot reach, and harm the electronic components.

The device is designed such that user can only reach the reservoir. There will be a cover on top, that user can open it and fill the reservoir (Fig 20c).

4..3 Test Results and Discussion

Since the outer design is not realized, the only tests are conducted to interior design.

1. ***Test Description***

   The following test are conducted in order to observe the characteristics of the flow, determining the error parameters, calculating the errors and determining the average poured food.

   The tests are conducted in a closed environment, that wind effects are neglected. The inner box is placed on to a solid surface, that there isn't any dent on the food mechanism. The test is conducted in an optimal environment in which, there isn't any pressure on the food mechanism rather than the gravitational force of food itself.

   In order to determine the average weight of the food poured the weight of the food collected in the receptacle is measured, whereas to determine the waste food amount the amount of the food poured to the surface (everywhere except the receptacle) is measured.

2. ***How was it conducted?***

   Firstly, the cylinder is arranged in an OFF situation. The food reservoir is half-filled with the food. The state of the cylinder is changed to ON state, then waited 0.3 sec, then the state of the cylinder is again changed to OFF state (Fig 17). The weight of the food poured into the receptacle and the floor is measured. This procedure is repeated until the reservoir is empty.

3. ***Calculation of error parameters***

   The test results can be seen Fig 21.

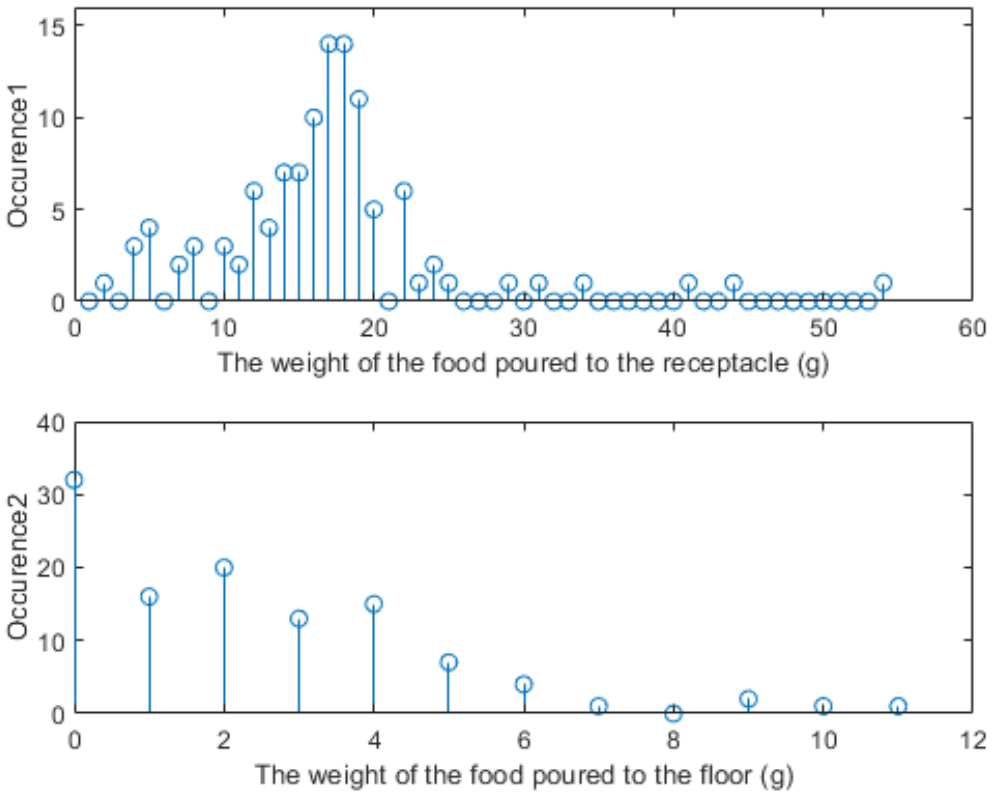


Figure 21: Technical Drawing for Food Mechanism

The first graph shows the weight of the food poured to the receptacle. Y axis shows the occurrence number and x axis shows the weight of the food flowed. For example among all trials there exist 4 occurrences in which the amount of food flowed to the receptacle is 10 g.

The second graph shows the weight of the food poured to the floor. Y axis shows the occurrence number and x axis shows the weight of the food flowed. For example among all trials there exist 20 occurrences in which the amount of food flowed to the floor is 2 g.

- ***Average Food Poured***

  Average food poured is calculated as follows.

$$m_{average} = 1/n \sum_{i=0}^{n} occurrence_1[i] * i = 16.7 = 17g \tag{9}$$

- ***Total Food Poured***

$$m_{total} = \sum_{i=0}^{n} occurrence_1[i] * i + occurrence_2[i] * i = 2131g \tag{10}$$

- ***Waste Rate***

  Waste is defined as the food amount which is poured to the surface rather than the receptacle.

$$m_{waste} = \sum_{i=0}^{n} occupancy_2[i] * i = 260g \tag{11}$$

$$WasteRate = m_{waste}/m_{total} = 0.12 = \%12 \tag{12}$$

4. ***Results***

   - It is observed that average food poured is 17g
   - The waste rate is around %12. This rate is higher than the maximum value defined in the requirements. It still needs improvement however, this result is also tolerable. The improvements are not done in order to improve the waste rate due to Covid-19 pandemic.

5. Electronics

5..1 Requirements

The requirements for the electronics subsystem are listed below.

- The subsystem should be rechargeable, and the charging time should be less than 5 hours.
- Rechargeable batteries should be non-removable.
- Battery duration should be at least 5 hours.
- The subsystem should be able to operate during charging.
- Power consumption of the subsystem should be less than 5 Watts at the maximum operation condition.
- Status of the food supply should be observable.
- Battery level for both charging and operating modes should be observable.
- The subsystem should be able to measure the amount of given food.

5..2 Subsystem Description

In this subsystem, all electronic components are connected and powered up for the use of other subsystems. The subsystem contains batteries, voltage regulators, battery chargers, relays, diodes, a microcomputer, a microcontroller, and a servo motor. Signal flowchart of this subsystem is shown in Figure 22and flowchart for the power supply units is shown in Figure 26.

1. Supply Unit

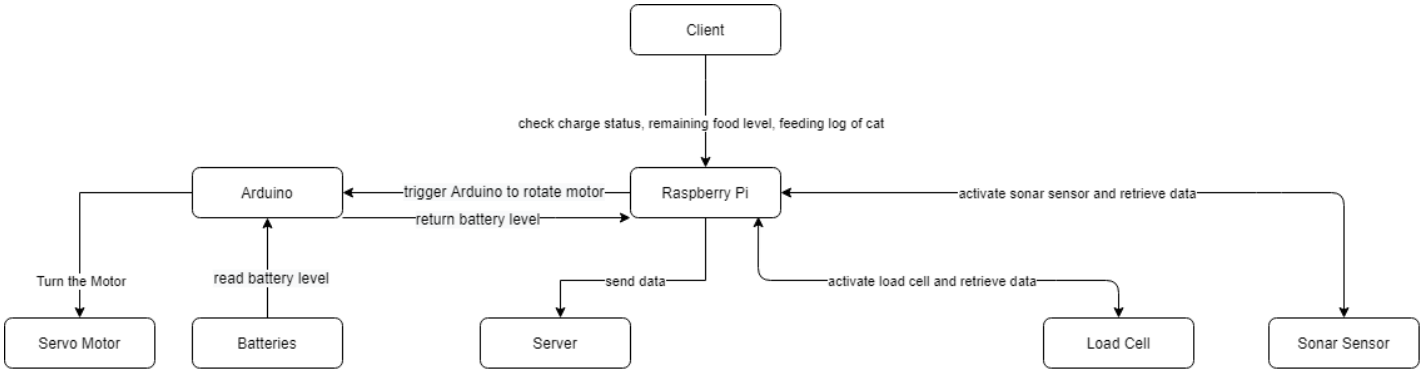


Figure 22: Flowchart of the Electronics Subsystem

Supply connections for the motor and controller units are shown in Figure 23. Two separate battery units are designed for motor and controller units to obtain better motor performance.

High motor torque is required for the rotation of the mechanical gate. 2 Lithium-Ion Batteries are connected in series to obtain a voltage in the input voltage range of the MG996R servo motor. Parallel and series combination of diodes are added at the end of this unit. Parallel diodes increase the maximum passing current of the node, and serial diodes are used to keep the voltage in the input range. Relays are added to decrease power consumption and increase charging speed. The relays RL1 and RL2 shown in 23 are used for parallel charging operation while the outer box is connected to the power outlet with a micro-USB adapter. Also, while the box is not plugged in, the batteries become serially connected, as explained above. However, in this way, the motor is not able to operate during the charging operation. Another relay RL3 is added to solve this problem. The power of the motor is supplied directly from the power outlet during the charging operation, and it is supplied from the batteries while the box is not plugged in. Furthermore, the relay RL4 is connected to decrease power consumption. It is controlled from the Raspberry Pi and turns on the motor when a cat is detected.

For Raspberry Pi and Arduino, a constant stable voltage input is required, so a 5V step-up voltage regulator is used. Two Lithium-Ion batteries are connected in parallel to increase usage time, and the batteries are directly connected to the step-up voltage regulator. Both Raspberry and Arduino are supplied from this regulator.

For charging operation, three different charger circuits are connected parallel, as seen from Figure 23, and all batteries are charged up synchronously.

Also, two resistors are used to read analog voltages from the Arduino.

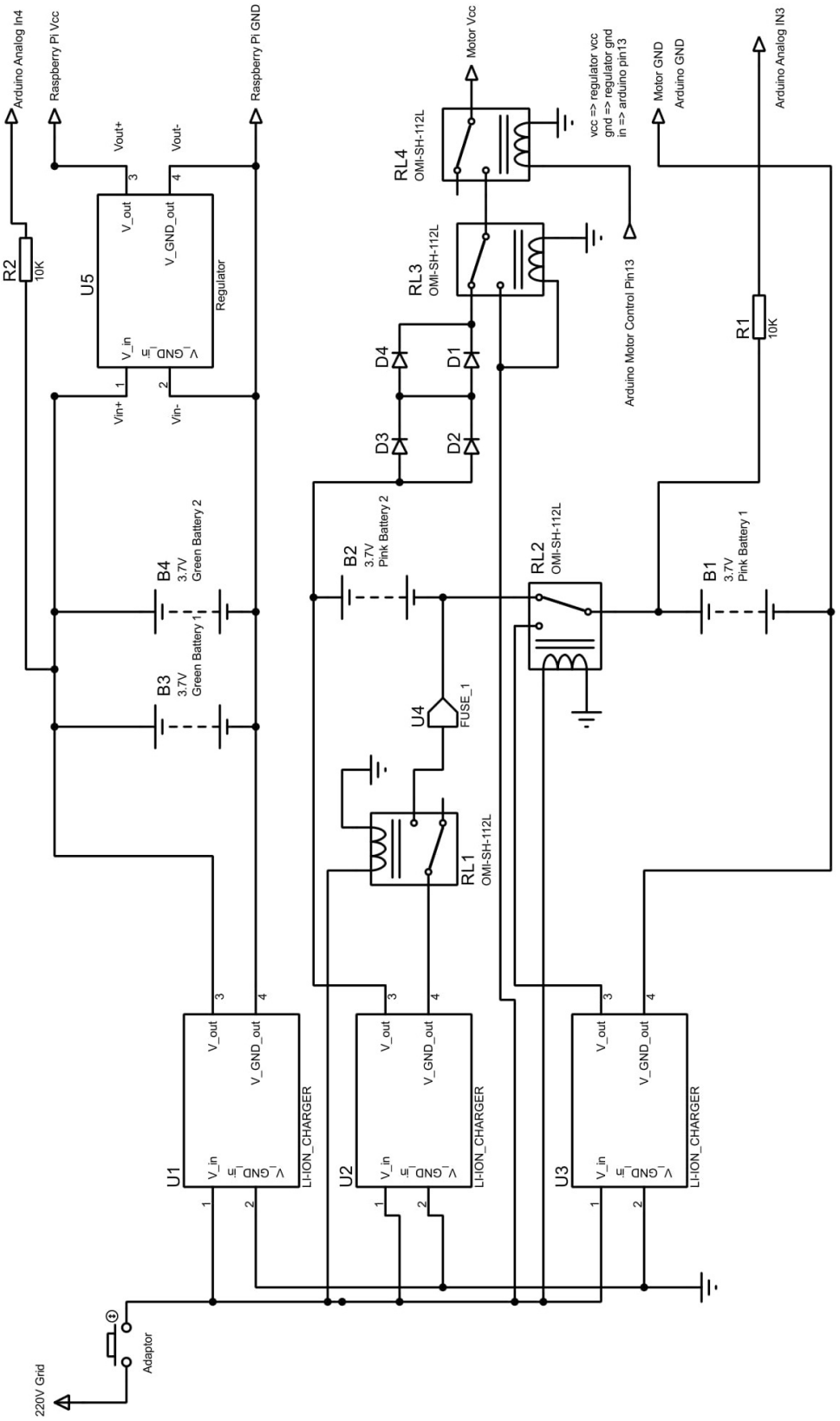


Figure 23: Circuit Design of the Supply Unit

2. Raspberry Pi and Arduino

   Raspberry Pi is the primary control mechanism of the project, and its operations are listed below.

   - Receiving images from the camera module and transferring them to the server.
   - Controlling the motor rotation by sending a control signal to Arduino through a serial communication channel, and this command opens the Relay 5 shown in Figure 23 and generates a PWM signal.
   - Measuring the distance coming from the ultrasonic sonar sensor, converting the measurement to volume, and sending it to the server.
   - Measuring the weight of the given food and sending the data to the server.
   - Receiving the analog battery voltages from the Arduino by using the I2C serial communication protocol to indicate battery level in the web interface.
   - Controlling the LEDs which indicate the detection of cats/dogs.
   - Controlling the operation of the light bulb according to the time.

   Arduino is the second controller used in the project, and its operations are listed below.

   - Controlling the servo motor by generating PWM signals when rotation command is received from Raspberry Pi.
   - Receiving the analog voltage data from the batteries and transmitting the data to the Raspberry Pi.

3. Servo Motor

   Servo motor rotates the food gate according to the amount of food calculated for each cat, and it is controlled by changing the duty cycle of the PWM signal generated in Arduino. The motor unit is explained in more detail in the mechanical design part of this report.

4. Measuring the Food Level

   The users are informed about the remaining food level in the reservoir. The food level is measured by the ultrasonic sonar sensor HC-SR04, which measures the distance between the food and sensor. The sensor generates an ultrasonic sound around 40kHz, and the amplitude of this sound is very low in order not to disturb the animals. The sensor contains two different transducers. The first transducer generates the sound wave, and the second transducer receives the echo of the transmitted wave. The following formula calculates the distance:

$$\text{distance} = \text{time} \times \text{speed of sound} \,/\, 2 \tag{13}$$

   The generated sound is transmitted from the transducer, and then the wave bounces back from the object; the returned wave is received from the second transducer. The distance is divided by two since the sound makes a round trip between the sensor and the food.

Sound speed is taken as 340 meters per second. The measured distances are proportioned with respect to the reference level, and volume measurement is obtained.

5. Measuring the Weight

   A weight sensor is used in order to obtain the amount of food given to the cats. The sensor has three input wires. The sensor returns analog voltage values with respect to the applied force on the sensor. This analog voltage should be converted into digital data. An analog to digital converter HX711 integrated circuit is used, and the digital data is read through a port in the raspberry pi and scaled according to the measured weights.

5..3 Test Results and Discussion

1. Test Procedure

   The test procedure is explained below.

   - Measure the voltage of batteries.
   - Charge the batteries with charger circuits until the voltages are balanced.
   - Connect batteries to regulators.
   - Measure the output voltage of regulators for different voltages inside the voltage range of the battery.
   - Connect relays.
   - Power up the Raspberry Pi, Arduino, and the motor.
   - Connect all sub-units properly.
   - Measure the distance with the sonar sensor.
   - Measure the weight of the given food.
   - Drive the motor.
   - On-Off control of the light bulb.
   - Obtain images from the camera.
   - Measure the maximum and minimum camera view distance.
   - Drive the motor while the camera is active.
   - Measure the distance while the motor and camera are active.
   - Measure the weight while distance sensor, motor, and camera are active.
   - Measure the currents for power analysis.

2. Test Results

   - Li-Ion batteries operate properly. Output voltage is between 3.7V and 4.2V.
   - Parallel charging for two different power supply units with 3 chargers operates correctly.

- Batteries work while charging. The subsystem requirement for rechargeability is satisfied.
- 5V 1.2A regulators return 5.35Volts when the input is 3.7V and 5.45 Volts when the input is 4.2V. This is undesirable; however, it does not cause any faults or performance loss for raspberry pi and Arduino operations.
- Relays work correctly.
- All connections are done on a board and placed in a fixed layer. For charging operation, a micro-USB socket is placed outside of the box. Thus, the subsystem requirement for non-removable batteries is satisfied. Also, solders and connectors are used in wiring, and cable connections are optimized and hidden.
- Raspberry Pi, Arduino, and Motor operate without a performance loss.
- Measured distances are correct if the distances are 2 centimeters away from the sensor, and results are shown in Figure 24.

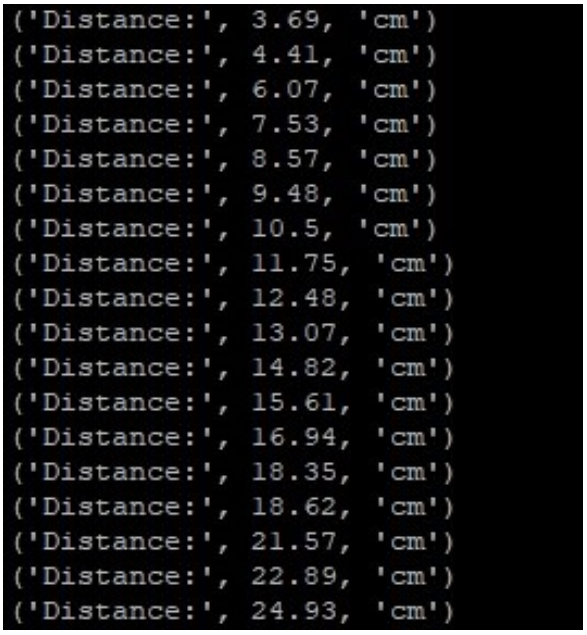
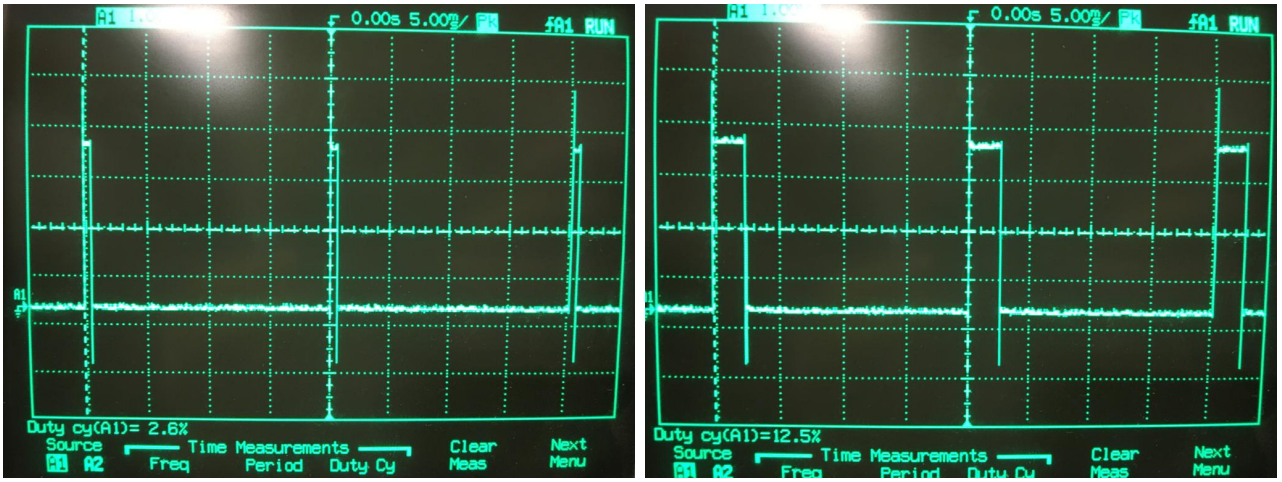


Figure 24: Measured Distances From Ultrasonic Sonar Sensor

- Weight sensor does not read small values. A better resolution sensor will be implemented.
- Motor is rotated according to the entered amount of food properly. Generated PWM signals can be seen in Figure 25.
- High-quality images are obtained.
- A dog can be detected from the taken image from the camera with a distance range of 10 cm to 180 cm away from the box. The average detection distance is 110cm.
- A cat can be detected from the taken image from the camera with a distance range of 15 cm to 153 cm away from the box.



(a) PWM Signal Duty Cycle = 2.5 (b) PWM Signal Duty Cycle = 12.5

Figure 25: Generated PWM signals

- The system operates properly while the motor and all sensors are working. The jittering issue mentioned in the conceptual design report is solved.
- The battery unit draws 0.84 A when they are fully discharged. Charging and operation times are mentioned in the power analysis section of the report.
- Currents are measured and written in the power analysis section of the report. The drawn currents for the units are reduced by using relays, and power consumption is decreased.

5..4 Power Analysis

Component	Average Idle Current (A)	Average Idle Power (W)	Max Current (A)	Max Power (W)
Raspberry Pi Zero WH	0.18	0.9	0.42	2.1
Arduino Uno R3	0.04	0.17	0.07	0.297
Servo Motor	0.03	0.1275	0.22	0.935

Table 2: Power Analysis of the Project

The Table 2 shows the power consumption under idle and maximum cases. The value of the idle case is the average over 20-minute measurement. The maximum case occurs while the camera is working. The power distribution diagram is shown in Figure 26.

Weight sensor and sonar sensors and analog to digital converter are supplied from the raspberry pi since they do not require high current.

Battery unit one shown in Figure 12 contains two Li-Ion batteries connected in series, which supplies 7.4 V to the servo motor. The idle current of this unit is 0.03 A, and the maximum current that occurs during the rotation is 0.22 A. The maximum current occurs for 0.2s, and then it returns to its idle case. The lithium-ion batteries used in this unit have 2000mAh capacity. According to mathematical calculations, this unit can operate for 90 hours on the

batteries at its maximum power-consuming operation. However, this time is not tested in the system, and it will be less than 90 hours due to the efficiency of the batteries. However, real-time will be higher than 5 hours. Thus the requirement about the operating time of 5 hours is satisfied.

Battery unit two shown in Figure 12 contains two Li-Ion Batteries that have 2650mAh capacity connected in parallel. Hence, the overall battery capacity is 5300mAh, and the unit supplies both Raspberry Pi and Arduino Uno. Raspberry pi draws 0.18 A average on the idle case and 0.42 A on the maximum case. The maximum case occurs when the camera module is activated. Also, Arduino draws 0.04 amperes average on the idle case and 0.07 A on the maximum case. Hence, the maximum current driven from the supply is 0.49A, and the system is able to operate on the batteries for at least 10 hours. Therefore, the requirement for operation time is satisfied.

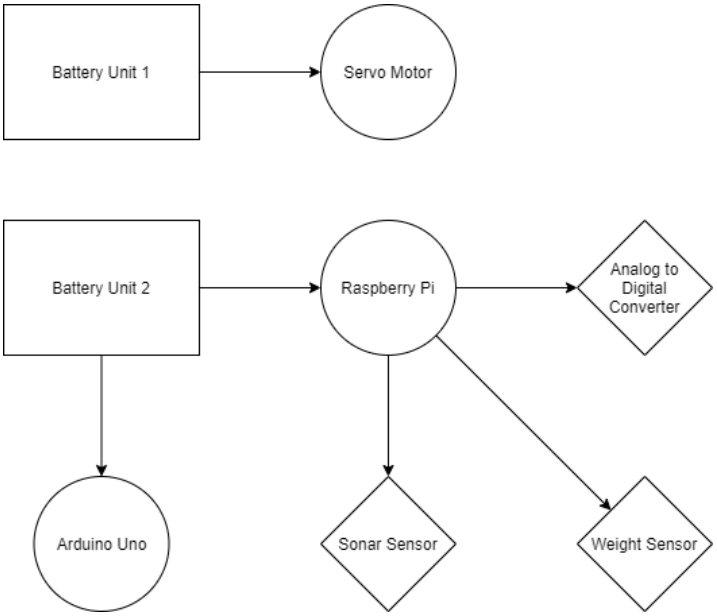


Figure 26: Power Distribution Diagram of the Project

3 Change of plans

a) Remotely completed subsystems

Remotely completed systems, their tasks and system descriptions are given in this section. The test results and discussions are included in the section 4 "Termination of the Project".

1. Data Set

Database is the collection of instances and we created a cat database, which can be found from this link. Our dataset includes 150 pictures of 30 different cats each taken from different angle and distance from the screen. It does not include images of real cats due to the Covid-19 pandemic. Some of the pictures are blurry to test different conditions for the identification algorithm.

Also, since the original photographs include the background of the screeen, they are cropped by using an algorithm, and the unwanted parts of the screen, excluding the cat, are removed. The cropped images can be found from this link.

The overall dataset is divided into three datasets, which are train, validation, and test datasets.

- The training dataset is the data used for the identification algorithm to learn the parameters of cats.
- The validation dataset is the sample data used to provide an evaluation of the model and tune the hyperparameters of the identifier. The parameters defined during the training are tuned based on the results of the validation set.
- The testing dataset is the dataset used to assess the performance of the specified identifier by the previous datasets. The reliability of the identifier and accuracies of the results are analyzed in this step.

The dataset splitting is crucial for the optimization of the system. In short, the system is trained with the training dataset, optimized by checking the performance on the validation dataset, and its performance is tested by an independent testing dataset.

An algorithm is written to create train, validation, and test datasets for different inputs. This algorithm reads a text file called dataset.txt shown in Figure 27, and automatically generates the datasets. In this algorithm some cats are excluded to test different scenarios. The numbers for metadata2 in Table 3 are explained below.

- Train 20 - 30 means use images from image 20 to image 30 for each cat except the excluded cats for training operation.
- Validation 45 - 70 means use images between image 45 and image 70 for each cat except the excluded cats for validation operation.
- Test 89 - 97 means test the system with images from 89 to 97 for each cat except the excluded cats.
- Excluded 2,12,7 means the cats labeled as cat2,cat12 and cat7 are not included in this train-validation-test cycle.

.

```
10#
Train
20
30
Validation
45
70
Test
89
97
Excluded
2
12
7
```

Figure 27: Dataset.txt file

Five different metadata files are used for the identification algorithm of the project, and these files have different datasets for the training, validating and testing steps, and some datasets are excluded from testing different scenarios. The content of these metadata files are given in 3. The numbers are randomly select to observe different scenarios. These numbers directly affect the results of the identifier, and these effects are explained in the Results of the new system section of the report.

	Train	Validation	Test	Excluded
metadata2	20-30	45-70	89-97	2,7,12
metadata3	12-15	49-81	82-135	1,17
metadata4	21-23	11-41	53-116	2,11,15
metadata5	2-3	45-69	70-127	2,9,16
metadata6	23-28	50-60	1-16	2,8,11

Table 3: Content of the metadata files

We used more images for testing datasets than for the training datasets in these five different scenarios to obtain more reliable results. Also, the reason for the less training set is that we do not have an extensive database, including more images of cats. The results of these tests are presented and analyzed in the Results of the new system section of this report.

2. Identification

There were several suggestions for the identification problem such as histograms, color spaces, feature descriptors, and neural networks for the identification process. Finally, feature descriptors are selected as they satisfy the requirements. SIFT and ORB descriptors are implemented and identification works based on the idea of vector distances between the descriptor vectors. This sub-section presents the algorithm, general overview of the identification methodology and its properties.

2..1 Algorithm

The algorithm is straightforward. It consists of three stages. In the first stage the database is created based on the train images. On the other hand, each vector is compared with each vector in the database using matches. The final stage is to optimize database.

In the first stage, vectors created from images are accumulated in each identity of cat. Then all of these vectors are saved into the database that contains all cats. For example, there are 3 cats and 2 pictures for each cat. Assume 5 vectors for each image. There will be $3 \times 2 \times 5 = 30$ vectors in total. Total of 3 sets are going to contain $2 \times 5 = 10$ vectors. These vectors are then matched with each other. Let's say they are 3 vectors for each image that matches with the other image in the same identity. Then, there are $3 \times 3 = 9$ vectors in total and 3 vectors for each identity. The sets are for each cat, and vectors contain all matched vectors from all images of that cat.

In the second stage, the algorithm continues with matches. A match is simply defined as the nearest vector in the database to the reference vector. Reference vector is the vector from source image where we want to label. For the matching algorithm, Lowe's paper is taken as reference and FLANN is used for feature matching algorithm.

In the light of this information, a pseudo-code is given. In 2, k is a parameter whose value is 5. In addition, dynamic threshold mentioned in the 3 is calculated in the database creation algorithm based on the data itself. The calculation is out of scope of this report.

```
initialize empty database or load existing database ;
for identity in all cat identities do
    extract feature descriptor vectors;
    for image1 in that class do
        for image2 in that class but not image1 do
            find matches between image1 and image2 using FLANN ;
            add qualified matches to database.identity ;
        end
    end
end
```

Algorithm 1: Algorithm for Creating Database

```
load database ;
load source image ;
for identity in all database identities do
    extract feature descriptor vectors for the source image;
    save the match distances and identity ;
end
find k best matches ;
return the identity that has the most frequent occurrence in that k matches ;
```

Algorithm 2: Algorithm for Identifying Cat

1. Eliminate similar vectors in the same class
2. Eliminate similar vectors in different identities


```
load database ;
for identity in all database identities do
    for vector1 in database.identity do
        for vector2 in database.identity but not equal to vector1 do
            if norm of (vector1 - vector2) is smaller than a dynamic threshold then
                eliminate vector1 ;
            end
        end
    end
end
for identity1 in all database identities do
    for identity2 in all database identities but not equal to identity1 do
        for vector1 in database.identity1 do
            for vector2 in database.identity2 do
                if norm of (vector1 - vector2) is smaller than a dynamic threshold then
                    eliminate vector1 ;
                    eliminate vector2 ;
                end
            end
        end
    end
end
```

Algorithm 3: Algorithm for Optimization Database

2..2 Working Principle

The algorithm and implementation are summarized in this part. The module performs the following operations:

- Creating database from existing photos
- Identifying a cat that is in database
- Recognizing new cats that are not in the database
- Optimizing database
- Database operations - save / load / compress

There are methods that provide these functionalities. Some of them were written before and some are added and modified in the quarantine period. The new ones are briefly explained.

- Identifier.optimizeDatabase: Optimization code that runs on parallel and provides acceleration for the algorithm.
- Identifier.addNewCat: Add new cat dynamically to the database, single frame cat identification is provided here.

Pre-existing ones are given briefly. Note that all of them are explained in detail in code. However, for better realization, their overview is included here.

- Identifier.loadDatabase: Loads database (uses compression if compression flag is on)
- Identifier.saveDatabase: Saves database (uses compression if compression flag is on)
- Identifier.importDirectory: Creates database by consuming data from directory.
- Identifier.resetDatabase: Resets database by deleting it permanently.
- Identifier.getCatId: Returns the cat identity.

3. User Interface

Even though this subsystem did not undergo any change of plans, it belongs here solely for the reason that it has been completed and integrated remotely.

The features that were introduced during the lock-down are listed below, followed by a brief explanation for each feature.

1. A power toggle for devices
2. A charging indicator
3. Food percentage indicator
4. Time of last refill
5. Customizable night light on/off times
6. Cat feeding logs in calendar view
7. Customizable cat food amounts

Power toggle: This is how the user can turn his/her device(s) on/off.

Charging indicator: Previously, only the charge was displayed. Now, the user knows whether the device is being charged or not.

Food percentage: The user will now be able to see how much food is remaining in the tank.

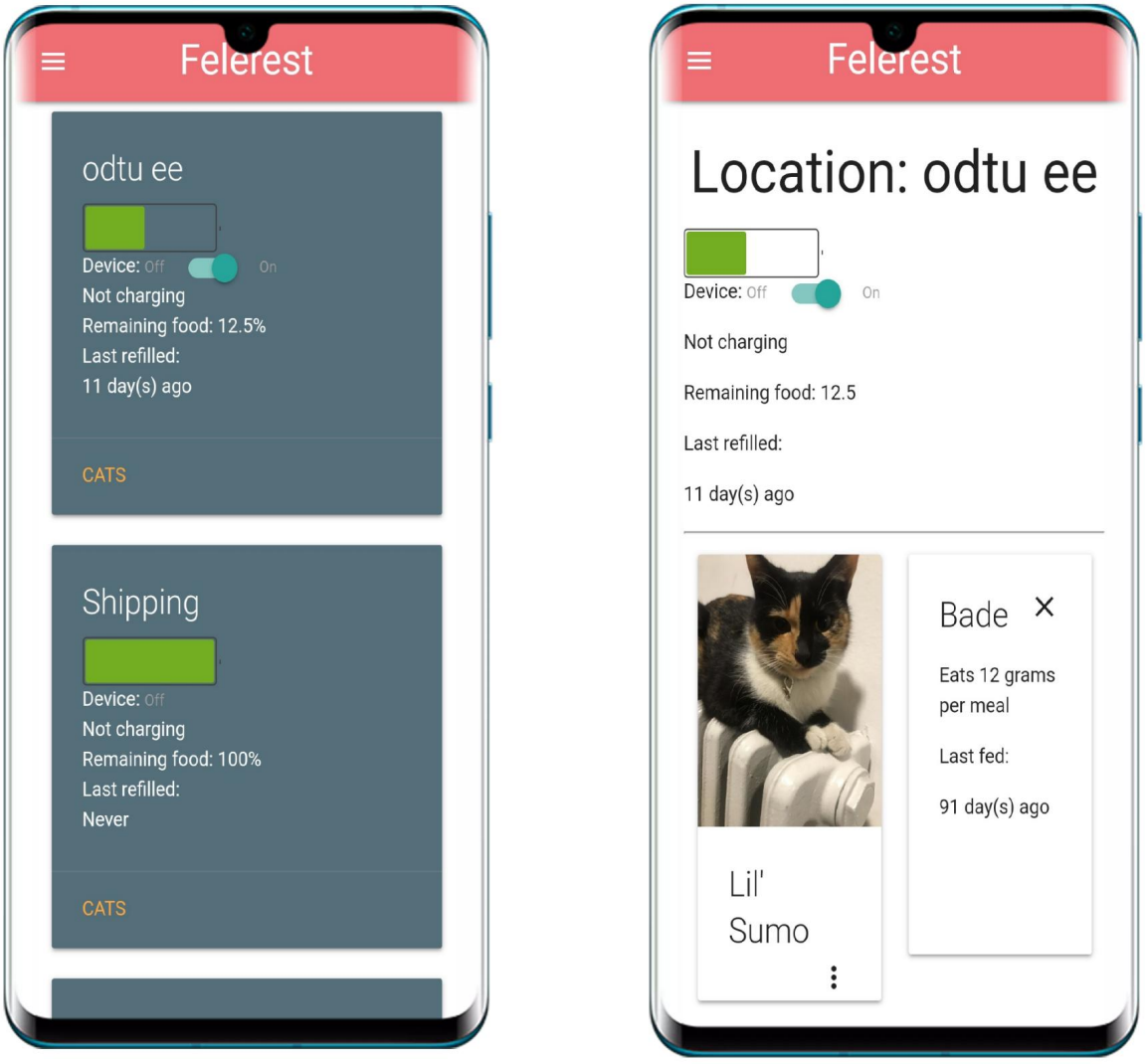
Last refill: Time of the last refill of the food tank will be displayed.

Light on/off times: There are lights on the device that allow it to work at night. When these lights are on/off can be customized by the user.

Feeding logs: The feeding logs of the cats are put in exquisite and detailed calendars.

Food amounts: The user can decide how much each cat gets to eat per meal.

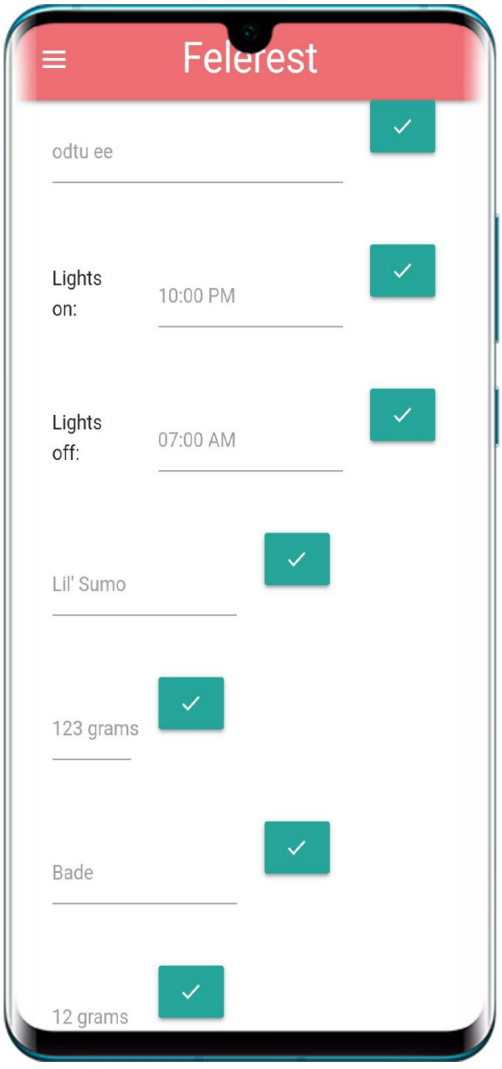
The new screenshots of the website with the added functionalities follow:



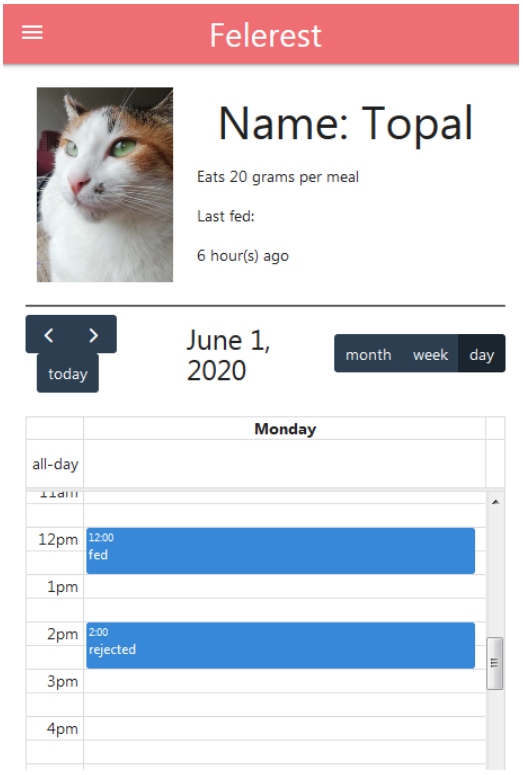
(a) New User Devices Page

(b) New Device Cats Page

Figure 28



(a) New User Account (Settings) Page



(b) Cat Feeding Log Page

Figure 29

b) Unfeasible subsystems

The mechanical and electronics subsystems are the ones that are affected by isolation situations. The inner design is made before the school break which can be seen in the current status section. The only remaining part for mechanical design is the outer box. This, however is not implemented but the 3D drawings are done and ready to be implemented as soon as pandemic finishes. It is planned to implement the outer box using epoxy. All the plans have done including the appointment with the qualified person which will cut the epoxy, however, due to the Covid-19 pandemic it is not developed as it planned, therefore the implementation of the outer design is postponed and eventually canceled. However the drawing is ready for the next group that will handle the mechanical design after the pandemic.

In the electronics parts all of the electronics components are ordered and most of them are connected to the inner design. The unconnected electronics components can be seen below.

- Sonar sensor for remaining food level: Sonar sensor is implemented, and it is operating as required as described in the current status section. However, since the box is not completed due to quarantine, the measured distances cannot be converted into volume data.
- Weighing the food: The weight sensor is implemented, but it does not read the small values. A better sensor is bought. However, before it is implemented, the quarantine has begun.
- Lighting: It is planned that our system will work on night. Therefore lighting is used in order to detect the cats although there is no natural lighting.

Some of the inner connections of the electronic devices require the outer box. Since the implementation of the outer box is delayed, these connections are not established. These connections can be seen below.

- Connection between Arduino and Raspberry Pi
- Connection between Arduino and batteries

In the absence of the outer design and the connections between electronics, the integration of all subsystems is not possible. Therefore the integration of mechanics and electronics with the classification and identification is not done since it requires human interaction.

c) Alternative solutions for uncompleted subsystems due to isolation conditions

Due to the Coronavirus pandemic, the outer design is not realized. Since the outer design is not realized the weight sensor and lighting are not tested. It is planned that the outer design will be realized when pandemic finishes. Therefore, in this period the drawing of the outer design is improved. Theoretical calculations are done. Everything was ready and waiting to be implemented. However pandemic did not finish on time. At this point, our plan as Felerest is to work on classification, identification, and web design and delaying the mechanical and electronics design which can be implemented without danger.

d) Remotely integrable subsystems

There are three subsystems that can be integrated without any physical work. They are,

1. Identification Module - Classification Module (named as CV Module)
2. Decision Making Module - CV Module
3. Decision Making Module - Web Interface Module

Decision making is the base module that creates the backbone of the server. Action to be taken such as feeding cat or deterring dog, food amount etc. are determined under this module. This module also a caller for the Identifier and Classifier, and it merges the modules. For the identifier - classifier relation, it is straightforward and depicted in figure 30. On the other hand, connection of web interface requires abstractions over objects such as Cat, Device, Server.

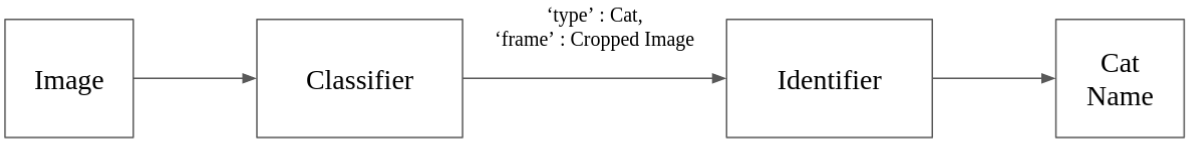


Figure 30: Classifier Identifier Connection

e) Breakdown of the work and responsibilities

1. Asude Aydin

- Classification algorithm
- Help in mechanical implementation
- Organizing electronics board
- Promotional video
- Demo video

2. Fatih Eser

- Server Client
- Identification Algorithm
- Interfacing between subsystems
- Tests in classification and identification
- Demo video content (classification & Identification)
- General help in mechanical implementation

3. Furkan Aldemir

- Initial work in identification
- Web design
- Help in demo video content (web design)

4. Doga Tolgay

- Entirety of mechanics (design & implementation)
- 3D drawings of mechanical subsystem
- Help in organizing electronic board
- Promotional Video
- Demo video

5. Utku Serhan Suicmez

- Entirety of electronics
- Mechanics and electronic subsystem's algorithm
- Dataset for computer vision
- Help in mechanical implementation
- Logistics (component shopping)

f) Demo Method

As a way of presenting our finished 'Smart Connected Cat Feeding & Monitoring System', we've decided to prepare a demonstration video of maximum 5 minutes. This video will encapsulate how each subsystem works and performs. The content of this video with regard to content's time flow will be explained in detail in the following section.

1. Demo Video

The video will put emphasise on 3 main subsystems. These subsystems are classification, identification, and web design. The server-client relation serves as a backbone and as an integrator between the subsystems in the back-end. This demonstration video's main purpose is as follows:

- The client should be able to understand the underlying technology in an intuitive manner for each subsystem.
- The client should have a brief understanding of how each algorithm is implemented and how it functions.
- Some of the most important tests will be demonstrated which give an extensive evaluation of the performance of the system. These tests will be conducted on the overall computer vision implementation. This should enable the client to asses how well the system works.

- Some real-life application limitations will be mentioned when test demonstrations are presented. These limitations must be obeyed by the client.
- The website will be presented to the client such that s/he is aware of its abilities, additional features, and its general usage.

In summary, the subsystems will be explained individually but the overall combined system will be put into test and evaluated. The full content is given below in sequence.

1. The classification method and the underlying technology will be explained.
2. The classification part's functioning (algorithm-wise) will be explained briefly.
3. Some general test results will be briefly mentioned to address its performance.
4. The identification method and the underlying technology will be explained.
5. The identification part's functioning (algorithm-wise) will be explained briefly.
6. Some general test results will be briefly mentioned to address its performance.
7. Two main overall tests will be demonstrated to asses computer vision's performance as a whole. These test will test speed and accuracy, as explained in the following:
   - **Speed:** This will evaluate how the computer vision speed is affected when the number of cats in the database is increased.
   - **Accuracy:** This will be a performance evaluation of the additive accuracy of classification and identification. It is the correctly predicted cases with respect to all predictions. So a correct case is as follows: a cat has been correctly identified in addition to being correctly classified.
8. There will be two case studies which will present the system in a nutshell. These cases are as follows:
   - When a new cat is present, can the system identify it correctly from start to finish? First classification will be performed and its confidence and object prediction location will be showed. Then, it will pass through the identification algorithm. It'll be shown that this cat is registered to the database correctly since it is a newcomer. The newly extracted sift vectors will be shown for this part. A timer will count the passed time for the whole procedure during computer vision to asses time required to register a newcomer.
   - When an already registered cat is present, can the system recognize it correctly from start to finish? First classification will be performed and its confidence and object prediction location will be showed. Then, it will pass through the identification algorithm. It'll be shown that this cat was recognized within the database and its new sift vectors and their matching to the database will be visualized. A timer will count the passed time for the whole procedure during computer vision to asses time required to recognize a cat.
9. The website will be shown when a new cat is registered for the first time. The cat will start existing in the website with no data/ blank spaces.

10. The same cat's data will be shown after a period of time passes. Some special user-defined spaces might be filled. Some average diet information will be visible alongside additional features which graph these values.

2. Promotional Video

The promotional video will be approximately 1.5 to 2 minutes. It will be similar to a PowerPoint presentation. The main idea behind this is to advertise our product to people that are unaware such a device with these qualities exist. It will be a take on how we've changed the advertisement of the product after the pandemic. It's not necessary, merely for entertainment and sales purposes.

4 Termination of the Project

a) Results of the new system

1. Identification

The data sets are described in the previous section. This section presents the results for the mentioned data sets. There are different tests and results are going to be presented in this section. More systematically, following results are included. Throughout this section, 5 data sets are investigated which are from the data sets described.

- Identification accuracy
- Identification confusion matrix
- Identification time
- All these results for the optimized code

1..1 Identification Accuracy

Accuracy for measuring the performance of identification process is one of the most widely used metrics in the literature. This is why it is included as the base metric in this project as well. Accuracy calculation is simple and is given by equation 14.

$$Accuracy = \frac{Number\ of\ Correctly\ Identified}{Total\ Identified} \tag{14}$$

The equation 14 gives a general formula which is adopted and adjusted for this project. Two different accuracy results are given. General accuracy, which is calculated as in equation 15, and accuracy based on identity, which is given in equation 16.

$$Accuracy = \frac{Total\ Number\ of\ Correctly\ Identified}{Total\ Identified} \tag{15}$$

$$Accuracy = \frac{Correctly\ Identified\ for\ Specific\ Cat}{Total\ Identified\ for\ Specific\ Cat} \tag{16}$$

The test results are given for train, validation and test sets. Note there were many tests, and giving them as figures make the report unreadable. Therefore, they are uploaded to here.

1..2 Identification Confusion Matrix

Confusion matrix is the representation of the module outputs for each cat. Each cat characteristics are given in this format. Useful information such as false positives can easily be extracted from this data. Note that, confusion plots are huge amount and including them in the report makes the report unreadable. Therefore, the results are given here.

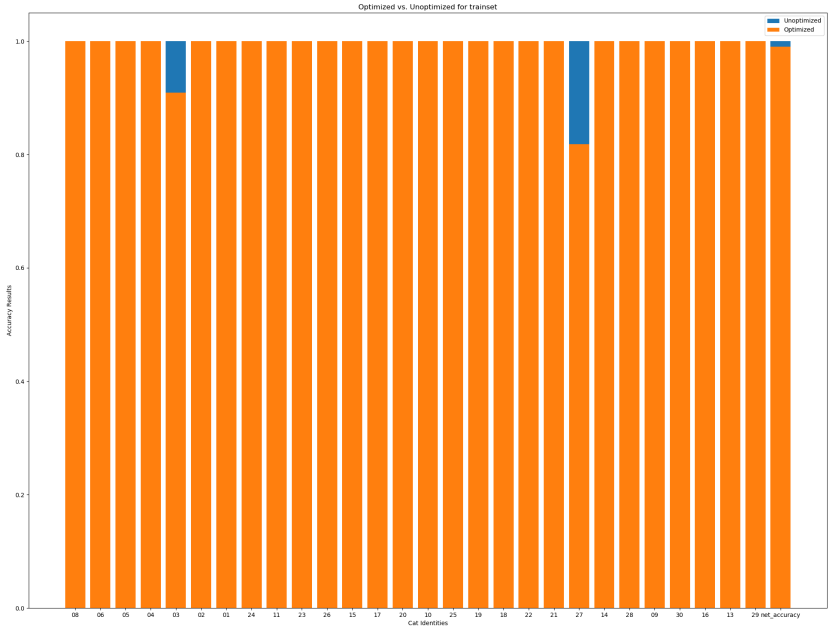


Figure 31: Graphical Accuracy Result for Metadata 2 Train Set

1..3 Identification time

Identification time is another important metric that determines the real world performance of both the system and the Identifier module. As stated in the requirements section, 10 seconds of a threshold is determined with 15 active cats saved to database at the same time. Times are given with each cat basis. Identification time results are also too many. Therefore, they are uploaded to here for better readability.

1..4 Graphical Performance Results

In the previous section, numeric performance results are presented. Finally, graphical results are included in this part for better understanding of the underlying method. The graphical results are directly given since they are not too much and they carry valuable information for the discussion part. x-axis shows different cat identity names, from 01 to 30. y-axis represents the time in seconds that take algorithm to identify a cat, and it represents accuracy for accuracy plots. Orange bars represent optimized results whereas blue bars represent unoptimized results.

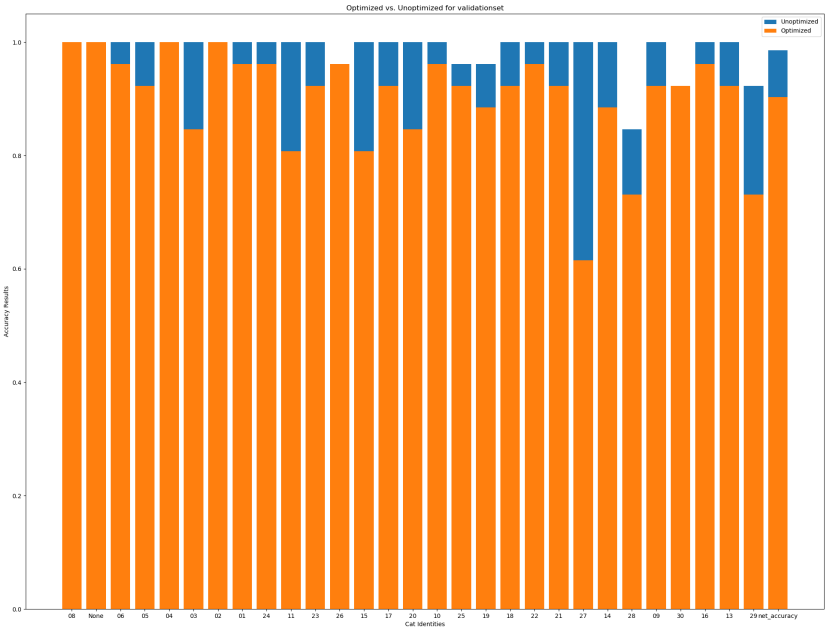


Figure 32: Graphical Accuracy Result for Metadata 2 Validation Set

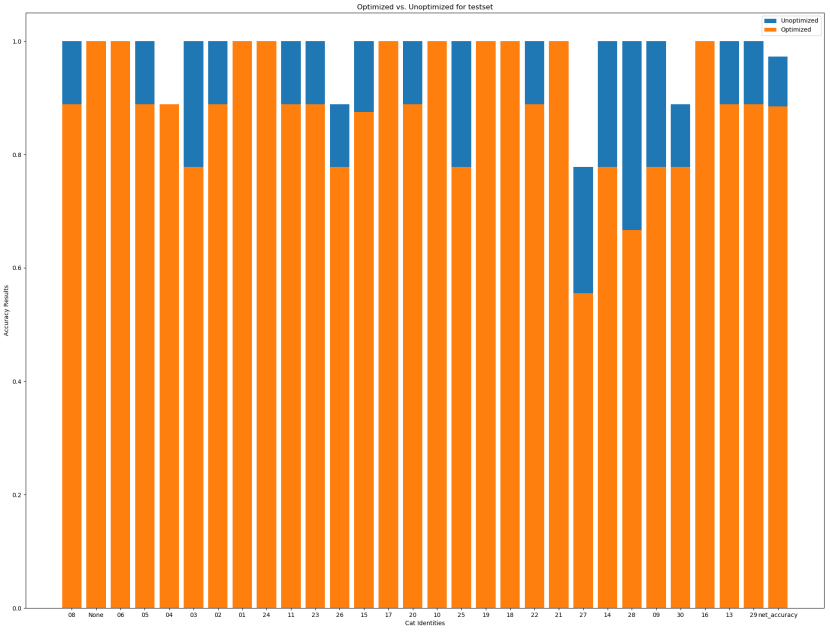


Figure 33: Graphical Accuracy Result for Metadata 2 Test Set

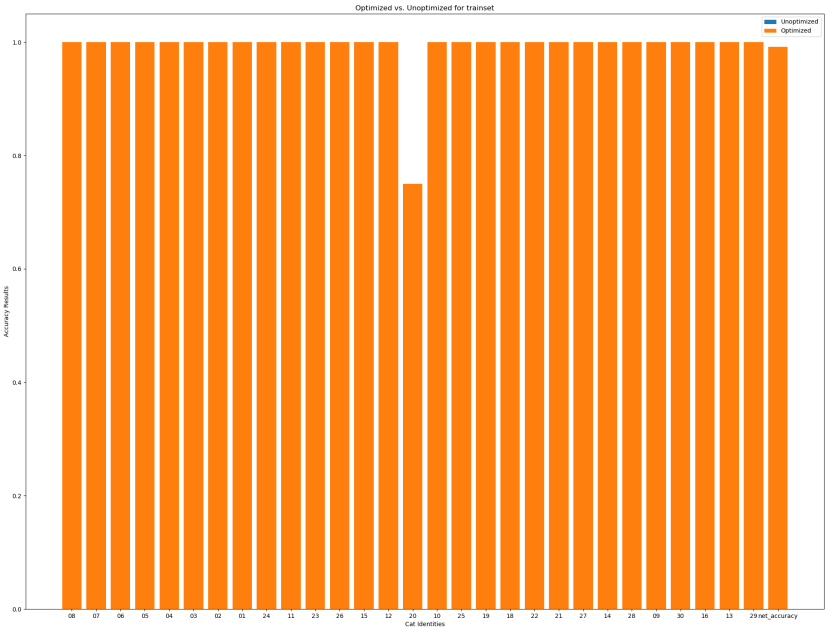


Figure 34: Graphical Accuracy Result for Metadata 3 Train Set

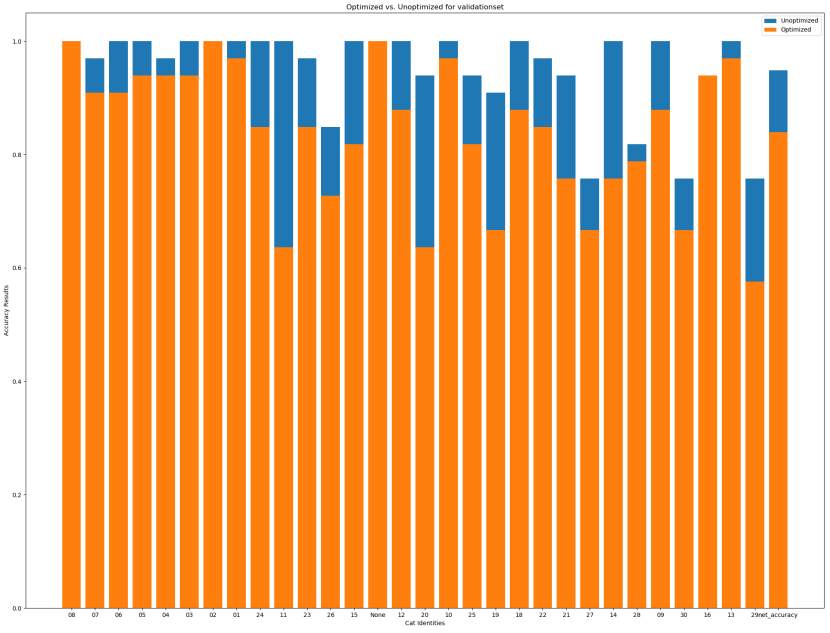


Figure 35: Graphical Accuracy Result for Metadata 3 Validation Set

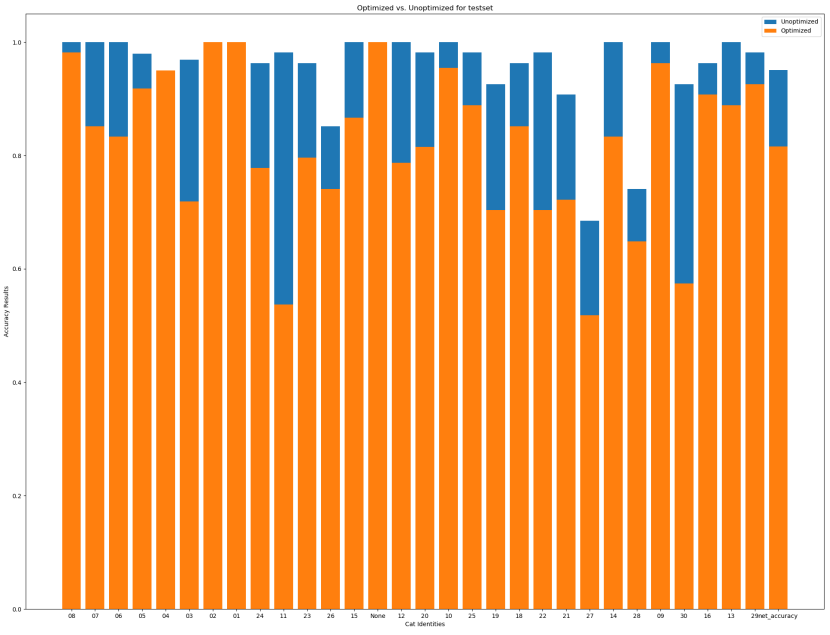


Figure 36: Graphical Accuracy Result for Metadata 3 Test Set

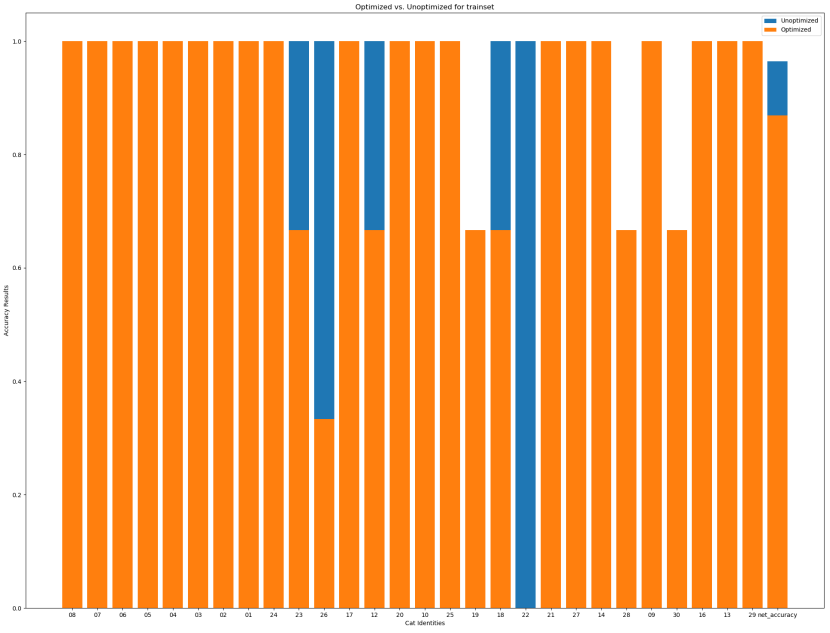


Figure 37: Graphical Accuracy Result for Metadata 4 Train Set

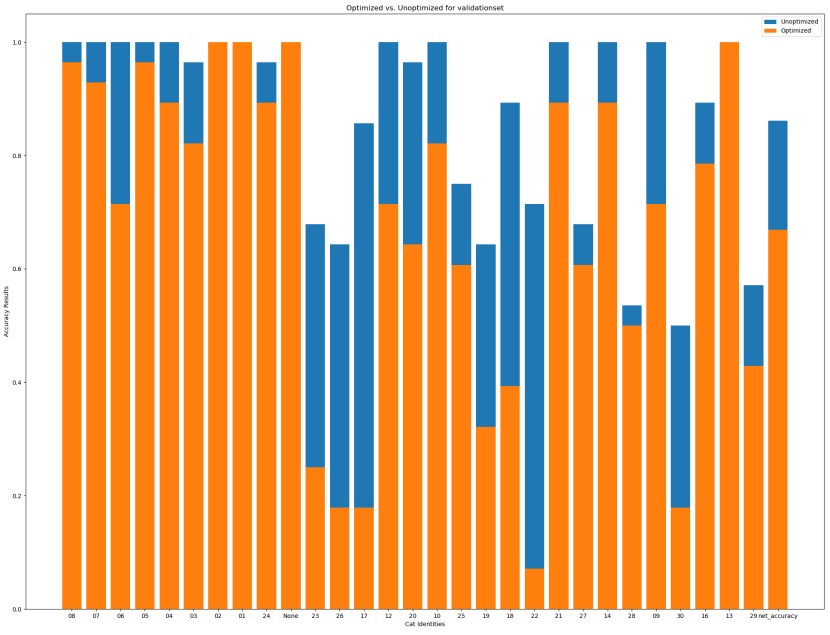


Figure 38: Graphical Accuracy Result for Metadata 4 Validation Set

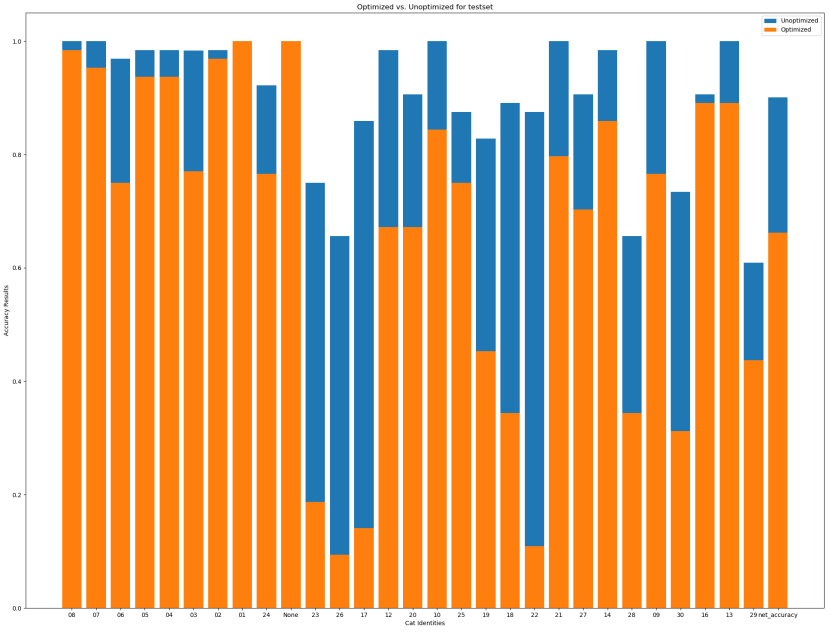


Figure 39: Graphical Accuracy Result for Metadata 4 Test Set

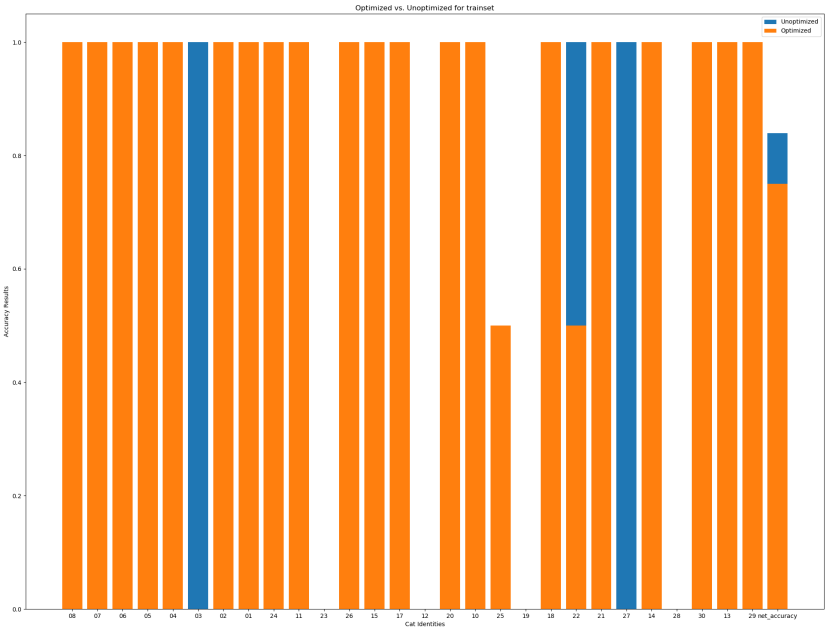


Figure 40: Graphical Accuracy Result for Metadata 5 Train Set

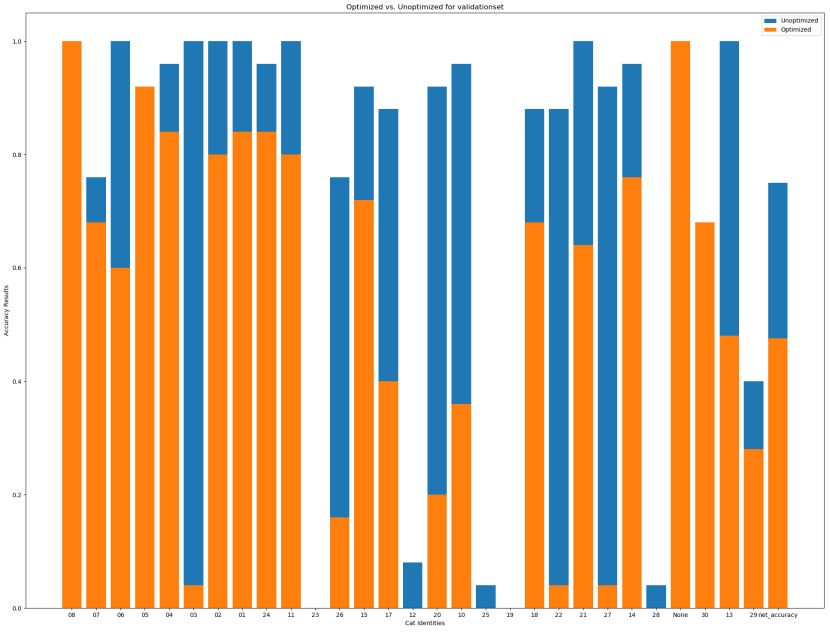


Figure 41: Graphical Accuracy Result for Metadata 5 Validation Set

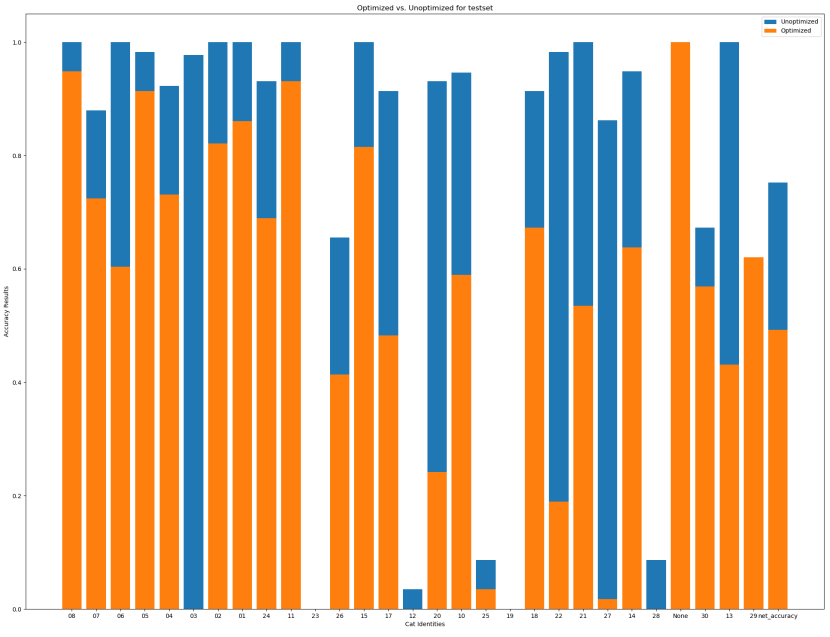


Figure 42: Graphical Accuracy Result for Metadata 5 Test Set

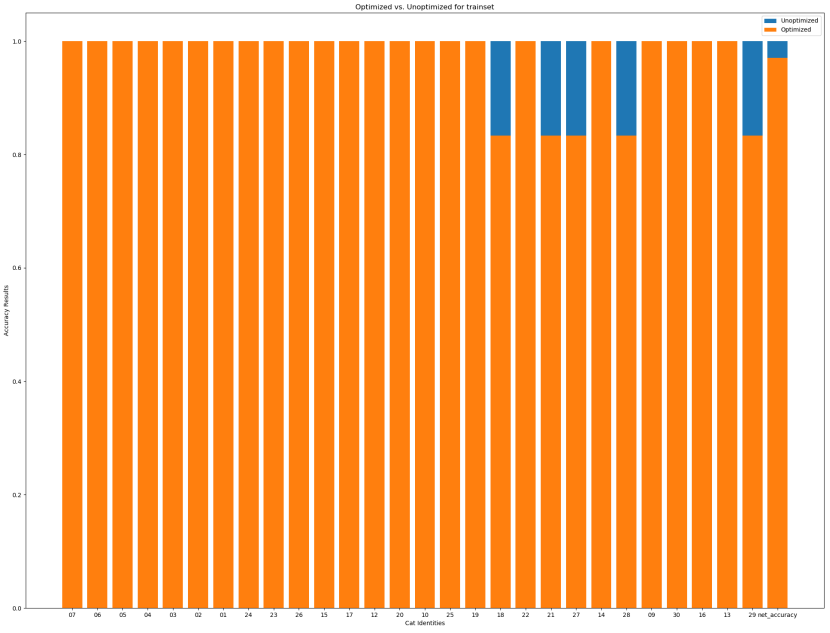


Figure 43: Graphical Accuracy Result for Metadata 6 Train Set

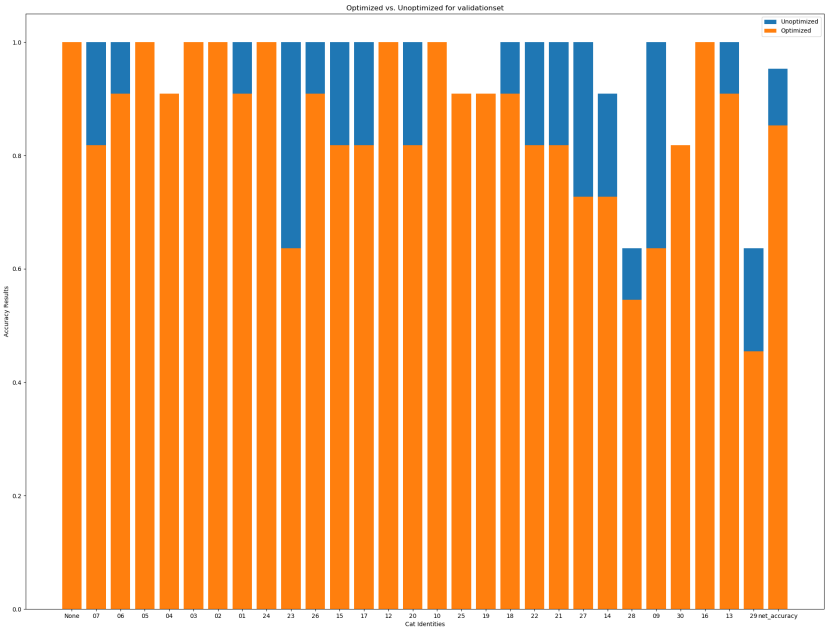


Figure 44: Graphical Accuracy Result for Metadata 6 Validation Set

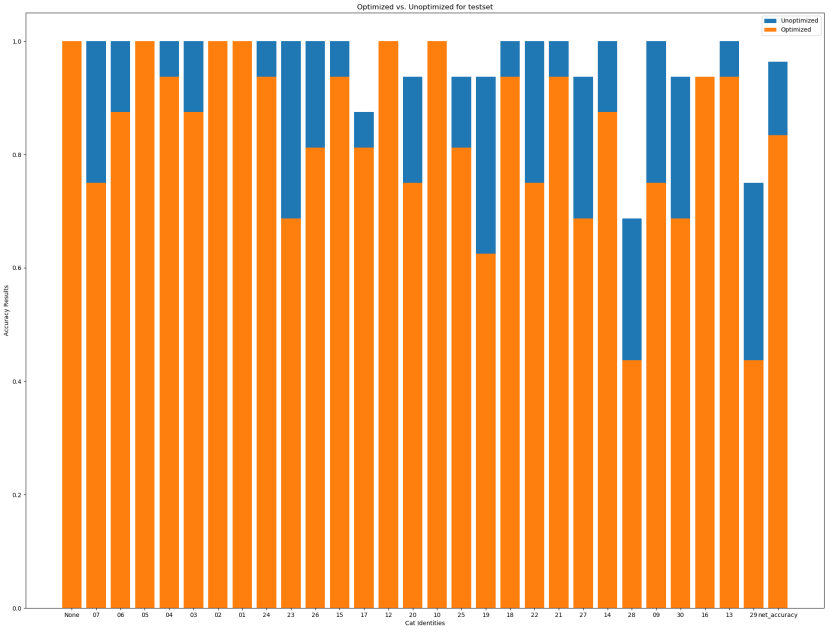


Figure 45: Graphical Accuracy Result for Metadata 6 Test Set

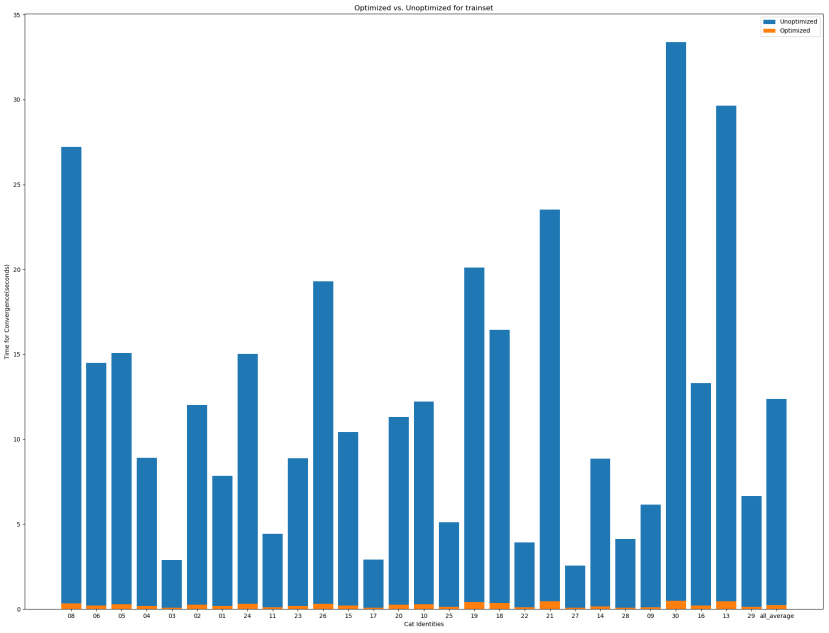


Figure 46: Graphical Speed Result for Metadata 2 Train Set

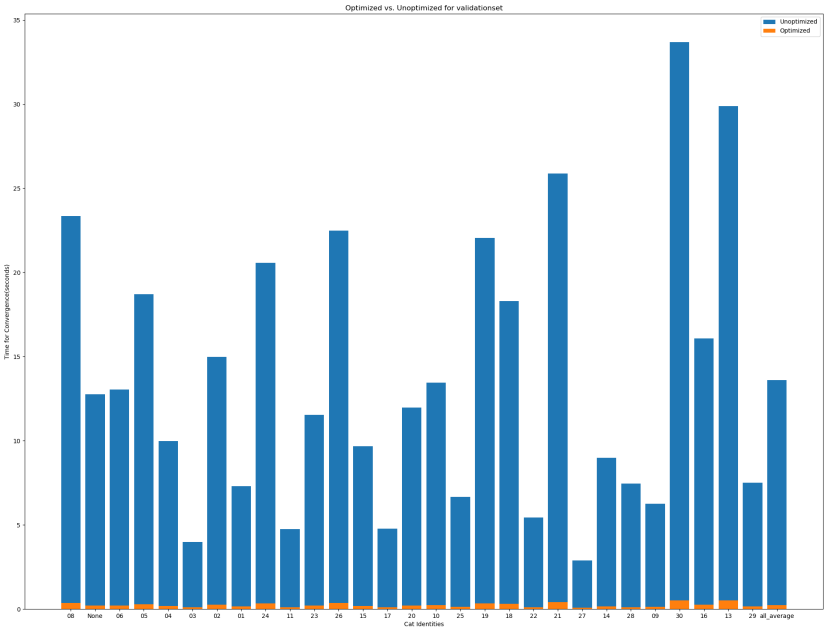


Figure 47: Graphical Speed Result for Metadata 2 Validation Set

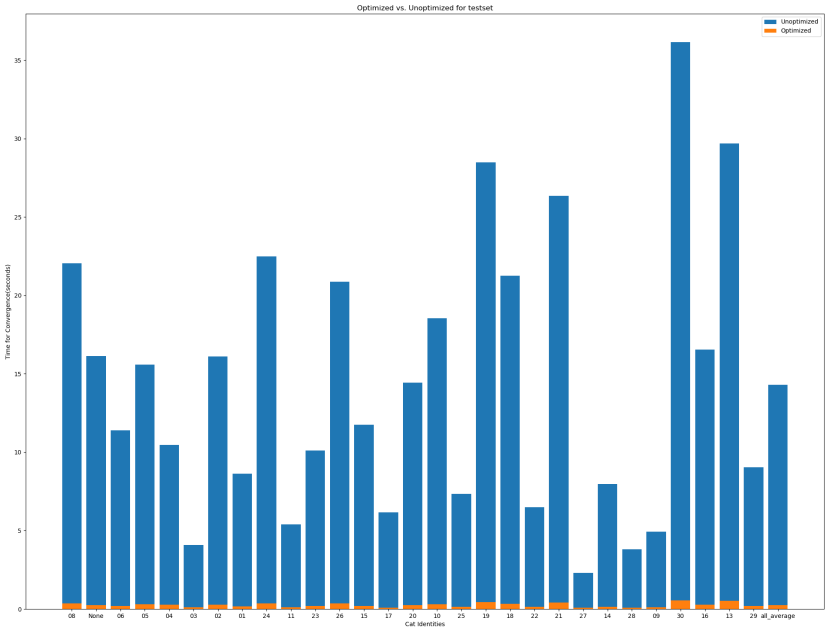


Figure 48: Graphical Speed Result for Metadata 2 Test Set

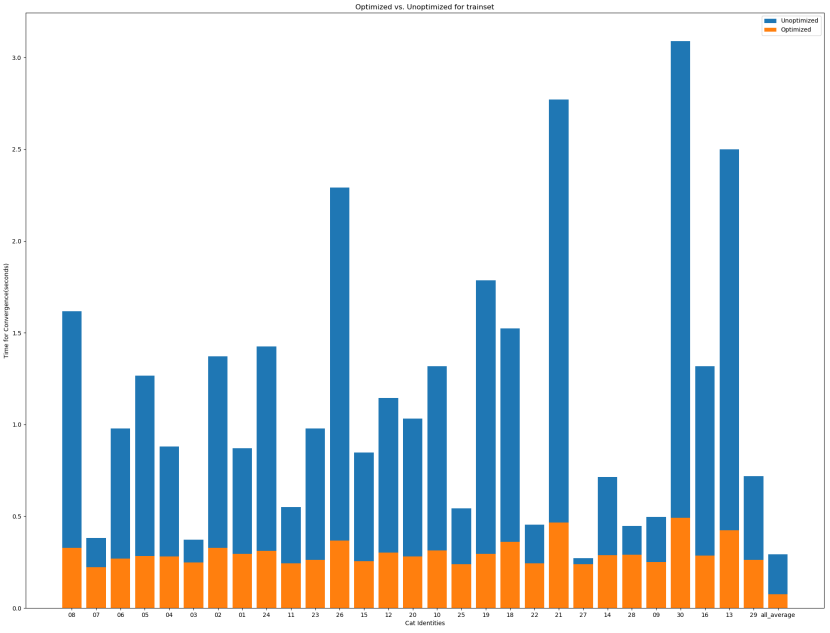


Figure 49: Graphical Speed Result for Metadata 3 Train Set

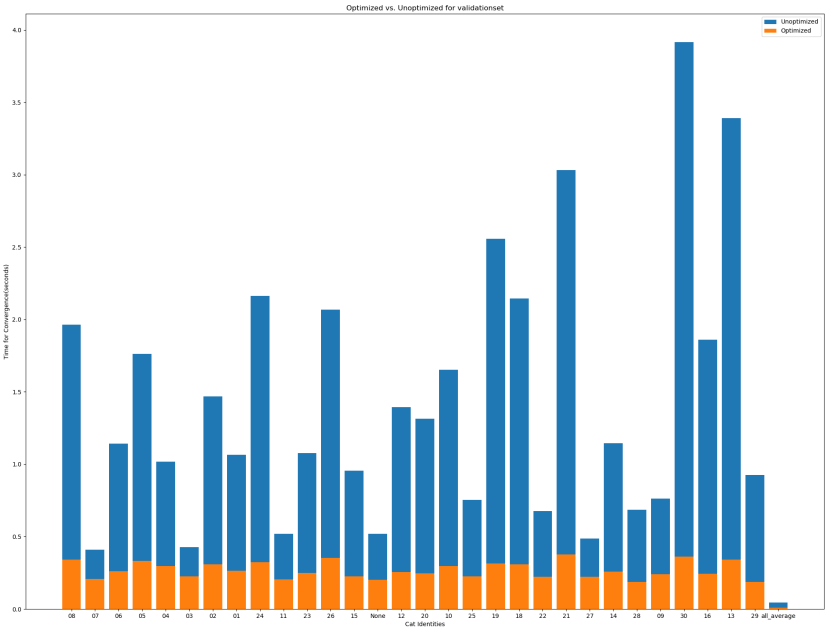


Figure 50: Graphical Speed Result for Metadata 3 Validation Set

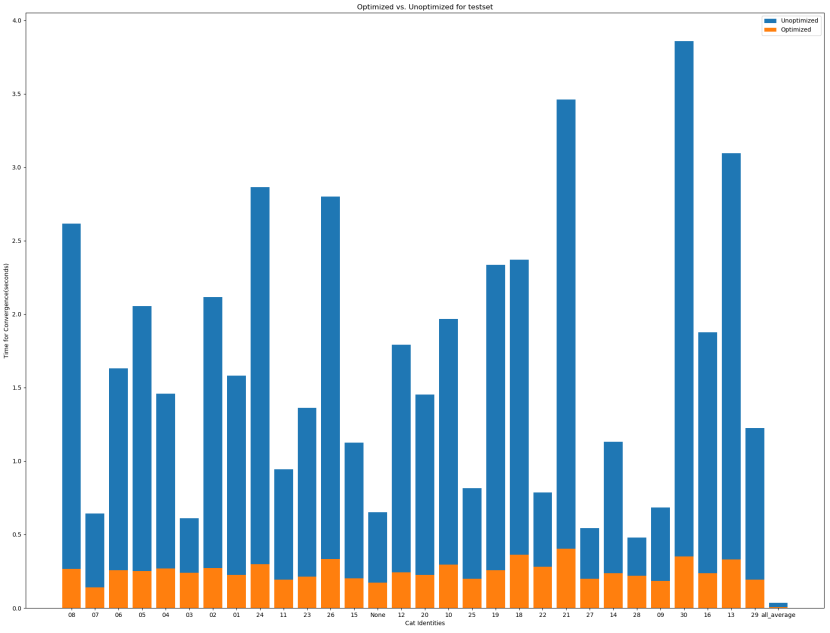


Figure 51: Graphical Speed Result for Metadata 3 Test Set

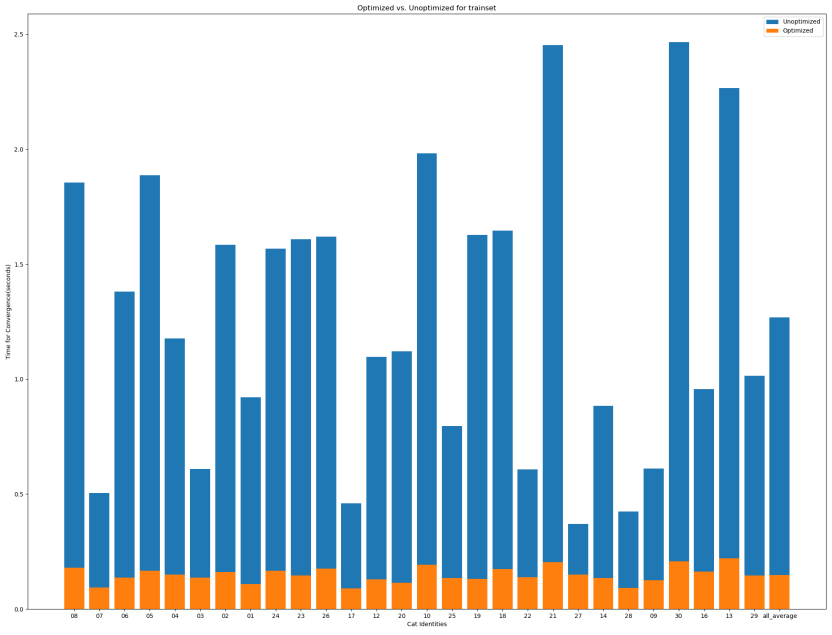


Figure 52: Graphical Speed Result for Metadata 4 Train Set

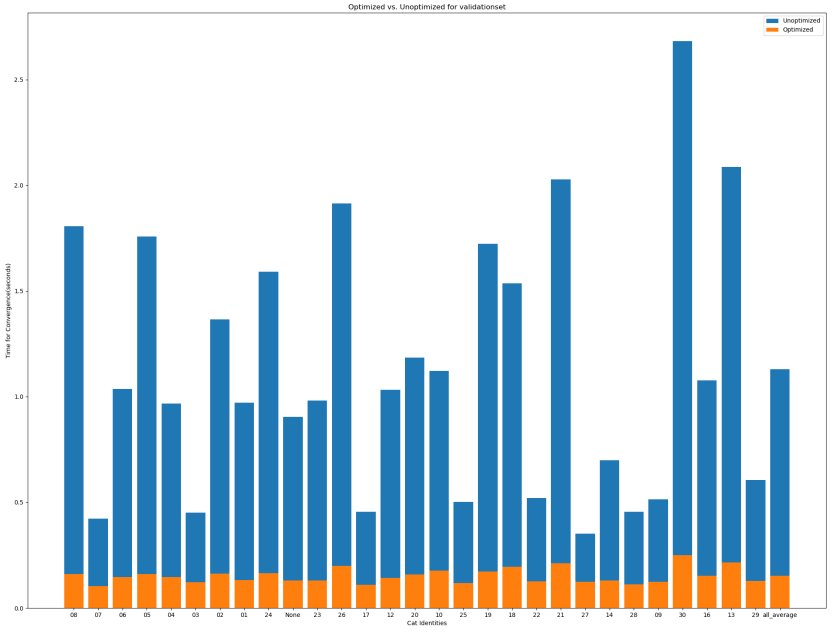


Figure 53: Graphical Speed Result for Metadata 4 Validation Set

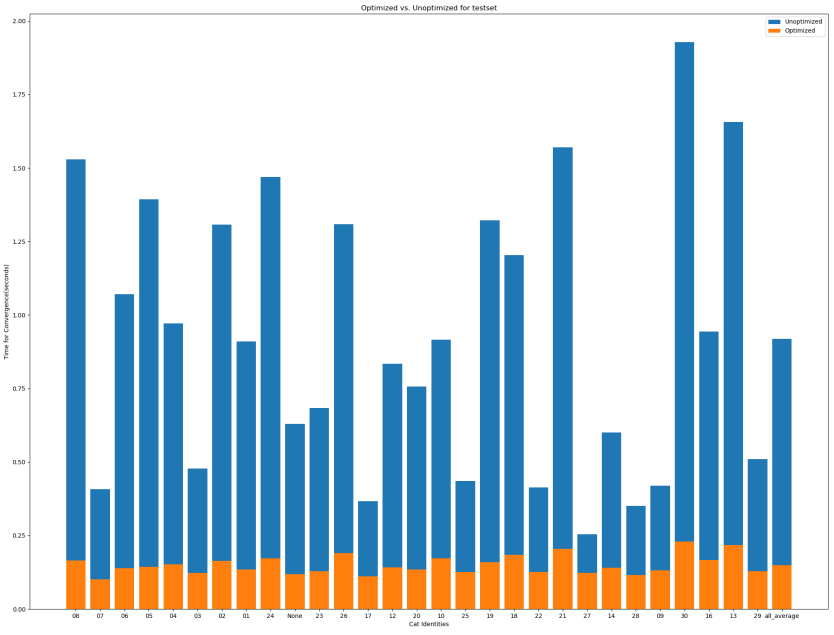


Figure 54: Graphical Speed Result for Metadata 4 Test Set

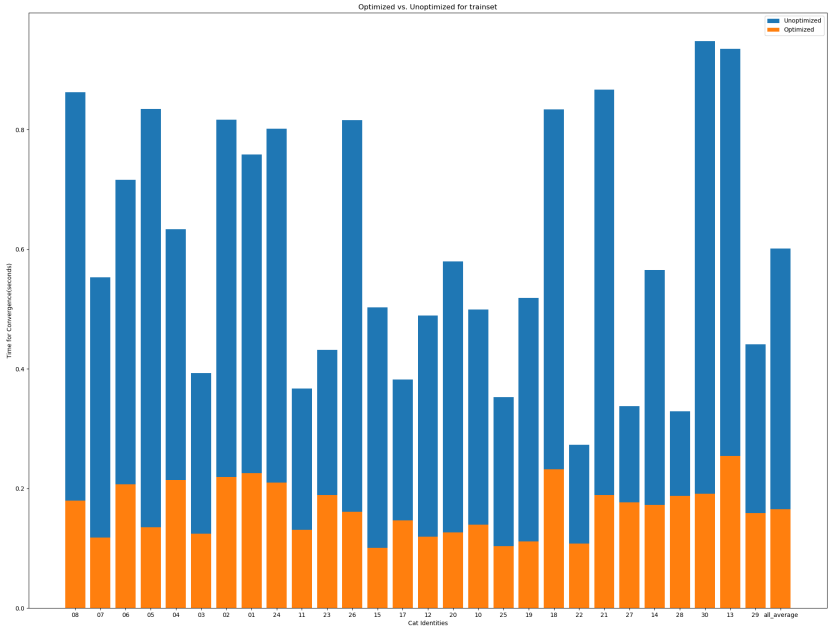


Figure 55: Graphical Speed Result for Metadata 5 Train Set

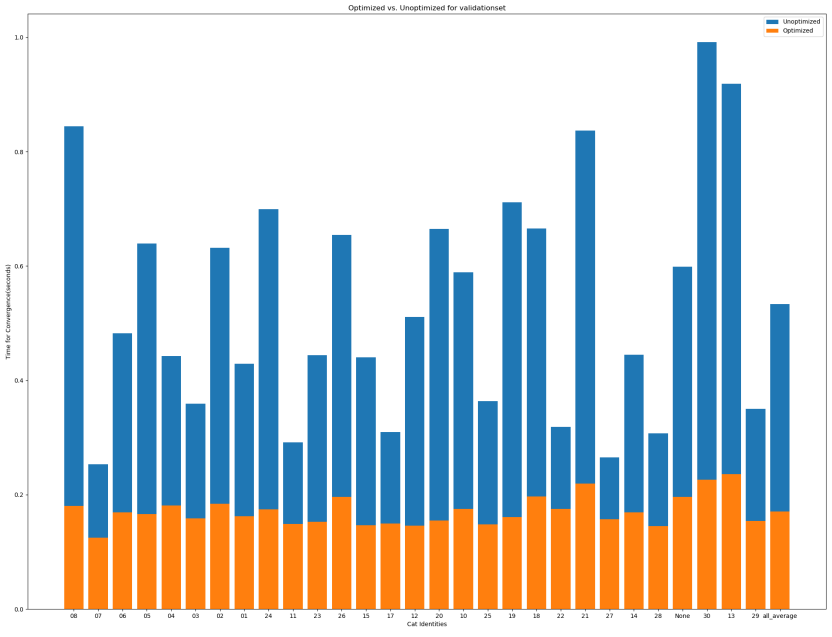


Figure 56: Graphical Speed Result for Metadata 5 Validation Set

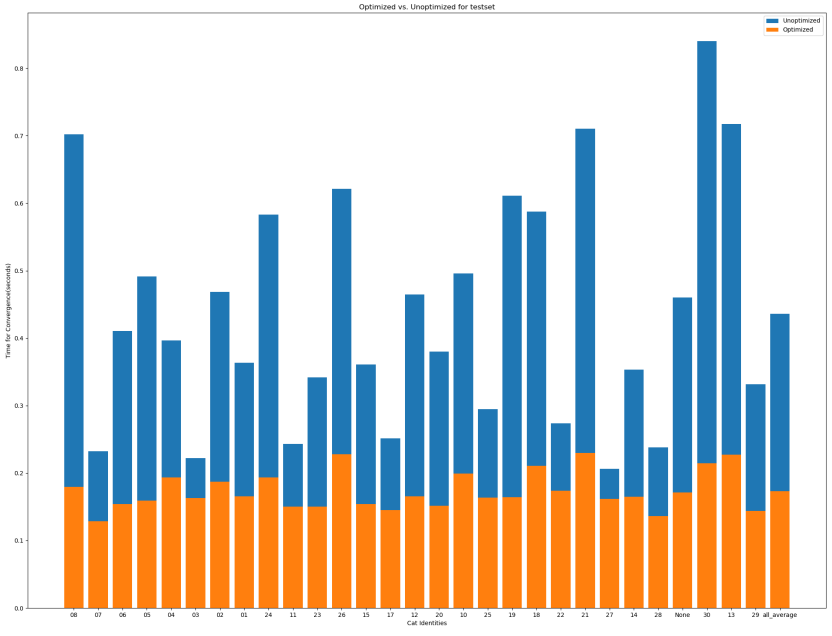


Figure 57: Graphical Speed Result for Metadata 5 Test Set

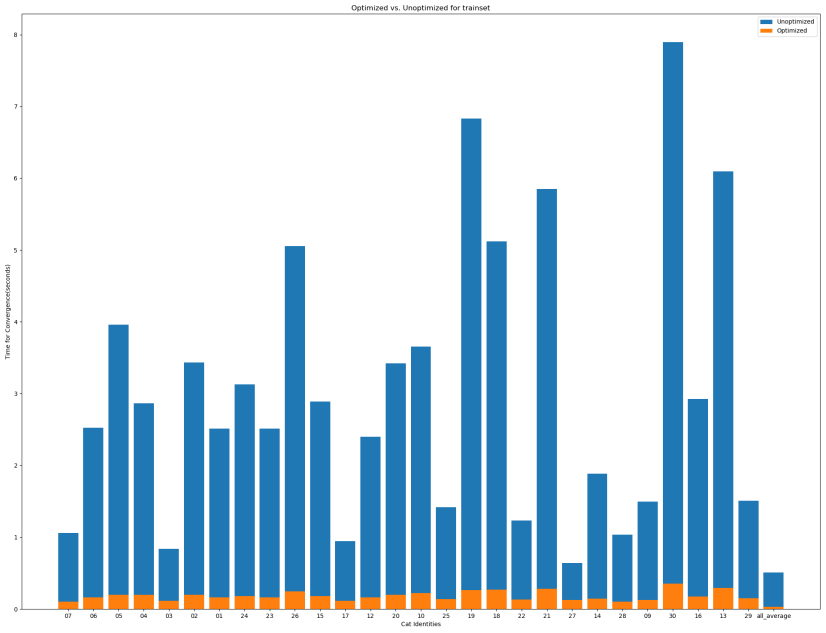


Figure 58: Graphical Speed Result for Metadata 6 Train Set

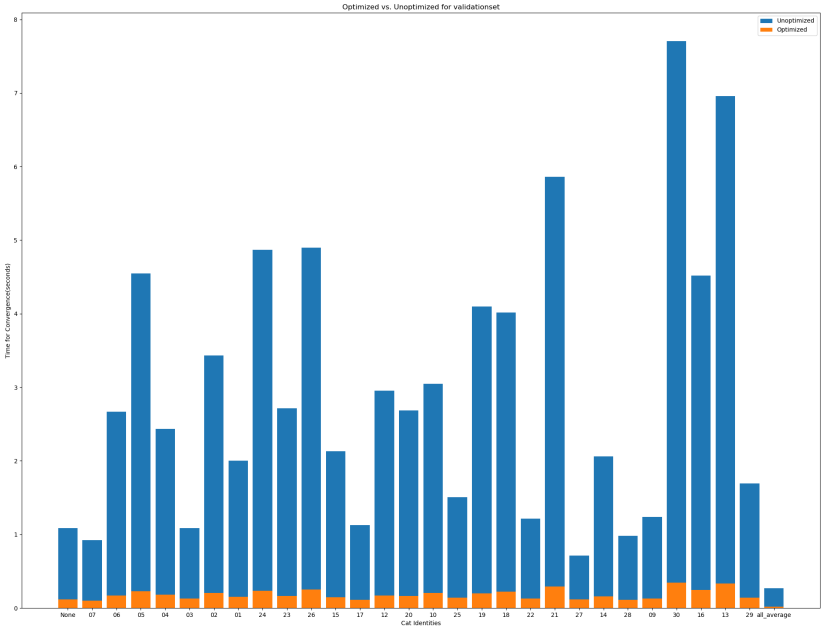


Figure 59: Graphical Speed Result for Metadata 6 Validation Set

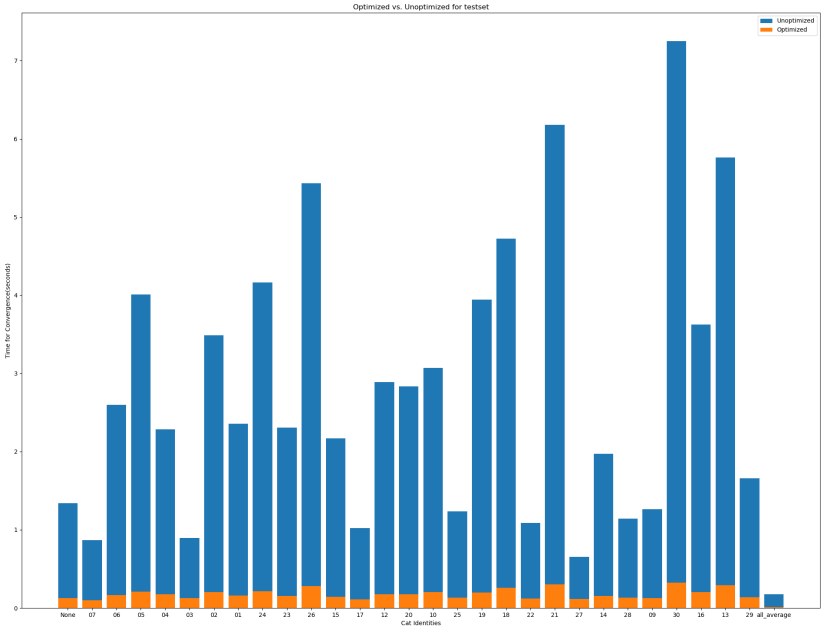


Figure 60: Graphical Speed Result for Metadata 6 Test Set

1..5 Summary

Previous sections present results in detail. To understand decision and results, a summary of them are given. Note that this part is for intuition and real results still lie in in the detailed sections. The results are for given both optimized, and unoptimized results. Test set results are given at the end as a final performance metric. Since all of the optimization is based on the validation set, their results are given in this section. Note, optimized results are shown with prefix "-O" in table 4. Time measurements are in seconds.

Data Set	Accuracy	O-Accuracy	Convergence Time	O-Convergence Time
Metadata 2	0.9854	0.9232	13.599	0.2261
Metadata 3	0.9484	0.8394	1.6283	0.1378
Metadata 4	0.8613	0.6687	1.2677	0.1519
Metadata 5	0.7503	0.4759	0.6015	0.1706
Metadata 6	0.9530	0.8527	2.6525	0.1489

Table 4: Average Results for Different Data Sets

The results are obtained using the hardware given below. Note that the tests are done in parallel and maximum number of cores exceeded which causes the solution time to take 70-80 % more. In a server processor, the results are much faster.

```
Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
```


```
Address sizes:                   43 bits physical, 48 bits virtual
CPU(s):                          8
On-line CPU(s) list:             0-7
Thread(s) per core:              2
Core(s) per socket:              4
Socket(s):                       1
NUMA node(s):                    1
Vendor ID:                       AuthenticAMD
CPU family:                      23
Model:                           24
Model name:                      AMD Ryzen 5 3500U
                                 with Radeon Vega Mobile Gfx
```

1..6 Discussion

The results show that increasing sample size increases the accuracy, but reduces speed. The balance between them are created using validation set where times are not too much, less than 2 seconds and accuracy is not less than a threshold 0.9. Below discussion digs into different data set characteristics. Note that most of the data are given under links referred in the previous parts which means only very brief results are introduced here that carry important role in real world application. Reader is encouraged to investigate full data that are mentioned for detailed understanding of the system.

Data set 5 consists of 2 training images. This case is also defined as the new cat case. When the cat is recognized at first, the system should recognize the cat later. The accuracy for this task is 0.75 which is not bad since this task is the hardest. Moreover, identification time for this task is 0.6015 seconds which is very good. Identifying a cat with single shot is defined as 2 shots because Raspberry Pi camera can take two consecutive shots. This part results are satisfactory for the defined application.

Data set 4 consists of 3 training images. This case is tried so that minimum time spent can be optimized. The accuracy for this part is 0.86 for validation set which is obviously better than data set 5. However, identification time is 1.2677 seconds which is relatively worse but still very good for this task.

Data set 3 consists of 4 training images. The results for this part is better than the former case which is expected. The average accuracy for validation set is 0.95 where average identification time is 1.6283 seconds. This is the most practical one for accuracy and identification time.

Data set 2 is for maximum accuracy and includes 11 train images per class as explained in data set section. The accuracy results are almost perfect for this data set. However, identification times are much far from real time performance. Therefore, this case is only used for benchmarking.

Finally, optimization code reduces the match time significantly. However, for this project, it is better to have less number of train cat images since number of cats are so limited in the final product. However, for larger cat sizes, such as 200-300 active cats, optimizing database plays a very important role. In that scenario, identification parameter 'new cat threshold' which is a threshold of vector distance for a new cat should be re-adjusted. Confusion matrices given for

optimized and unoptimized results suggest that most cases of the wrong identification result from identifying a known cat as a new cat, or namely 'None'. Therefore, it is left for further modifications if necessary as parameter optimization. However, as mentioned, current system is capable of handling the situation satisfactorily; therefore, no new cat parameter optimization is used anymore, but included as a comment in case of any further improvement needed. As a final word, the system best performs with Metadata 3 which is selected based on the validation results. This configuration gives test accuracy 0.9510 and test time 1.3324 seconds without any optimization. This is the most unbiased performance result of the system.

b) Deliverables

There will be six deliverables that Felerest will provide to a future team of engineers and they are listed below.

- The inner box.
- Materials and design of the outer box.
- Software codes of the classification, identification, and decision making algorithms, weight sensor, sonar sensor,camera and GPIO driver for the raspberry pi, communication protocol between raspberry and arduino, web application !!FATİH BURAYI TAMAMLASANA!!
- Dataset of the cats and dogs.
- A mobile/web application that shows food and battery level, feeding logs of the cats.
- A thorough manual for the current system setup, mobile/web application, and how to complete the designed outer box and merge with other subsystems.

c) Cost Analysis

After choosing the components used in the design, a price comparison for each component has been made to minimize the cost without performance loss. The components are shown in the Table 5.

For microcomputer unit, we bought Raspberry Pi Zero WH which is the least expensive Raspberry containing a wireless module inside it.

For camera unit, three different options are compared in terms of cost and performance, which are Raspberry Pi camera module v1, Raspberry Pi camera module v2, and a webcam. The performance of the webcam does not satisfy our image quality requirements, and the camera module v2 is expensive, Raspberry Pi camera module v1 is selected as the camera.

For motor unit, servo motors MG996R, SG90 are compared. Although the SG90 motor is cheaper than MG996R, its output torque does not satisfy the required rotation for the mechanical part. Hence, MG996 motor is selected.

For batteries, Li-Ion, Li-Po, and power bank options are compared in terms of the performance, capacity, and price, and 18650 Li-Ion batteries are selected.

For distance measurement unit, ultrasonic sonar sensor HC-SR04 and infrared sensor GP2Y0A710K0F are compared in terms of performance and price. The ultrasonic sensor is selected due to its

Item	Quantity	Price ($)	Total ($)
Raspberry Pi Zero WH	1	20	20
Raspberry Pi Camera V1	1	16	16
Raspberry Pi Camera Cable	1	3.5	3.5
Raspberry Pi Camera Case	1	4.5	4.5
16 gb MicroSD Card	1	5	5
2650 mAh Li-Ion Battery	4	5	20
1A Lithium Battery Charger	3	1	3
Micro USB Adapter	1	4	4
MicroUSB-USB Cable	1	4	4
5V Voltage Regulator	1	2	2
Ultrasonic Sonar Sensor HC-SR04	1	1	1
Weight Sensor	1	1.5	1.5
Servo Motor MG996R	1	1	5
5V Relay	4	0.8	3.2
Arduino Uno R3	1	6.5	6.5
Mechanical Components	-	-	20
Outer Box	-	-	20
3D Printed Components	-	-	13
		Total Cost	153.2

Table 5: Estimated Cost Analysis for the Project

lower price.

For cables, regulators, relays, and other components, unit prices are similar compared to different options, so the selection is made with respect to the performance.

5 Conclusion

Looking at the environment one can see the food leftovers that cause visual pollution which also lead to health and hygiene problems. Felerest are determined to solve these problems while protecting our beloved friends. To do so, Felerest engineers are determined to create novel ideas which will lead the field of automatic cat feeding systems. Our goal is to improve the life standard of the cats and humans by using the blessings of technology.

The purpose of this project is not only to feed cats, but also to come up with a new product that considers cat needs. Felerest also offer features for extensive cat feeding such that it respects the cats for the required amount of food while also determining their eating behaviours, without any human interaction. This project is useful for the owners which can not leave home due to their cats and for those cats who live in the streets or in sanctuaries who are in need of human care.

Every module's test results, planned future test ideas, the foreseen risks and disadvantages were discussed thoroughly. Furthermore, each engineer's following responsibilities and the role division was separated reasonably. This report includes flow charts that explain the algorithms implemented in the computer vision part, 3D designs of the mechanical structures, pictures of test results for each subsystem, compatibility analysis of overall system, expected power consumption, and cost analysis. This report was written in order to show a clear picture of the project for customers and members of the company. We believe that customers should always have the chance to personalize their products. This is why Felerest provides customer oriented additional features to the mobile/web applications.

With this report, Felerest team shows that it is capable of taking this project to the next level and mass produce autonomous cat feeding systems. All sub-modules are up to par and ready to be a part of the end result. This is demonstrated by test results and relevant pictures.

Having concluded the report, we are excited to announce that the project itself is also concluded. Following this report will be some informational videos, which are the latest fruits of a year long research and development phase. Considering the obstacles that we faced along the way, we are happy with achieving our goals and bringing this project to life. This thoroughly planned first ever connected, smart, cat feeder will surely be an important step for the pet society.

References

[1] Jennifer Coates. How much should i feed my cat, 2012. Last accessed 20 November 2019, `https://www.petmd.com/blogs/nutritionnuggets/cat/jcoates/2012/june/how_much_should_you_feed_your_cat-23694l`.

[2] Anastasis Kratsios. Universal approximation theorems, 2019.

[3] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018.

[4] Xiangpo Wei, Xuchu Yu, Bing Liu, and Lu Zhi. Convolutional neural networks and local binary patterns for hyperspectral image classification. *European Journal of Remote Sensing*, 52(1):448–462, 2019.