Report
Classify dog and cat images

1. Datasets

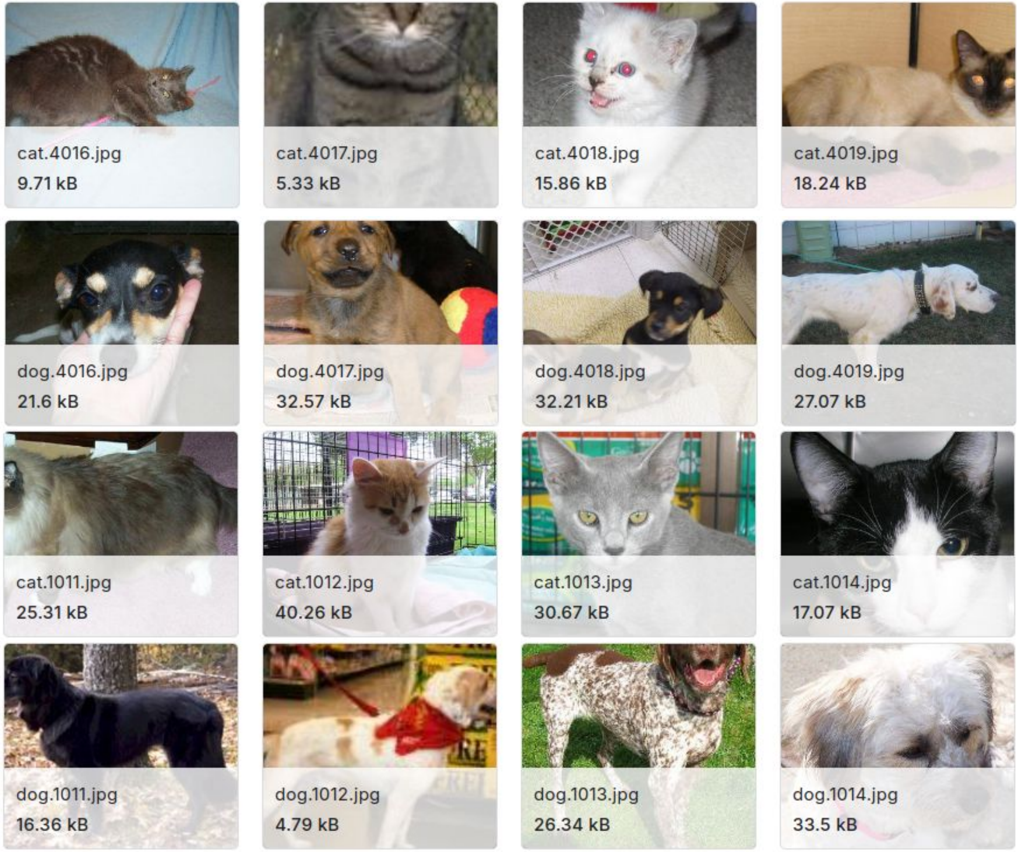
Training: https://www.kaggle.com/datasets/tongpython/cat-and-dog

- test_set
  - test_set
    - cats
    - dogs
- training_set
  - training_set
    - cats
    - dogs

cats: 1011 images
dogs: 1012 images

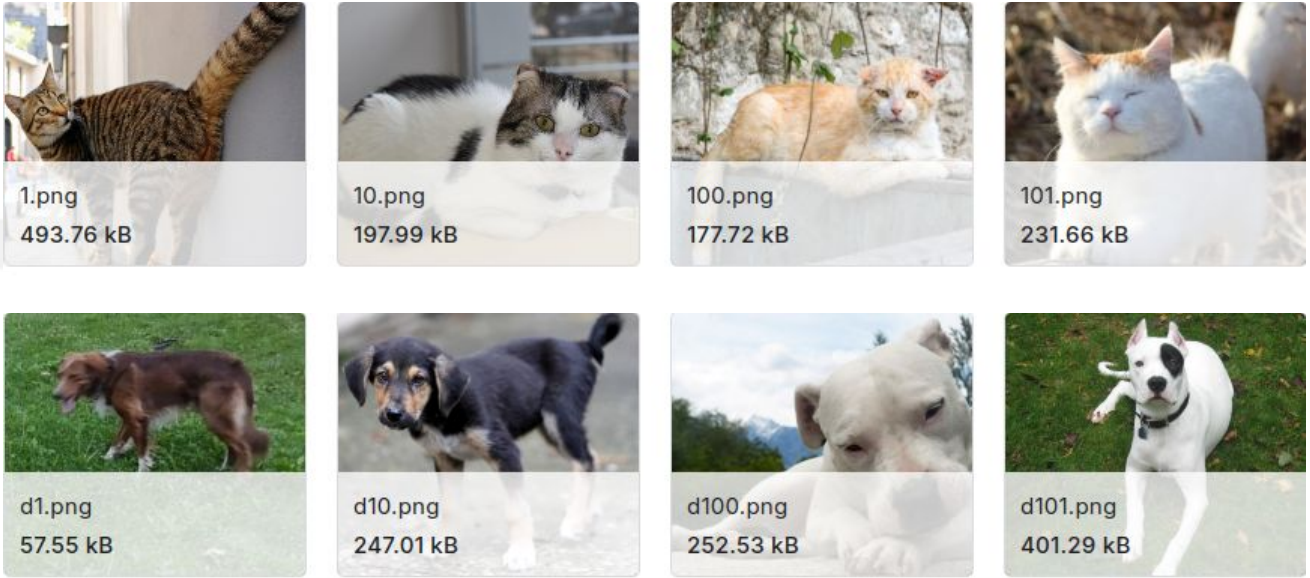
cats: 4000 images
dogs: 4005 images

use for training and valid



1. Datasets

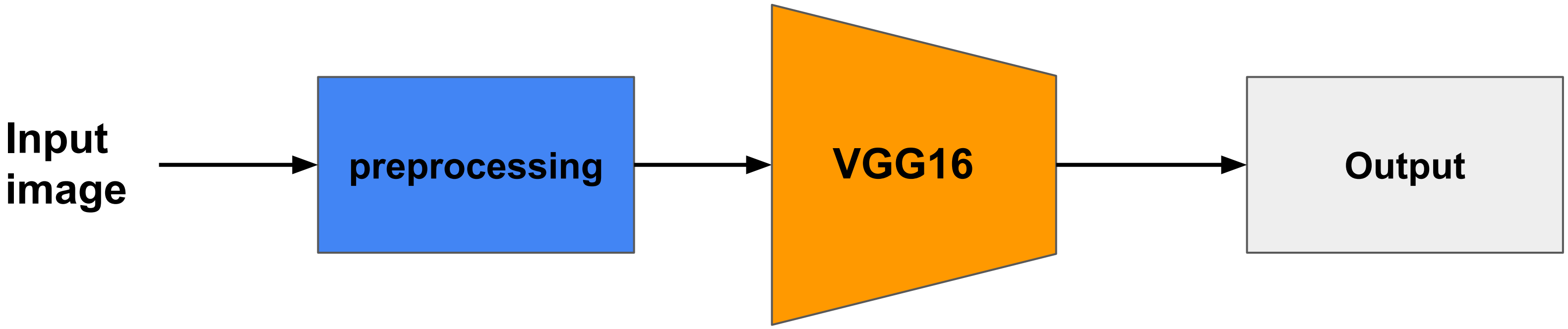
Test: https://www.kaggle.com/datasets/erkamk/cat-and-dog-images-dataset



cats: 500 images
dogs: 500 images

use for testing trained model

2. Pipeline



2. Pipeline

Data Preprocessing:

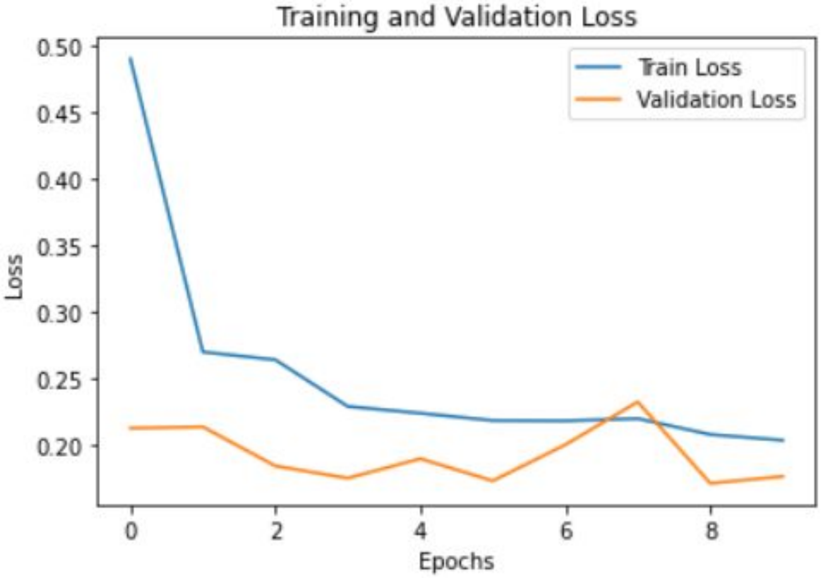
- For training data: Data augmentation techniques like rotation, width/height shifting, shearing, zooming, and horizontal flipping are applied to make the model more robust to various transformations.
- For validation and test data: Only rescaling (normalizing pixel values) is applied to evaluate model performance on clean, untransformed images.

Model Architecture:

- VGG16 is used as the base model, with pre-trained weights on ImageNet. The top (fully connected) layers are removed, and a new classifier with two output classes (`cat`, `dog`) is added. This model is fine-tuned on the new dataset.

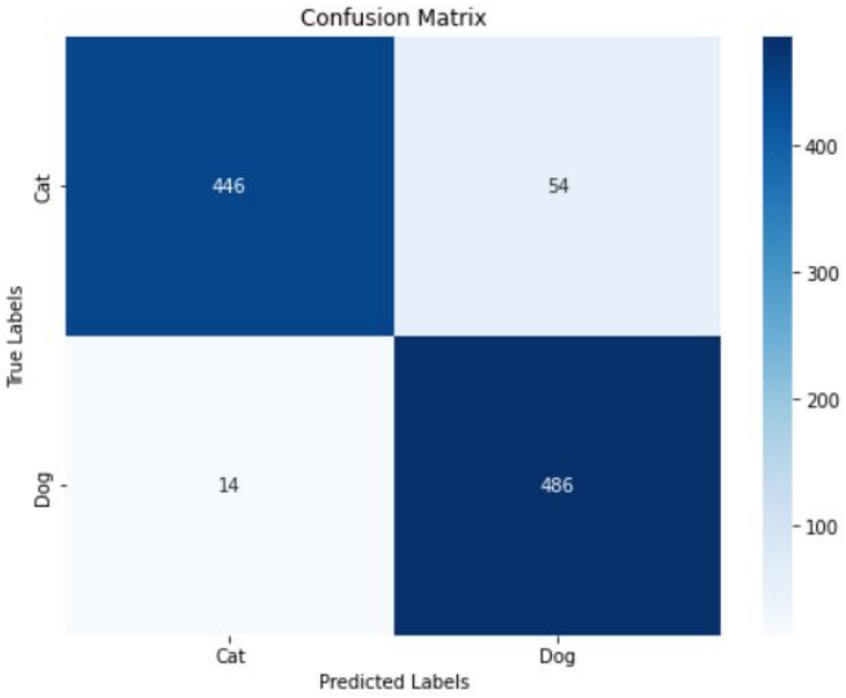
3. Training

- The model is trained using the training data, with loss and accuracy tracked over multiple epochs. The validation data is used to evaluate the model during training and adjust the weights accordingly.
- The `Adam` optimizer is used, with a categorical cross-entropy loss function for multi-class classification.



4. Evaluation

- After training, the model's performance is evaluated on the test set. Metrics such as accuracy and loss are calculated.
- A confusion matrix is plotted to visualize where the model tends to make errors between the `cat` and `dog` classes.




```
Test Loss: 0.17765329778194427
Test Accuracy: 0.9319999814033508
```

5. Problems:

- **Limited Diversity:** The model's performance may not generalize well to unseen or diverse real-world scenarios if the test data is too similar to the training data.
- **Evaluation Metrics:** High accuracy alone may not capture all aspects of model performance

6. Solutions:

- **Real-World Testing:** Evaluate the model under real-world conditions or on different data sets beyond the test set to ensure stable performance in practical scenarios.
- **Data Augmentation:** Consider using data augmentation techniques to enhance generalization and handle unseen situations in the training data.
- **Collect More Data:** Gather additional data to improve model robustness and cover a broader range of scenarios.
- **Additional Evaluation Metrics:** Incorporate metrics such as ROC Curve and AUC to assess the model's performance across different classification thresholds and better understand its behavior, especially in cases of class imbalance.