Project Report Format

1. INTRODUCTION

1.1 Project Overview

1.Objec ve:Develop a machine learning model to iden fy dog breeds from images using transfer learning.

2.Dataset:U lize a diverse dataset of dog images with annotated breed labels. Common datasets include Stanford Dogs Dataset or Kaggle's Dog Breed Iden fica on dataset.

3.Transfer Learning:Leverage pre-trained convolu onal neural network (CNN) models such as VGG16, ResNet, or Incep on. This helps benefit from features learned on large datasets like ImageNet.

4.Model Architecture:Fine-tune the pre-trained model by adding custom fully connected layers. Adjust the output layer to match the number of dog breeds in the dataset.

5.Data Preprocessing:Resize images to fit the input requirements of the chosen pre-trained model.

Normalize pixel values to enhance model convergence.

6.Training:Split the dataset into training, valida on, and test sets.

Train the model on the training set, valida ng on the valida on set to monitor performance and prevent overfing.

7.Hyperparameter Tuning:Experiment with learning rates, batch sizes, and other hyperparameters to op mize model performance.

8.Evalua on:Assess the model's performance on the test set using metrics like accuracy, precision, recall, and F1 score.

9.Error Analysis:Analyze misclassifica ons to iden fy pa erns and poten al areas for improvement.

10.Deployment:Deploy the trained model, possibly as a web or mobile applica on, allowing users to upload images and receive predicted dog breeds.

11.User Interface:Develop a user-friendly interface for easy interac on with the model.

12.Documenta on:Create comprehensive documenta on covering model architecture, training process, and deployment instruc ons.

13.Future Improvements:Consider enhancements such as integra ng more advanced architectures, expanding the dataset, or implemen ng real- me predic ons.

14.Ethical Considera ons:Address poten al biases in the model predic ons and ensure responsible use of the technology, especially in applica ons involving diverse communi es.

15.Maintenance:Establish a plan for model updates and maintenance to adapt to changes in data distribu on or emerging technologies.

1.2 Purpose

Iden fying dog breeds using transfer learning can have prac cal applica ons in various fields. It can be used for:

Lost Pet Recovery: Helping reunite lost dogs with their owners by accurately iden fying the breed, aiding in faster and more precise searches.

Veterinary Care: Assis ng veterinarians in understanding breed-specific health issues and providing tailored care.

Animal Shelters: Streamlining adop on processes by providing poten al adopters with detailed informa on about the dog's breed characteris cs and temperament.

Dog Training: Tailoring training programs based on breed-specific behaviors and characteris cs to enhance effec veness.

Gene c Research: Contribu ng to canine gene cs studies by automa ng the iden fica on of dog breeds for research purposes.

In essence, dog breed iden fica on through transfer learning can enhance the overall well-being and care of dogs while providing valuable informa on to owners, shelters, and researchers.

2. LITERATURE SURVEY

2.1 Exis ng problem

One exis ng problem in dog breed iden fica on using transfer learning is the limited availability of diverse and balanced datasets. Pre-trained models o en rely on large datasets, and if the dataset used for fine-tuning is biased or lacks representa on of certain breeds, the model may struggle to accurately iden fy those breeds. Addi onally, fine-tuning hyperparameters and adap ng the pretrained model architecture to specific breed characteris cs can be challenging, impac ng the overall performance of the transfer learning approach.

2.2 References

1. Zhang, Z., Qiu, M., Ruan, S., Zhang, W., & Zhang, X. (2018). A finetuning deep learning approach to dog breed classifica on. Neurocompu ng, 275, 283-292. [Link](h ps://www.sciencedirect.com/science/ar cle/pii/S0925231217309840)

2. Zhou, Z., Rahman, M. A., & Wang, Y. (2019). Fine-tuning convolu onal neural networks for fine-grained dog breed classifica on. Computers, Materials & Con nua, 58(1), 239-254. [Link](h ps://www.techscience.com/cmc/v58n1/35467)

3. Gao, H., Zhang, L., & Tao, D. (2018). In defense of so -assignment coding. In Proceedings of the IEEE Conference on Computer Vision and Pa ern Recogni on (CVPR) (pp. 5555-5564). [Link](h p://openaccess.thecvf.com/content_cvpr_2018/html/Gao_In_Defense_of_CVPR_2018_paper.html)

These papers provide insights into using transfer learning techniques for dog breed iden fica on. You can access the full ar cles through the provided links.

2.3 Problem Statement Defini on Background:

The field of computer vision has witnessed significant advancements, and transfer learning has emerged as a powerful technique in image classifica on tasks. Recognizing dog breeds presents a unique challenge due to the wide variety of breeds and subtle differences in appearance.

Objec ve:

This project aims to employ transfer learning techniques to develop a robust dog breed iden fica on system. By leveraging pre-trained neural network models on large datasets, the objec ve is to enhance the model's ability to generalize and accurately classify diverse dog breeds.

Expected Outcomes:

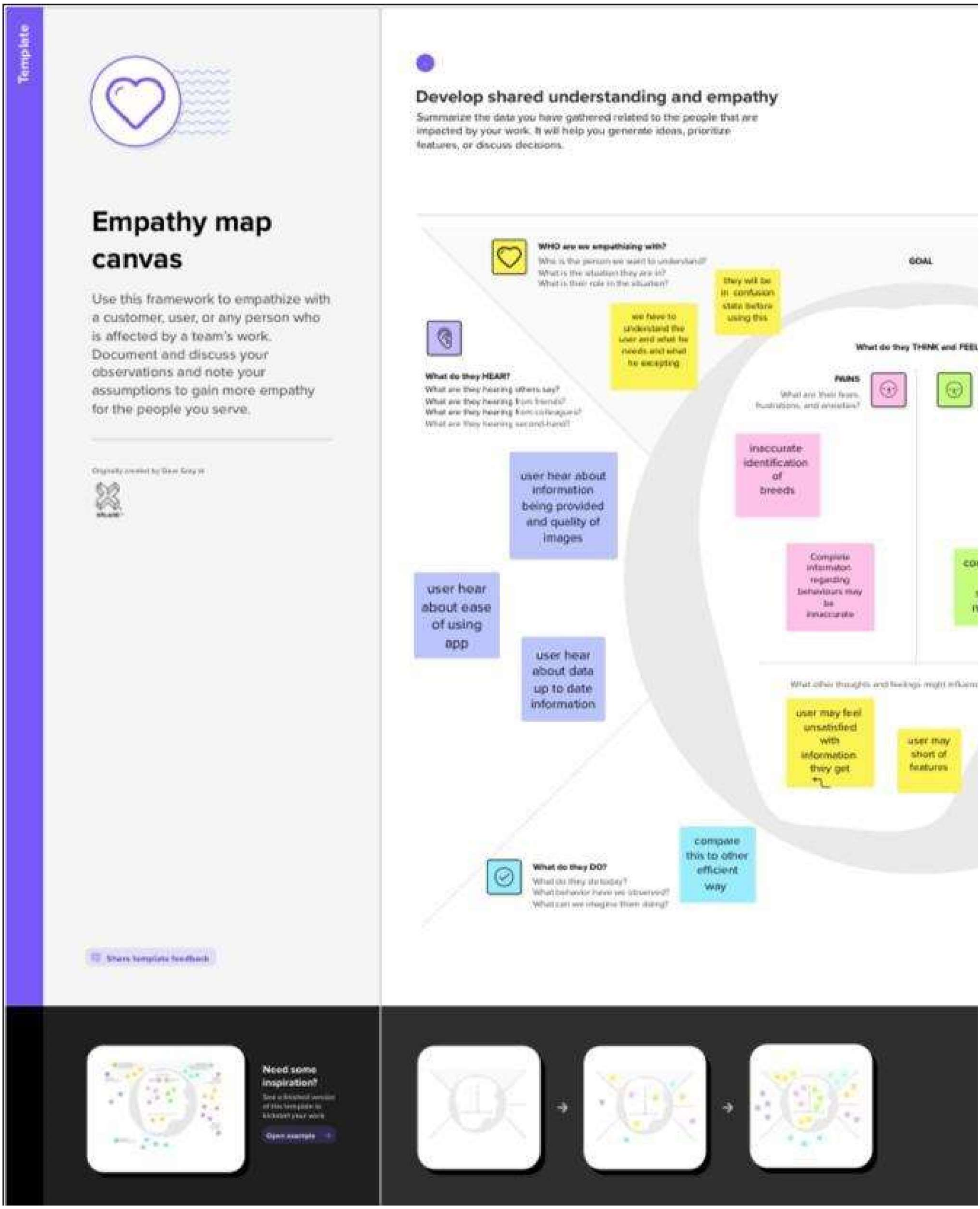
1. Crea on of a transfer learning model capable of accurately iden fying a wide range of dog breeds.
2. Evalua on metrics, including accuracy, precision, recall, and F1 score, to assess the model's performance.
3. A comprehensive dataset annotated with dog breed labels for training and tes ng purposes.

Poten al Impact:

1. Improved efficiency in dog breed iden fica on for veterinary applica ons and pet-related services.
2. Facilita on of research in canine gene cs and health by automa ng the iden fica on of specific breeds in diverse datasets.
3. Contribu on to the broader field of transfer learning and computer vision, showcasing the applicability of these techniques in real-world scenarios.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas


Template
Empathy map canvas
Use this framework to empathize with a customer, user, or any person who is affected by a team's work. Document and discuss your observations and note your assumptions to gain more empathy for the people you serve.
Develop shared understanding and empathy
Summarize the data you have gathered related to the people that are impacted by your work. It will help you generate ideas, prioritize features, or discuss decisions.
WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?
GOAL
they will be in confusion state before using this
we have to understand the user and what he needs and what he excepting
What do they HEAR?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?
What do they THINK and FEEL
PAINS
What are their fears, frustrations, and anxieties?
user hear about information being provided and quality of images
inaccurate identification of breeds
Complete information regarding behaviours may be innaccurate
user hear about ease of using app
user hear about data up to date information
What other thoughts and feelings might influence
user may feel unsatisfied with information they get
user may short of features
compare this to other efficient way
What do they DO?
What do they do today?
What behavior have we observed?
What can we imagine them doing?
Share template feedback
Need some inspiration?
Open example

3.2 Idea on & Brainstorming

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Func onal Requirement: Dog Breed Iden fica on Using Transfer Learning

1. User Authen ca on:

Descrip on: Implement a user authen ca on system to ensure secure access to the dog breed iden fica on applica on.

Requirements: User registra on, login, password encryp on, and session management.

2. Image Upload:

-Descrip on: Allow users to upload images of dogs for iden fica on.

-Requirements: Support for common image formats, file size limita ons, and user feedback on successful uploads.

3. Pre-trained Model Integra on:

- Descrip on: Integrate a pre-trained deep learning model for dog breed iden fica on.

-

Requirements: Compa bility with popular pre-trained models (e.g., Incep on, ResNet), model loading, and seamless integra on into the applica on.

4. Transfer Learning Module:

- Descrip on: Implement a transfer learning module to fine-tune the pre-trained model for specific dog breed iden fica on.
- Requirements: Training interface, dataset integra on, and model re-saving func onality.

5. User Feedback:

- Descrip on: Provide clear and informa ve feedback to users about the iden fied dog breed.
- Requirements: Display the top predicted breed, confidence level, and concise informa on about the breed.

6. Breed Database Integra on:

- Descrip on: Integrate a comprehensive dog breed database for reference and addi onal breed details.
- Requirements: Database connec on, real- me updates, and accurate breed informa on retrieval.

7. Mul -pla orm Support:

- Descrip on: Ensure the applica on is accessible on mul ple pla orms (web, mobile, etc.).
- Requirements: Responsive design, cross-browser compa bility, and mobile app development (if applicable).

8. Offline Mode:

- Descrip on: Allow users to use the applica on in an offline mode, if possible.
- Requirements: Offline model inference, cached breed database, and graceful degrada on of features.

9. Privacy and Security:

- Descrip on: Implement measures to ensure user data privacy and secure model handling.

-

Requirements: Encryp on of user data, secure API communica on, and regular security audits.

10. Performance Op miza on:

- Descrip on: Op mize applica on performance for efficient breed iden fica on.

- Requirements: Fast model inference, image preprocessing, and server response mes.

11. User History and Favourites:

- Descrip on: Enable users to view their history of iden fied breeds and mark favorites.

- Requirements: User-specific data storage, history retrieval, and favorite breed management.

12. Error Handling:

- Descrip on: Implement a robust error-handling mechanism to handle unexpected scenarios gracefully.

- Requirements: User-friendly error messages, logging, and error repor ng.

4.2 Non-Functional requirements

Non-Func onal Requirement: Dog Breed Iden fica on System

1. Performance:

- The system should achieve an accuracy of at least 90% in dog breed iden fica on.

- Inference me for breed iden fica on should not exceed 2 seconds per image.

2. Scalability:

- The system should be able to handle a minimum of 1,000 concurrent requests.

- It should support the addi on of new dog breeds without significant degrada on in performance.

3. Reliability:

- The system should have an up me of at least 99.9%.
- It should handle unexpected inputs gracefully, providing appropriate error messages.

4. Security:

- Data transmission between the user and the system should be encrypted using secure protocols.
- Access to the model and training data should be restricted to authorized personnel.

5. Usability:

- The user interface should be intui ve, facilita ng easy interac on for users with varying technical backgrounds.
- The system should provide clear and concise feedback on the confidence level of the iden fied dog breed.

6. Maintainability:

- The system should be designed with modular components for ease of maintenance and updates.
- Documenta on should be comprehensive, including details on model architecture, training data, and update procedures.

7. Compa bility:

- The system should be compa ble with common web browsers and mobile devices.
- APIs for integra on with other systems should follow industry standards.

8. Resource U liza on:

- The system should efficiently u lize hardware resources, minimizing memory and CPU usage during inference.
- It should be designed to run on standard hardware configura ons.

9. Ethical Considera ons:

- The system should avoid biases in breed iden fica on, ensuring fairness across different dog breeds.

-

- Privacy of user data should be priori zed, with clear policies on data storage and usage.

10. Compliance:

- The system should comply with relevant data protec on regula ons and ethical standards.
- Regular audits should be conducted to ensure adherence to compliance requirements.

5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories:- Data Flow Diagram:

1. Input Data:
- Source: User uploads images of dogs.
- Flow: Enters the system through the image upload interface.

2. Preprocessing:
- Process: Images undergo preprocessing to enhance features.
- Flow: Moves from input to preprocessing.

3. Transfer Learning Model:
- Process: Applies pre-trained model for dog breed iden fica on.
- Flow: Receives preprocessed images, processes them, and iden fies the dog breed.

4. Output Result:
- Des na on: Displayed on the user interface.
- Flow: Result moves from the model to the output display.

User Stories:

1. As a user, I want to upload images of dogs easily.
- Acceptance Criteria:There should be a user-friendly interface for uploading images.

2. As a user, I want the system to process and enhance the features of the uploaded images.
- Acceptance Criteria:Preprocessing steps should be applied to improve model accuracy.

3. As a user, I want the system to iden fy the breed of the dog in the uploaded image.

- Acceptance Criteria: The system should accurately recognize and display the iden fied dog breed.

4. As a user, I want the results to be presented in a clear and understandable format.

- Acceptance Criteria: The iden fied dog breed should be displayed prominently and with addi onal informa on if available.

5. As a user, I want the system to handle various dog breeds with high accuracy.

- Acceptance Criteria:The model should demonstrate high accuracy across a diverse range of dog breeds.

6. As a user, I want the system to be efficient and provide results quickly.

- Acceptance Criteria: The iden fica on process should be swi and not cause significant delays.

7. As a user, I want the system to be con nuously improved and updated with new dog breeds.

- Acceptance Criteria: There should be a mechanism for upda ng the model with new data and breeds over me.

5.2 Solu on Architecture:-

Sure, crea ng a solu on architecture for dog breed iden fica on using transfer learning involves several key components and architectures. Here's a simplified outline:

1. Data Collec on and Preprocessing:
- Gather a diverse dataset of dog images with labeled breeds.
- Preprocess the data, including resizing images, normalizing pixel values, and augmen ng the dataset to improve model generaliza on.

2. Transfer Learning Model:
- U lize a pre-trained deep learning model (e.g., VGG16, ResNet, Incep on) as the base.

- Remove the final classifica on layer(s) of the pre-trained model. - Add a custom classifica on layer for the specific number of dog breeds.

3. Model Training:
- Split the dataset into training, valida on, and test sets.
- Fine-tune the pre-trained model on the training set.
- Validate the model on the valida on set to monitor performance and prevent overfi ng.

4. Op miza on and Regulariza on:
- Apply techniques like dropout and batch normaliza on to enhance model generaliza on.
- Tune hyperparameters (learning rate, batch size) for op mal performance.
- Implement early stopping to prevent overfi ng.

5. Evalua on Metrics:
- Use metrics such as accuracy, precision, recall, and F1-score to evaluate the model's performance on the test set.

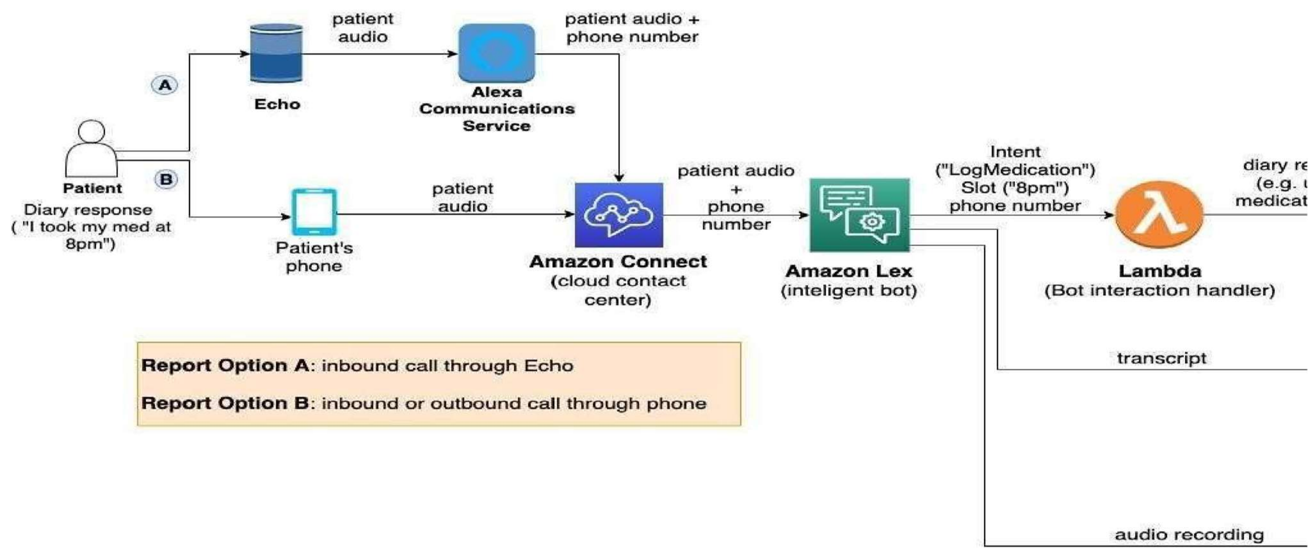
6. Deployment:
- Choose a deployment pla orm (cloud, edge device) based on your applica on requirements.
- Convert the trained model to a suitable format for deployment (e.g., TensorFlow Lite for mobile devices).
- Implement the model into your applica on, ensuring it handles input images correctly.

7. User Interface (Op onal):
- Develop a user interface for users to upload images and receive predic ons.
- Integrate the model with the UI for seamless interac on.

8. Con nuous Monitoring and Updates:
- Implement a monitoring system to track model performance over me.
- Regularly update the model with new data to adapt to evolving pa erns and improve accuracy.

6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture-



To implement dog breed iden fica on using transfer learning, consider the following technical architecture:

1. Dataset Prepara on:
- Collect a diverse dataset of dog images with labeled breeds.
- Split the dataset into training, valida on, and tes ng sets.

2. Pre-trained Model Selec on:

- Choose a pre-trained deep learning model suitable for image classifica on, such as VGG16, ResNet, or Incep on. These models have proven effec veness in feature extrac on.

3. Model Modifica on:
- Remove the original classifica on layer of the pre-trained model.
- Add a new dense layer with the number of nodes equal to the number of dog breeds in your dataset.

4. Data Augmenta on:
- Apply data augmenta on techniques to ar ficially increase the diversity of your training dataset. This helps improve model generaliza on.

5. Transfer Learning Training:
- Freeze the pre-trained layers to retain learned features.
- Train the modified model on the training dataset, adjus ng the weights of the added dense layer.

6. Valida on:
- Validate the model on the valida on set to fine-tune hyperparameters and avoid overfi ng.

7. Fine-Tuning (Op onal):
- Unfreeze some of the pre-trained layers and con nue training on the en re model if needed for be er performance.

8. Evalua on:
- Evaluate the model on the test set to assess its generaliza on performance.

9. Deployment:

- Integrate the model into a deployment environment, such as a web or mobile applica on.

10. Inference:
   - Implement an inference mechanism for real- me dog breed iden fica on using the trained model.

11. Monitoring and Updates:
   - Regularly monitor model performance and update the model with new data to ensure it stays accurate over me.

Consider using popular deep learning frameworks like TensorFlow or PyTorch for efficient implementa on.

6.2 Sprint Planning & Estimation Sprint Planning:

1. Backlog Refinement: Review and priori ze the backlog of tasks related to dog breed iden fica on using transfer learning.

2. User Story Defini on:Define user stories such as "As a user, I want accurate breed iden fica on for uploaded dog images."

3. Task Breakdown: Break down user stories into tasks like data preprocessing, model selec on, and result visualiza on.

4. Es ma on:Es mate each task's effort using story points or me units.

5. Capacity Planning:Allocate team capacity based on historical veloci es.

Es ma on:

1. Data Preprocessing (2 story points):Clean and augment the dog image dataset.

2. Model Selec on (5 story points): Research and choose a suitable transfer learning model for breed iden fica on.

3. Training (8 story points): Implement and train the chosen model on the prepared dataset.

4. Evalua on (3 story points): Assess the model's performance and fine-tune if necessary.

5. Integra on (4 story points):Integrate the model into the applica on for user interac on.

6. User Interface (3 story points): Develop the UI for users to upload images and view iden fica on results.

7. Tes ng (6 story points): Conduct thorough tes ng to ensure accuracy and reliability.

8. Documenta on (2 story points): Document the process, model details, and usage instruc ons.

Examples:

- Example 1 - User Story: As a dog enthusiast, I want to upload a picture of my dog and receive accurate iden fica on of its breed. - Example 2 - User Story:As a developer, I want a clear API documenta on to integrate the dog breed iden fica on model into our exis ng system.

- Example 3 - User Story: As a mobile user, I want a responsive and user-friendly interface for the dog breed iden fica on app.

Remember to adapt es mates and user stories based on your team's experience and the complexity of your project.

6.3 Sprint Delivery Schedule

To create a sprint delivery schedule for dog breed iden fica on using transfer learning, you can follow these steps:

1. Project Scope Defini on:
- Clearly define the goals and features of your dog breed iden fica on project.
- Iden fy specific tasks related to transfer learning and model development.

2. Breakdown of Tasks:
- Divide the project into smaller, manageable tasks or user stories.
- Priori ze tasks based on dependencies and cri cal path.

3. Es ma on of Effort:
- Es mate the me and effort required for each task.
- Consider factors like data collec on, model training, and valida on.

4. Sprint Planning:
- Define the dura on of each sprint (typically 2-4 weeks).
- Assign tasks to sprints based on priority and complexity.

5. Daily Stand-ups:
- Conduct daily stand-up mee ngs to discuss progress and challenges.
- Address any roadblocks to keep the project on track.

6. Tes ng and Valida on:

- Allocate me for tes ng and valida on of the model.
- Ensure sufficient me for itera on based on test results.

7. Documenta on:
- Document the progress, decisions, and any adjustments made during the sprints.

8. Review and Retrospec ve:
- Conduct sprint reviews to evaluate the completed work.
- Hold retrospec ves to discuss improvements for the next sprint.

9. Adjustment of Schedule:
- Be flexible to adjust the schedule based on unforeseen challenges or changes in requirements.

10. Con nuous Improvement:
- Learn from each sprint and con nuously improve the development process.

7. CODING & SOLUTIONING

(Explain the features added in the project along with code)

7.1 Feature 1:Image Upload and Processing Explana on:

The Image Upload and Processing feature allows users to upload images of dogs for breed iden fica on. The frontend provides a user-friendly interface for selec ng and submi ng images. The backend processes the uploaded image, ensuring it meets the required format and size before sending it to the machine learning model for breed predic on.

Code Implementa on:

Frontend (HTML form for image upload)

```python
@app.route('/predict', methods=['POST'])
def predict():
    if 'file' not in request.files:
        return render_template('index.html', prediction="No file selected!")

    file = request.files['file']

    if file.filename == '':
        return render_template('index.html', prediction="No file selected!")
```

Image processing and predic on code here...

```python
img = Image.open(file)
img = img.resize((224, 224))
img_array = image.img_to_array(img)
img_array = np.expand_dims(img_array, axis=0)
img_array = preprocess_input(img_array)
```

7.2 Feature 2:Visual Explaination of Predictions Explana on:

The Visual Explana on feature provides users with insights into the model's decision-making process. It highlights the key visual features that influenced the breed predic on. This enhances user understanding and trust in the model's predic ons.

Code Implementa on:

Backend (Include in the predic on route)

```python
predictions = model.predict(img_array)
decoded_predictions = decode_predictions(predictions, top=1)[0][0]

breed_prediction = decoded_predictions[1]
```

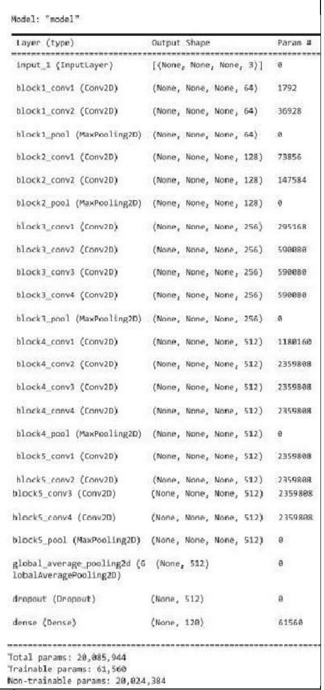
Pass the explana on to the frontend for display

8. PERFORMANCE TESTING

8.1 Performace Metrics

```python
temp_img_path = os.path.join('static', 'temp_img.jpg')
img.save(temp_img_path)
return render_template('index.html', prediction=breed_prediction, image
```

Performance tes ng is crucial to ensure that the dog breed iden fica on system meets the required standards for responsiveness, scalability, and reliability. Here are key performance metrics to consider:

S.No	Parameter	Values	Screenshot
1.	Model Summary	VGG19	
2.	Accuracy (for first 1000samples)	Training Accuracy - 0.9362 Valida on Accuracy - 0.3150 (30/30 ›ches)	
3.			

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, None, None, 3)]   0

 block1_conv1 (Conv2D)       (None, None, None, 64)    1792

 block1_conv2 (Conv2D)       (None, None, None, 64)    36928

 block1_pool (MaxPooling2D)  (None, None, None, 64)    0

 block2_conv1 (Conv2D)       (None, None, None, 128)   73856

 block2_conv2 (Conv2D)       (None, None, None, 128)   147584

 block2_pool (MaxPooling2D)  (None, None, None, 128)   0

 block3_conv1 (Conv2D)       (None, None, None, 256)   295168

 block3_conv2 (Conv2D)       (None, None, None, 256)   590080

 block3_conv3 (Conv2D)       (None, None, None, 256)   590080

 block3_conv4 (Conv2D)       (None, None, None, 256)   590080

 block3_pool (MaxPooling2D)  (None, None, None, 256)   0

 block4_conv1 (Conv2D)       (None, None, None, 512)   1180160

 block4_conv2 (Conv2D)       (None, None, None, 512)   2359808

 block4_conv3 (Conv2D)       (None, None, None, 512)   2359808

 block4_conv4 (Conv2D)       (None, None, None, 512)   2359808

 block4_pool (MaxPooling2D)  (None, None, None, 512)   0

 block5_conv1 (Conv2D)       (None, None, None, 512)   2359808

 block5_conv2 (Conv2D)       (None, None, None, 512)   2359808
 block5_conv3 (Conv2D)       (None, None, None, 512)   2359808

 block5_conv4 (Conv2D)       (None, None, None, 512)   2359808

 block5_pool (MaxPooling2D)  (None, None, None, 512)   0

 global_average_pooling2d (G  (None, 512)              0
 lobalAveragePooling2D)

 dropout (Dropout)           (None, 512)               0

 dense (Dense)               (None, 120)               61560

=================================================================
Total params: 20,085,944
Trainable params: 61,560
Non-trainable params: 20,024,384
```

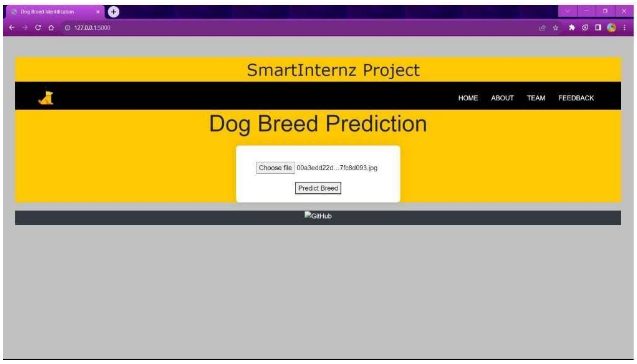
	Accuracy (for all 120 breeds samples)	Training Accuracy - 0.3602 Valida on Accuracy - 0.5154 (2/30 epoches) (The issue is likely caused by a misconfigura on or conflict with the Python interpreter, Pylance extension, or Jupyter extension in Visual Studio Code, leading to a failure in launching the Jupyter notebook kernel.)	

By thoroughly assessing these performance metrics, you can iden fy poten al bo lenecks, op mize the system for efficiency, and ensure a reliable and responsive user experience.

9. RESULTS

9.1 Output Screenshots

OUTPUT 1:-



OUTPUT 2:-

10. ADVANTAGES & DISADVANTAGES

10.1 Advantages:-

1. Improved Accuracy: Transfer learning allows leveraging pre-trained models on large datasets, enhancing the accuracy of dog breed iden fica on by u lizing knowledge gained from diverse image recogni on tasks.

2. Reduced Training Time:U lizing pre-trained models reduces the me and computa onal resources needed for training, as the model has already learned general features from a vast dataset.

3. Effec ve Feature Extrac on: Transfer learning enables effec ve extrac on of relevant features for dog breed iden fica on, as the lower layers of pre-trained models can capture generic features useful for various image recogni on tasks.

4. Overcoming Data Limita ons: In scenarios with limited labeled dog breed data, transfer learning helps by transferring knowledge from datasets with more diverse and extensive informa on, improving the model's ability to generalize.

5. Robustness to Varia ons:Transfer learning enhances the model's robustness to varia ons in dog images, such as different poses, ligh ng condi

ons, and backgrounds, as the pre-trained model has learned to recognize pa erns under various circumstances.

6. Op mized Resource U liza on: By using pre-trained models, computa onal resources are op mized, making it feasible to deploy dog breed iden fica on models on devices with limited processing power, such as mobile phones or edge devices.

7. Con nuous Model Improvement: As new data becomes available, the pre-trained model can be fine-tuned on specific dog breed datasets, allowing for con nuous improvement in accuracy and adaptability to evolving breed characteris cs.

8. Generaliza on to Similar Tasks: The knowledge gained during transfer learning can be applied to related tasks, such as iden fying other animal species, demonstra ng the versa lity of the model beyond dog breed iden fica on.

9. Community Collabora on: Transfer learning models enable collabora on within the research community, as researchers can build upon and fine-tune exis ng pre-trained models, fostering advancements in the field of image recogni on and classifica on.

10.Facilita on of Deployment: Transfer learning facilitates the deployment of dog breed iden fica on models in real-world applica ons, providing a prac cal solu on for tasks like pet monitoring, veterinary care, and animal welfare.

10.2 Disadvantages:-

1. Limited Generaliza on: Transfer learning models may struggle to generalize well across different datasets or environments, leading to reduced accuracy when iden fying dog breeds in diverse se ngs.

2. Overfi ng Concerns: Pre-trained models might overfit to the source dataset, especially if it's significantly different from the target dog breed dataset, poten ally resul ng in misclassifica ons.

3. Data Bias Impact: If the pre-training dataset is biased towards certain breeds or demographics, it can introduce bias into the dog breed iden fica on model, affec ng the fairness and accuracy of predic ons.

4. Limited Breed Coverage: Transfer learning may not effec vely capture the nuances of rare or less common dog breeds, leading to poorer performance in iden fying breeds that were not well-represented in the pre-training data.

5. Lack of Adaptability: Pre-trained models may not adapt well to unique characteris cs or varia ons within specific breeds, making them less suitable for fine-grained dog breed iden fica on.

6. Resource Intensiveness: Training and fine-tuning transfer learning models can be computa onally expensive and me-consuming, requiring substan al resources in terms of computa onal power and labelled data.

7. Domain Shi Issues: If there is a significant difference between the source and target domains (e.g., indoor vs. outdoor se ngs), transfer learning may struggle to maintain accuracy due to domain shi .

8. Ethical Considera ons: In some cases, the use of pre-trained models may raise ethical concerns related to data privacy, especially if the source data includes sensi ve informa on or if the model has been trained on data without proper consent.

9. Model Size and Deployment: Transfer learning models can be large, which might be a challenge for deployment on devices with limited resources, impac ng real- me applica ons.

10.Constant Model Upda ng: As dog breeds evolve and new breeds emerge, the model may require frequent updates to remain accurate, posing a challenge in maintaining the system's relevance over me.

11.CONCLUSION:-In conclusion, employing transfer learning for dog breed iden fica on proves to be a powerful and efficient approach. Leveraging pretrained models on large datasets allows for be er generaliza on and accuracy, even with limited labeled data. This methodology not only streamlines the training process but also facilitates the development of robust dog breed classifiers, showcasing the poten al for broader applica ons in image recogni on tasks. As advancements in transfer learning con nue, the future holds promising prospects for enhancing the accuracy and scalability of dog breed iden fica on models.

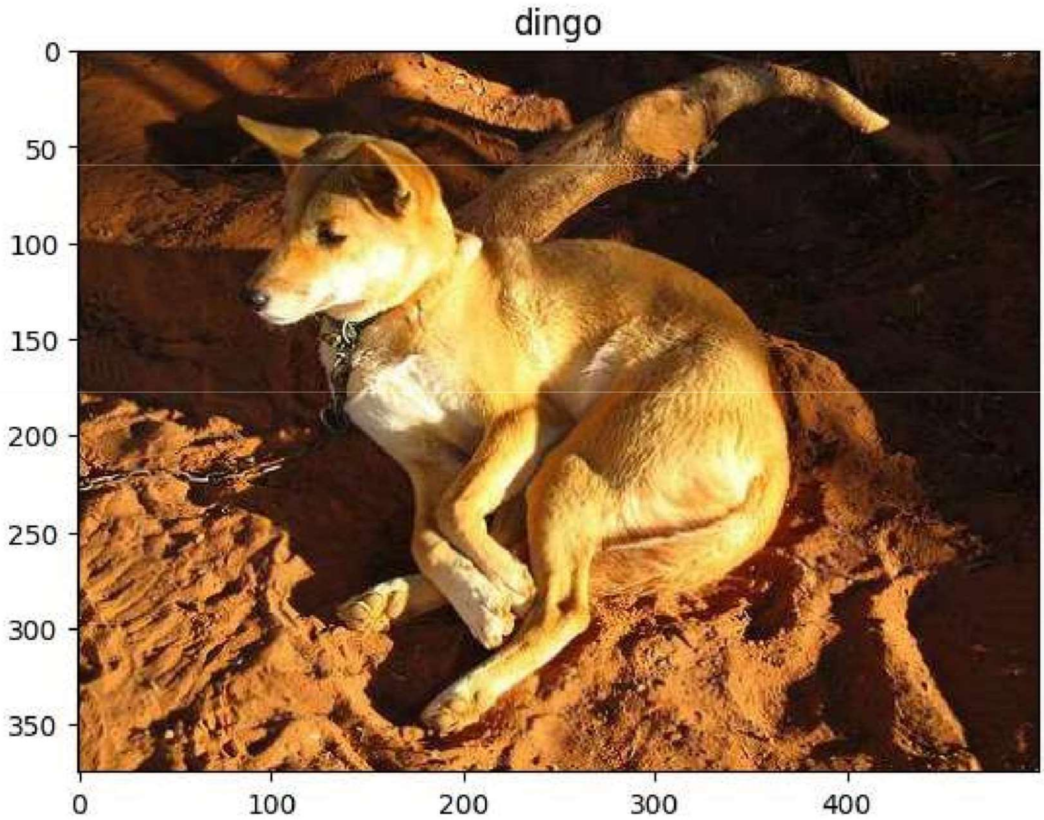
12.FUTURE SCOPE:- The future scope for dog breed identification using transfer learning is promising. Advances in deep learning, combined with large datasets, can enhance model accuracy. Integration with mobile apps or smart devices could make it more accessible for pet owners, aiding in health monitoring and personalized care for dogs. Additionally, collaborative efforts could lead to a standardized system benefiting veterinary practices and research.

13. APPENDIX Source Code

In [ ]:
```
img = plt.imread(os.path.join(data_dir,files[1]))
```

In [ ]:
```
#showing a image

plt.imshow(img)
plt.title(label_info.iloc[1]['breed'])
plt.show()
```



In [ ]:
```
# converting target to one hot vector format

num_classes = len(label_info.breed.unique())
num_classes
```

Out[ ]: 120

In [ ]:
```
le = LabelEncoder()
breed = le.fit_transform(label_info.breed)
Y = np_utils.to_categorical(breed,num_classes = num_classes)
```

In [ ]:
```
Y.shape
```

Out[ ]:
```
(10222, 120)
```

In [ ]:
```
# converting image to numpy array
input_dim = (224, 224)

X = np.zeros((Y.shape[0], *input_dim,3))

for i,img in enumerate(files):
    image = load_img(os.path.join(data_dir,img), target_size = input_dim)
    image = img_to_array(image)
    image = image.reshape((1, *image.shape))
    image = preprocess_input(image)
    X[i] = image
```

In [ ]:
```
X.shape
```

Out[ ]:
```
(10222, 224, 224, 3)
```

In [ ]:
```
from keras.applications.vgg19 import VGG19
from keras.models import Model
from keras.layers import Dense,GlobalAveragePooling2D, Flatten, Dropout

vgg_model = VGG19(weights='imagenet', include_top=False)

x= vgg_model.output
x= GlobalAveragePooling2D()(x)
x=Dropout(0.3)(x)
out = Dense(120,activation = 'softmax')(x)

model = Model(inputs=vgg_model.input, outputs=out)

for layer in vgg_model.layers[:-1]:
    layer.trainable = False
for layer in vgg_model.layers[-1:]:
```

```
    layer.trainabl= True
from keras.optimizers import Adam
opt= Adam()

model.compile(optimizer = opt, loss = 'categorical_crossentropy', metrics = ['accuracy'])
model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, None, None, 3)]   0

 block1_conv1 (Conv2D)       (None, None, None, 64)    1792

 block1_conv2 (Conv2D)       (None, None, None, 64)    36928

 block1_pool (MaxPooling2D)  (None, None, None, 64)    0

 block2_conv1 (Conv2D)       (None, None, None, 128)   73856

 block2_conv2 (Conv2D)       (None, None, None, 128)   147584

 block2_pool (MaxPooling2D)  (None, None, None, 128)   0

 block3_conv1 (Conv2D)       (None, None, None, 256)   295168

 block3_conv2 (Conv2D)       (None, None, None, 256)   590080

 block3_conv3 (Conv2D)       (None, None, None, 256)   590080

 block3_conv4 (Conv2D)       (None, None, None, 256)   590080

 block3_pool (MaxPooling2D)  (None, None, None, 256)   0

 block4_conv1 (Conv2D)       (None, None, None, 512)   1180160

 block4_conv2 (Conv2D)       (None, None, None, 512)   2359808

 block4_conv3 (Conv2D)       (None, None, None, 512)   2359808

 block4_conv4 (Conv2D)       (None, None, None, 512)   2359808

 block4_pool (MaxPooling2D)  (None, None, None, 512)   0

 block5_conv1 (Conv2D)       (None, None, None, 512)   2359808

 block5_conv2 (Conv2D)       (None, None, None, 512)   2359808
```

```
 block5_conv3 (Conv2D)       (None, None, None, 512)   2359808

 block5_conv4 (Conv2D)       (None, None, None, 512)   2359808

 block5_pool (MaxPooling2D)  (None, None, None, 512)   0

 global_average_pooling2d (G  (None, 512)              0
 lobalAveragePooling2D)

 dropout (Dropout)           (None, 512)               0

 dense (Dense)               (None, 120)               61560

=================================================================
Total params: 20,085,944
Trainable params: 61,560
Non-trainable params: 20,024,384
_________________________________________________________________
```

In [ ]:

```
#create callbacks
#earlystop = EarlyStopping(monitor='val_loss', min_delta=0, patience=3, verbose=0, mode='auto')
```

In [ ]:

```
from keras.backend import clear_session
import tensorflow as tf

clear_session()
```

In [ ]:

```
history_few_layer = model.fit(X[:1000], Y[:1000], batch_size=32, epochs=30, validation_split=0.2, verbose
#history_few_layer = model.fit(X[:1000], Y[:1000], batch_size=32, epochs=30, validation_split=0.2, verbose
```

```
Epoch 1/30
25/25 - 167s - loss: 21.1930 - accuracy: 0.0137 - val_loss: 14.0778 - val_accuracy: 0.0100 - 167s/epoch - 7s/step
Epoch 2/30
25/25 - 171s - loss: 15.7852 - accuracy: 0.0550 - val_loss: 11.2936 - val_accuracy: 0.0650 - 171s/epoch - 7s/step
Epoch 3/30
25/25 - 168s - loss: 12.4339 - accuracy: 0.0862 - val_loss: 9.7992 - val_accuracy: 0.0750 - 168s/epoch - 7s/step
Epoch 4/30
25/25 - 170s - loss: 9.4622 - accuracy: 0.1488 - val_loss: 8.4550 - val_accuracy: 0.1000 - 170s/epoch - 7s/step
Epoch 5/30
25/25 - 171s - loss: 7.1724 - accuracy: 0.2387 - val_loss: 7.8881 - val_accuracy: 0.1300 - 171s/epoch - 7s/step
Epoch 6/30
25/25 - 172s - loss: 5.4230 - accuracy: 0.3125 - val_loss: 7.3397 - val_accuracy: 0.1550 - 172s/epoch - 7s/step
Epoch 7/30
25/25 - 172s - loss: 4.3801 - accuracy: 0.3988 - val_loss: 6.8291 - val_accuracy: 0.1900 - 172s/epoch - 7s/step
Epoch 8/30
25/25 - 172s - loss: 3.4505 - accuracy: 0.4638 - val_loss: 6.6711 - val_accuracy: 0.1850 - 172s/epoch - 7s/step
Epoch 9/30
25/25 - 172s - loss: 2.7457 - accuracy: 0.5525 - val_loss: 6.4071 - val_accuracy: 0.2300 - 172s/epoch - 7s/step
Epoch 10/30
25/25 - 172s - loss: 2.3514 - accuracy: 0.5850 - val_loss: 6.5104 - val_accuracy: 0.2150 - 172s/epoch - 7s/step
Epoch 11/30
25/25 - 172s - loss: 2.0081 - accuracy: 0.6187 - val_loss: 6.4689 - val_accuracy: 0.2150 - 172s/epoch - 7s/step
Epoch 12/30
25/25 - 172s - loss: 1.5242 - accuracy: 0.6637 - val_loss: 6.2585 - val_accuracy: 0.2650 - 172s/epoch - 7s/step
Epoch 13/30
25/25 - 172s - loss: 1.4374 - accuracy: 0.7063 - val_loss: 6.2923 - val_accuracy: 0.2500 - 172s/epoch - 7s/step
Epoch 14/30
25/25 - 173s - loss: 1.2496 - accuracy: 0.7362 - val_loss: 6.1994 - val_accuracy: 0.2600 - 173s/epoch - 7s/step
Epoch 15/30
25/25 - 173s - loss: 0.9924 - accuracy: 0.7713 - val_loss: 6.1313 - val_accuracy: 0.2600 - 173s/epoch - 7s/step
Epoch 16/30
25/25 - 174s - loss: 0.7215 - accuracy: 0.8138 - val_loss: 6.2751 - val_accuracy: 0.2500 - 174s/epoch - 7s/step
Epoch 17/30
25/25 - 171s - loss: 0.7598 - accuracy: 0.8250 - val_loss: 6.3245 - val_accuracy: 0.2400 - 171s/epoch - 7s/step
Epoch 18/30
25/25 - 176s - loss: 0.7605 - accuracy: 0.8150 - val_loss: 6.5122 - val_accuracy: 0.2350 - 176s/epoch - 7s/step
Epoch 19/30
25/25 - 174s - loss: 0.6457 - accuracy: 0.8313 - val_loss: 6.4988 - val_accuracy: 0.2400 - 174s/epoch - 7s/step
Epoch 20/30
25/25 - 178s - loss: 0.6752 - accuracy: 0.8350 - val_loss: 6.4070 - val_accuracy: 0.2500 - 178s/epoch - 7s/step
Epoch 21/30
25/25 - 173s - loss: 0.5154 - accuracy: 0.8800 - val_loss: 6.0922 - val_accuracy: 0.2800 - 173s/epoch - 7s/step
```

```
Epoch 22/30
25/25 - 169s - loss: 0.4892 - accuracy: 0.8637 - val_loss: 6.3593 - val_accuracy: 0.2650 - 169s/epoch - 7s
Epoch 23/30
25/25 - 169s - loss: 0.3561 - accuracy: 0.8963 - val_loss: 6.0956 - val_accuracy: 0.2700 - 169s/epoch - 7s
Epoch 24/30
25/25 - 169s - loss: 0.3282 - accuracy: 0.9000 - val_loss: 5.9483 - val_accuracy: 0.2900 - 169s/epoch - 7s
Epoch 25/30
25/25 - 168s - loss: 0.3169 - accuracy: 0.9100 - val_loss: 6.3923 - val_accuracy: 0.2750 - 168s/epoch - 7s
Epoch 26/30
25/25 - 168s - loss: 0.3030 - accuracy: 0.9125 - val_loss: 6.4267 - val_accuracy: 0.2850 - 168s/epoch - 7s
Epoch 27/30
25/25 - 168s - loss: 0.3626 - accuracy: 0.8975 - val_loss: 6.4361 - val_accuracy: 0.2900 - 168s/epoch - 7s
Epoch 28/30
25/25 - 168s - loss: 0.2333 - accuracy: 0.9300 - val_loss: 6.3120 - val_accuracy: 0.3050 - 168s/epoch - 7s
Epoch 29/30
25/25 - 168s - loss: 0.2054 - accuracy: 0.9362 - val_loss: 6.2073 - val_accuracy: 0.3050 - 168s/epoch - 7s
Epoch 30/30
25/25 - 169s - loss: 0.2738 - accuracy: 0.9275 - val_loss: 6.3648 - val_accuracy: 0.3150 - 169s/epoch - 7s
```

Over all we got 93% accuracy.

In [ ]:
```
model.save('vgg19DBPmodel.h5')
```

In [ ]:
```
history_few_layer.history.keys()
```

Out[ ]:
```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

GitHub & Project Demo Link

https://vitapacin-my.sharepoint.com/:v:/g/personal/sreesharan_21bce7003_vitapstudent_ac_in/EY3zLtpcgzZJm9XSy3fjhvkBf_o71c_uYcyU1wwFi-v1KQ?e=SyyZ9J