L.N. Gumilyov Eurasian National University
Faculty of Information Technology
Department of Information Systems

Cat and Dog recognition
Domestic animals

presented by Zulkhairatuly Yerlan & Mustafa Zhansaya

Project Overview

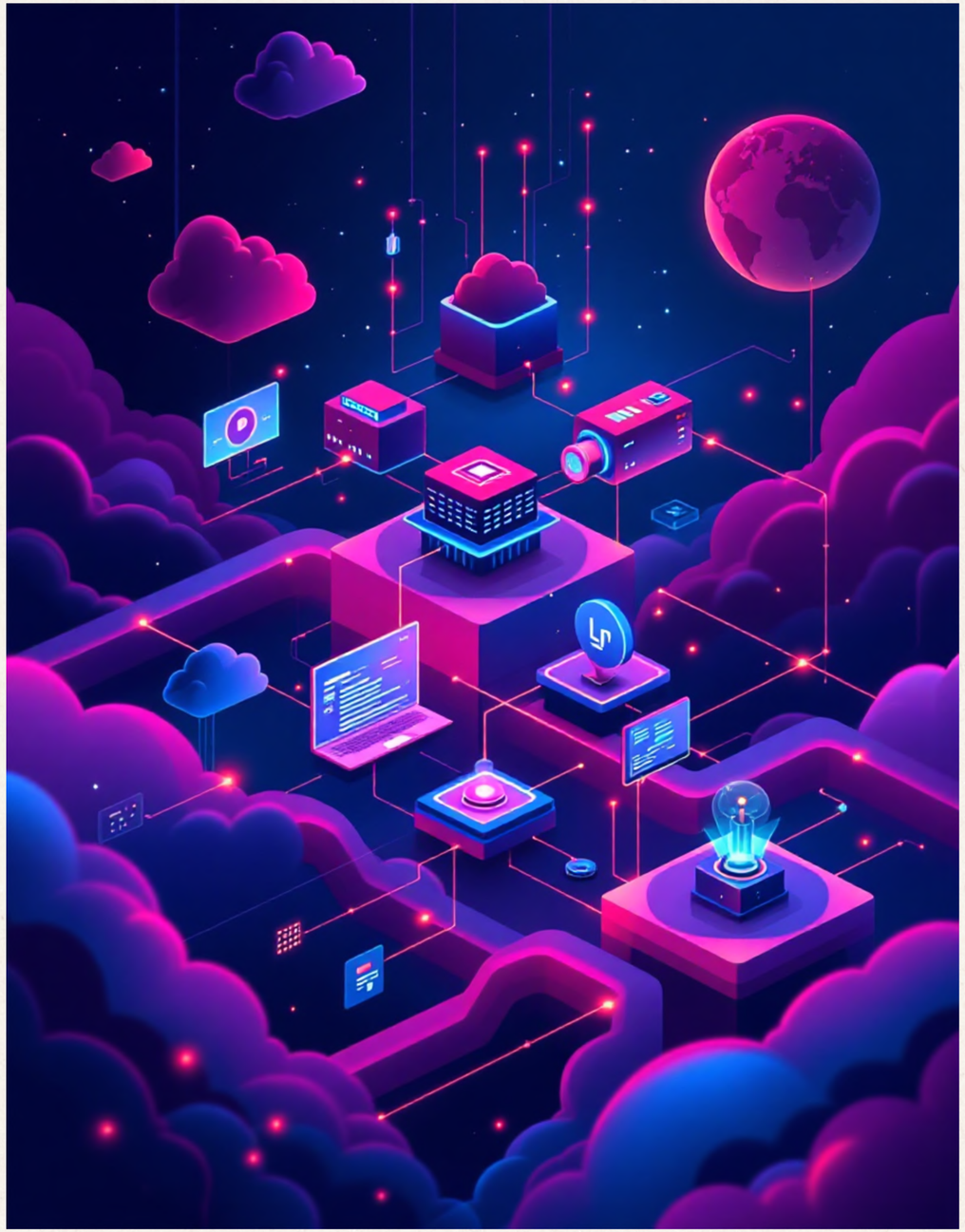
This project is a complete desktop application designed for real-time dog&cat breed recognition. It uses computer vision techniques together with a convolutional neural network to analyze an image of a dog/cat and determine its most likely breed. The system includes a user-friendly graphical interface built with PyQt5, allowing users to upload images, capture frames from a webcam, and view predictions instantly. The goal of this project is to create a simple yet powerful recognition tool that can classify up to 37 different dog/cat breeds with high accuracy.

Dataset and Training Process

The Oxford-IIIT Pet Dataset is a popular image recognition dataset developed by the University of Oxford.
It includes 37 pet breeds (12 cats and 25 dogs) and contains 7,390 images, varying in pose, facial expression, lighting, and background.
Each image has:

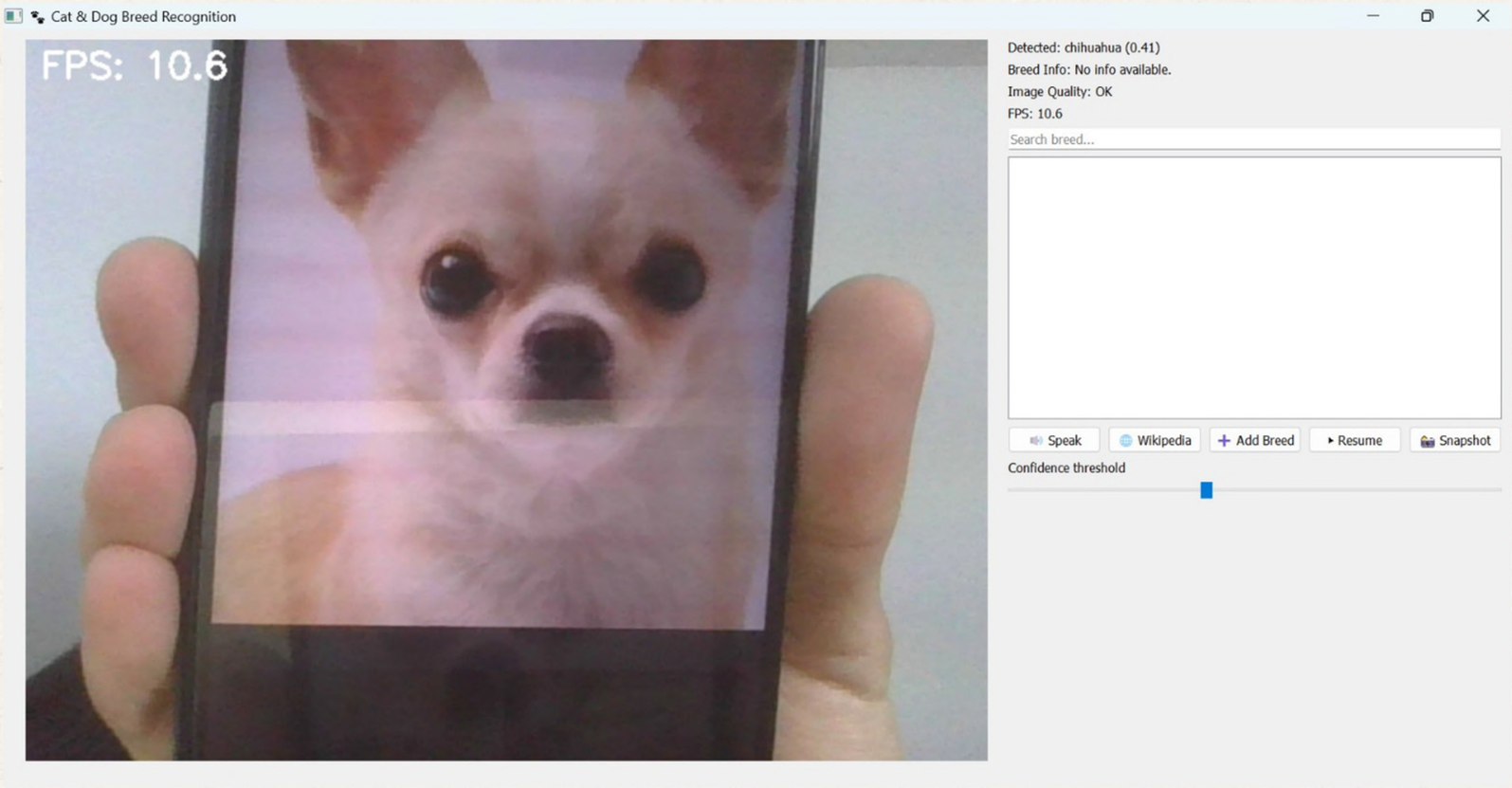
- A breed label
- A segmentation annotation (trimap)
- XML files with additional information

This dataset is widely used for training classification and segmentation models, especially in combination with neural networks such as MobileNet, ResNet, and EfficientNet.



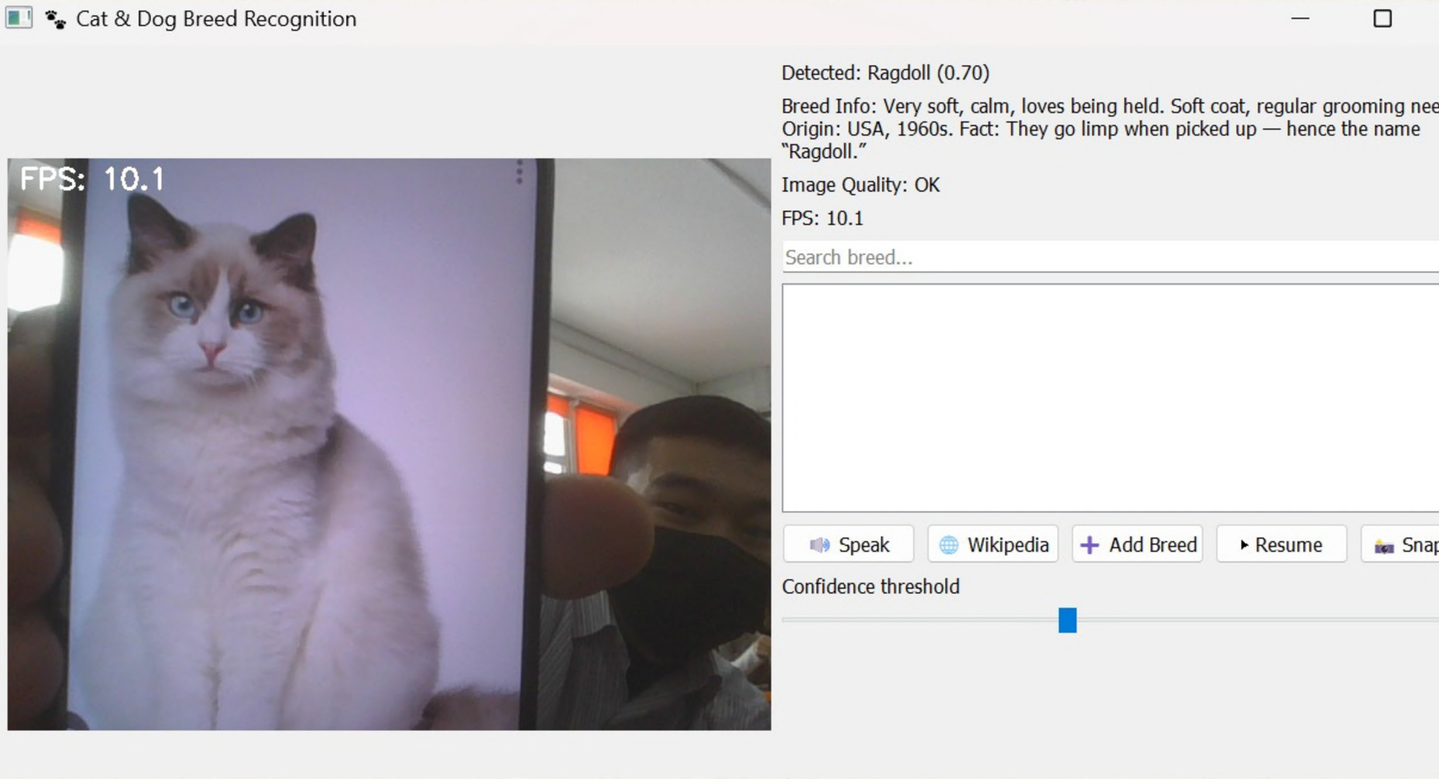
Machine Learning Model

The core of the system is a convolutional neural network implemented using TensorFlow and Keras. This neural network consists of several convolutional layers responsible for extracting features from the images, such as edges, textures, and shapes. These features are then fed into fully connected layers which analyze patterns and produce a final classification. The model is saved in H5 format and loaded during runtime, allowing fast predictions without requiring retraining. The architecture was chosen to balance accuracy and efficiency, ensuring smooth performance even on typical home computers.



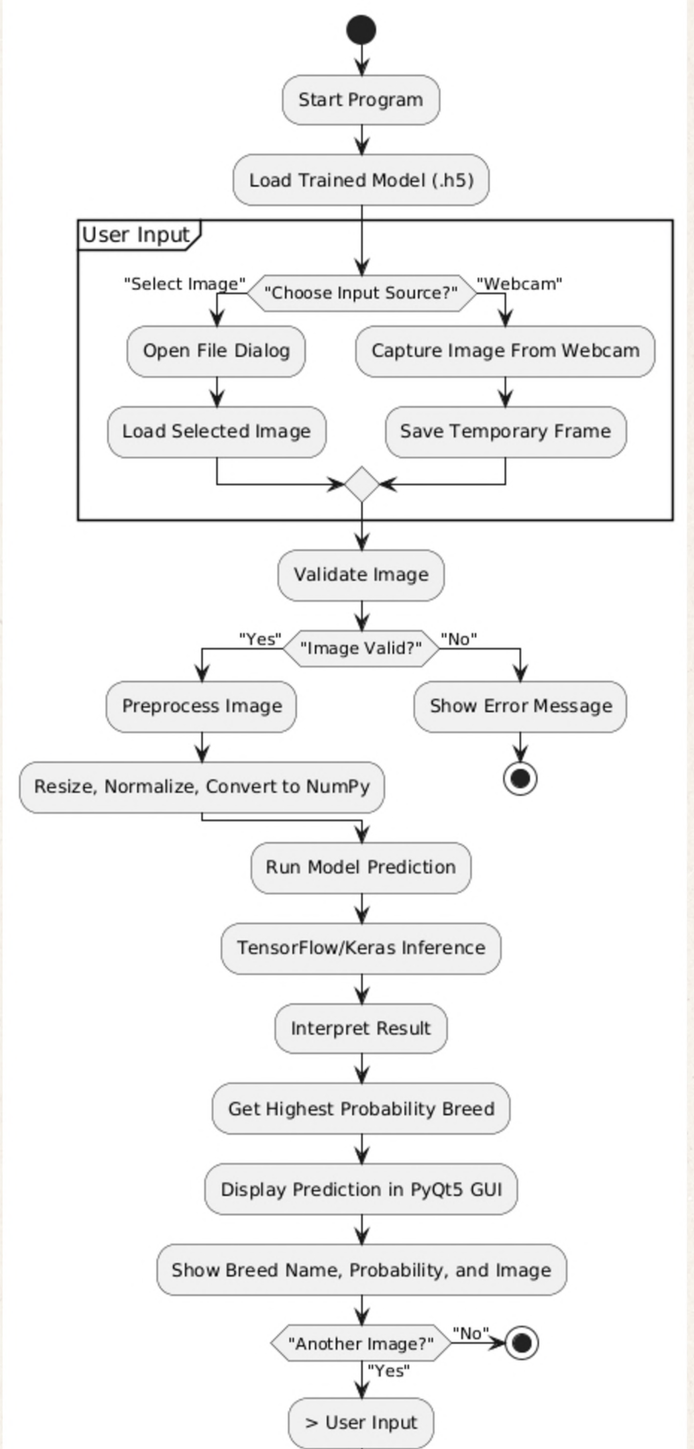
Application User Interface

The graphical interface was built using the PyQt5 framework. It includes buttons for selecting an image, displaying it inside the window, and running the recognition process. The interface also allows webcam integration so users can take a live picture and instantly see the predicted breed. The GUI displays the prediction result along with the probability value returned by the neural network. All elements of the interface were designed for clarity, minimalism, and ease of use, making the application accessible to both casual users and developers.



Recognition Workflow

When a user selects or captures an image, the application first validates the input and ensures the file can be processed. The image is then resized to the model's required dimensions and transformed into a numerical array. This array is normalized and passed into the neural network to perform the actual prediction. Once the model returns an output vector, the system identifies the breed with the highest probability and displays it on the screen. The entire process—from image selection to prediction—takes less than a second, depending on system hardware.



Results and Accuracy

After training for fifty epochs using the selected dataset, the model demonstrated high accuracy across most dog breeds. Breeds with rich and distinctive visual characteristics such as color patterns or fur shapes performed exceptionally well. Some breeds that are visually similar showed slightly lower accuracy, which is expected in smaller datasets. Overall, the model provides reliable predictions for all ten breeds and performs consistently on both training and unseen test images. Additional improvements can be made in the future by increasing dataset size and expanding the number of breed categories.

main.py | лдлд.py | 33333333333.py | flower_care.db | Erllll.py

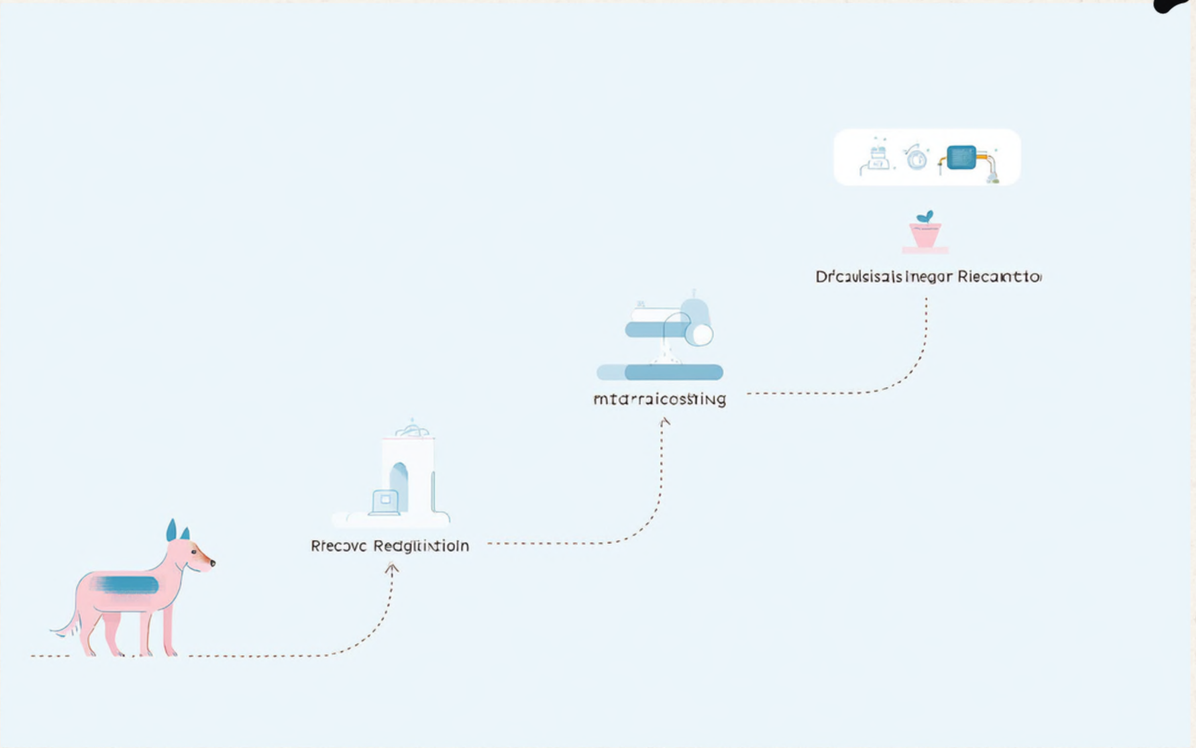
```
1  import tarfile
2
3  tar = tarfile.open( name: 'images.tar.gz',  mode: 'r:gz')
4  for member in tar.getmembers():
5      print(member.name)
6  tar.close()
7
```

лдлд

```
C:\Users\User\PycharmProjects\bill\pythonProject2\.venv\Scripts\python.
images
images/boxer_16.jpg
images/chihuahua_165.jpg
images/pug_183.jpg
images/english_setter_1.jpg
images/chihuahua_170.jpg
images/english_cocker_spaniel_17.jpg
images/samoyed_39.jpg
images/Egyptian_Mau_62.jpg
images/samoyed_36.jpg
images/german_shorthaired_3.jpg
images/Ragdoll_183.jpg
```

Application User Interface

The graphical interface was built using the PyQt5 framework. It includes buttons for selecting an image, displaying it inside the window, and running the recognition process. The interface also allows webcam integration so users can take a live picture and instantly see the predicted breed. The GUI displays the prediction result along with the probability value returned by the neural network. All elements of the interface were designed for clarity, minimalism, and ease of use, making the application accessible to both casual users and developers.



Conclusion

The dog and cat breed recognition project successfully demonstrates how modern deep learning techniques can be integrated into a practical, user-friendly desktop application. By combining a convolutional neural network with an intuitive PyQt5 interface, the system provides fast and accurate breed predictions from both uploaded images and live webcam input. The model, trained on a curated dataset of ten breeds, delivers reliable results and showcases the potential of AI-driven image classification in real-world scenarios.

Reference

1. Oxford-IIIT Pet Dataset (https://www.robots.ox.ac.uk/~vgg/data/pets/)
2. Computer Vision Using Deep Learning: Neural Network Architectures with Python and Keras — Vaibhav Verdhan (2021)
3. A Guide to Convolutional Neural Networks for Computer Vision — Salman Khan, Hossein Rahmani и др. (2018)
4. Computer Vision: Algorithms and Applications — Richard Szeliski
5. Computer Vision and Image Processing: A Practical Approach Using CVIPTools — Scott E. Umbaugh
6. Hands-On Computer Vision with TensorFlow 2 — Benjamin Planche, Eliot Andres
7. Deep Learning for Image Recognition — Rajeev Ratan
8. Deep Learning for Computer Vision with Python — Adrian Rosebrock
9. Practical Computer Vision Projects — Gary Bradski, Adrian Kaehler

GitHub

Thank you!