# Enhancing Object Detection Model Performance

## USING THE OXFORD-IIIT PET DATASET

**Team Eagle GPT**
Varit Kobutra, Monica Joya, Angel Candelas, Aaron David, Saif UR Rehman
Professor Patricia McManus
Houston Community College
26 July, 2024

# INTRODUCTION

**Objective**: Enhance the performance of an object detection model
**Dataset**: Oxford-IIIT Pet Dataset
**Model**: SSD MobileNet V2

## Key Steps:

**Dataset selection** → **Preprocessing** → **Model Training** → **Fine-Tuning** → **Quantization** → **Custom Image Testing**

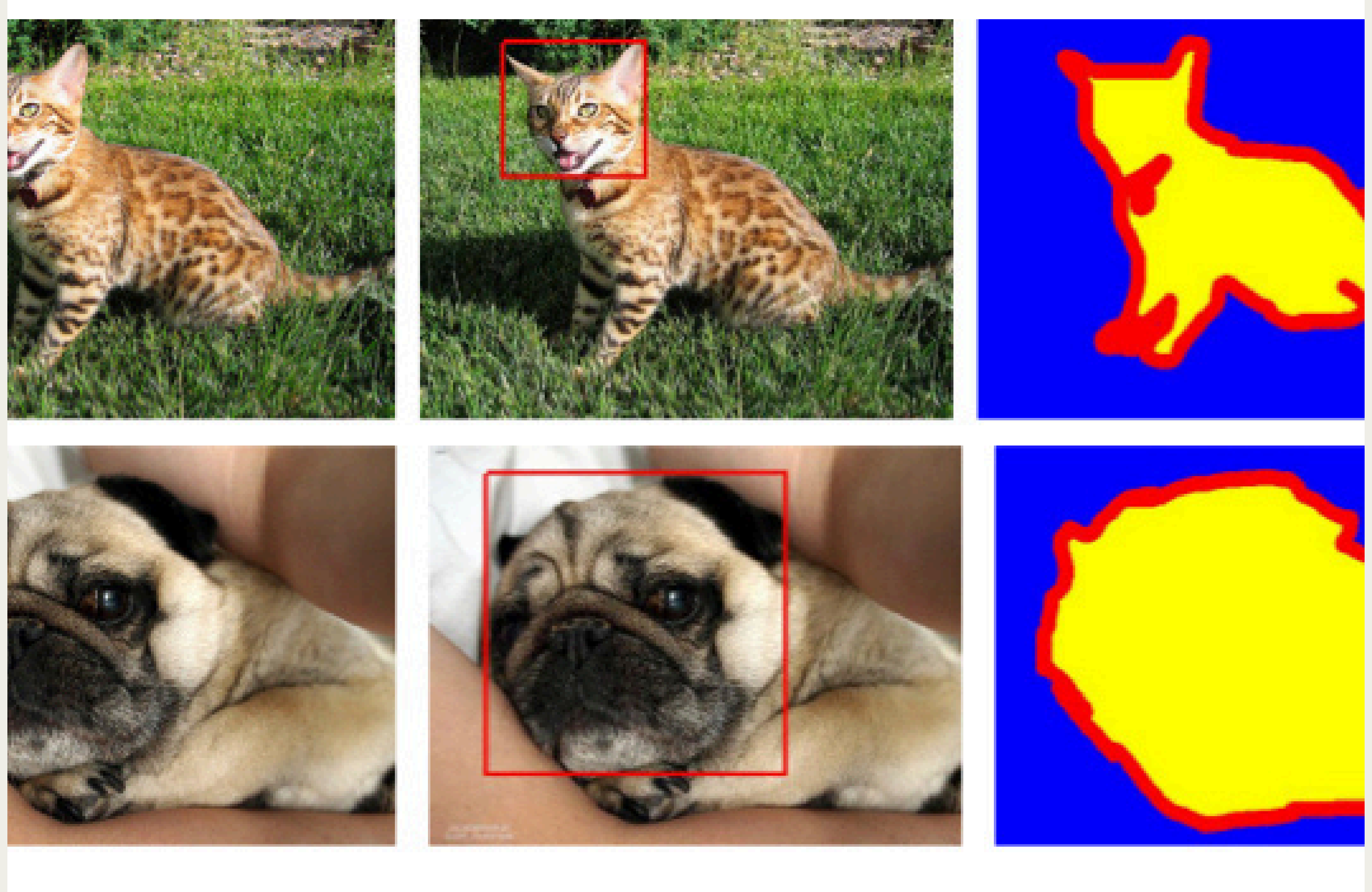
# DATASET SELECTION

## Dataset

*Oxford-IIIT Pet Dataset*

**~7,000**
*images*

**37**
*pet breeds*



## Justification

- *Manageable size for limited computational resources*
- *Diverse and complex enough to provide a challenging task*

| Breed | Count |
|---|---|
| rican Bulldog | 200 |
| rican Pit Bull Terrier | 200 |
| et Hound | 200 |
| le | 200 |
| r | 199 |
| uahua | 200 |
| ish Cocker Spaniel | 196 |
| ish Setter | 200 |
| nan Shorthaired | 200 |
| t Pyrenees | 200 |
| nese | 200 |
| nese Chin | 200 |
| hond | 199 |
| berger | 200 |
| ature Pinscher | 200 |
| Foundland | 196 |
| eranian | 200 |
| | 200 |
| t Bernard | 200 |
| yoed | 200 |
| tish Terrier | 199 |

| Breed | Coun |
|---|---|
| Abyssinian | |
| Bengal | |
| Birman | |
| Bombay | |
| British Shorthair | |
| Egyptian Mau | |
| Main Coon | |
| Persian | |
| Ragdoll | |
| Russian Blue | |
| Siamese | |
| Sphynx | |
| Total | 2 |

2.Cat Breeds

| Family | Coun |
|---|---|
| Cat | 2 |
| Dog | 4 |
| Total | 7 |

3.Total Pets

# DATA PREPROCESSING

**Steps**:

- Resizing images to 128x128 pixels
- Normalizing pixel values to [0, 1] range
- Applying random horizontal flips and brightness adjustments

```
# Data preprocessing and augmentation function
def preprocess_and_augment(data):
    image = tf.image.resize(data['image'], (128, 128))
    image = tf.image.random_flip_left_right(image)
    image = tf.image.random_brightness(image, 0.1)
    image = image / 255.0  # Normalize to [0, 1] range
    return image, data['objects']['label']
```

**Purpose:** Enhance model generalization

# MODEL ARCHITECTURE

**Base Model:** SSD MobileNet V2

**Modifications:**

- Added global average pooling layer
- Dense layer with ReLU activation
- Dropout layer for regularization
- Final dense layer for classification

**Compilation:** Adam optimizer, sparse categorical cross-entropy loss, accuracy metric

```
# Model architecture
base_model = tf.keras.applications.MobileNetV2(input_shape=(128, 128,
3),
                                               include_top=False,
                                               weights='imagenet')

base_model.trainable = False

model = models.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(1024, activation='relu'),
    layers.Dropout(0.2),
    layers.Dense(37, activation='softmax')
])

model.compile(optimizer=tf.keras.optimizers.Adam(),
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

# TRAINING AND EVALUATION

**Training:**

- **Epochs:** 10
- **Metrics:** Training loss, training accuracy, validation loss, validation accuracy

**Results:**

- **Baseline Accuracy:** Improved from 68.15% to 79.56%
- **Fine-tuned Accuracy:** Started at 83.72% and stabilized at 79.50%

```
# Train the model
history = model.fit(train_dataset,
                    validation_data=test_dataset,
                    epochs=10)

# Evaluate the model
loss, accuracy = model.evaluate(test_dataset)
print(f'Test accuracy: {accuracy:.2f}')
```

# FINE TUNING

**Method:**

- Unfreezing some layers of the base model
- Training with a lower learning rate

**Results:**

- Significant improvement in accuracy

```python
# Load and fine-tune the model
fine_tune_at = 100
base_model.trainable = True

for layer in base_model.layers[:fine_tune_at]:
    layer.trainable = False

model.compile(optimizer=tf.keras.optimizers.Adam(1e-5),
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

history_fine = model.fit(train_dataset,
                         validation_data=test_dataset,
                         epochs=10)

# Evaluate the fine-tuned model
loss, accuracy = model.evaluate(test_dataset)
print(f'Test accuracy after fine-tuning: {accuracy:.2f}')
```

# MODEL QUANTIZATION

**Purpose:** Reduce model size and increase inference speed

**Method:** Post-training quantization

**Tool:** TensorFlow Lite

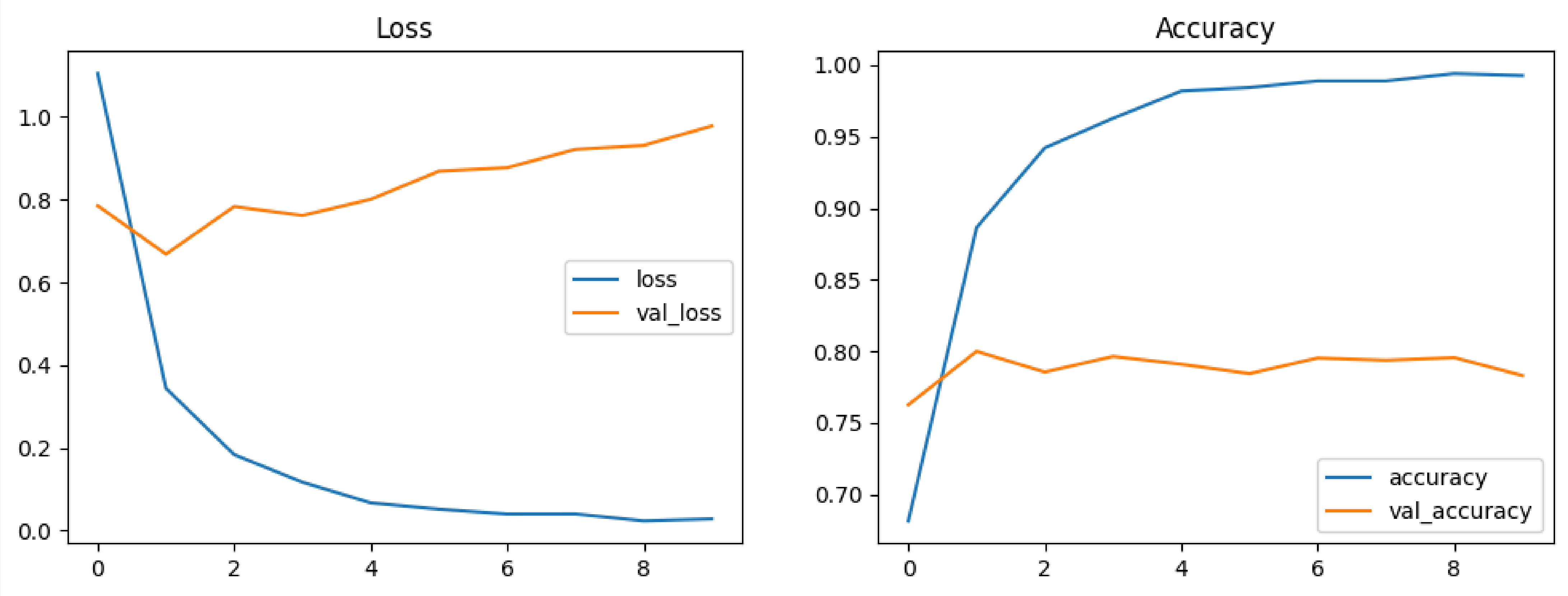
```python
# Convert the model to TensorFlow Lite format with quantization
converter = tf.lite.TFLiteConverter.from_keras_model(model)
converter.optimizations = [tf.lite.Optimize.DEFAULT]
tflite_quant_model = converter.convert()

# Save the quantized model
with open('model_quant.tflite', 'wb') as f:
    f.write(tflite_quant_model)
```

# PERFORMANCE METRICS

**Baseline Training Results**

- **Epoch 1/10**:
  - **Training Loss**: 1.1051
  - **Training Accuracy**: 68.15%
  - **Validation Loss**: 0.7854
  - **Validation Accuracy**: 76.26%
- **Epoch 10/10**:
  - **Training Loss**: 0.0278
  - **Training Accuracy**: 99.27%
  - **Validation Loss**: 0.9785
  - **Validation Accuracy**: 78.30%



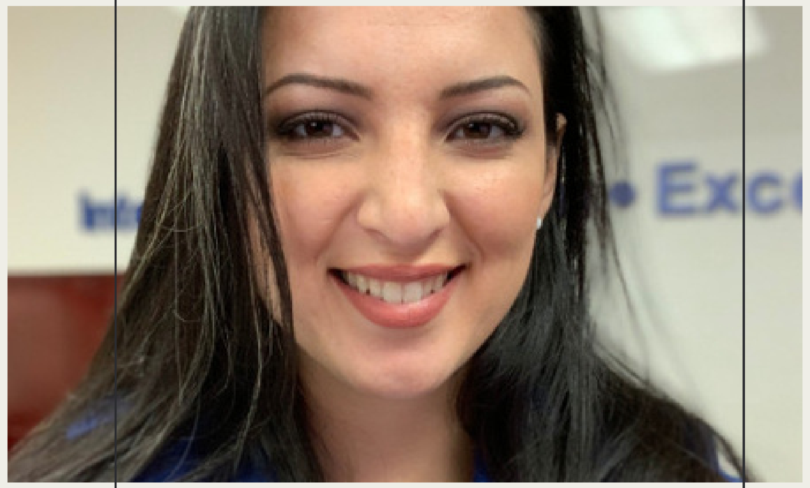

The enhanced model showed improved accuracy and reduced model size compared to the baseline model.

# MEET THE TEAM

*Planning & Advice*

**Varit (Henry) Kobutra**

Henry focused on ensuring the project stayed on track and provided guidance on technical challenges.




*Creative & Code*

**Monica Joya**

Monica led the presentation and visual design, and handled significant portions of the coding tasks.

*Documentation*

**Angel Candelas**

Angel was responsible for preparing detailed documentation and ensuring clarity and coherence in the project reports.

*Research & Review*

**Aaron David & Saif UR Rehman**

Aaron and Saif conducted thorough research and critical evaluation of information, supporting the team with accurate and quality content.

# CONCLUSION

The project effectively enhanced model performance through data preprocessing, fine-tuning, and quantization. These methods resulted in improved accuracy and efficiency, demonstrating practical applicability for object detection tasks.

# Thank you!

## AN OBJECT DETECTION PRESENTATION

**Team Eagle GPT**
Varit Kobutra, Monica Joya, Angel Candelas, Aaron David, Saif UR Rehman
Professor Patricia McManus
Houston Community College
26 July, 2024

# REFERENCES


- Chollet, F. (2018). *Deep Learning with Python. Manning Publications*.
- TensorFlow Datasets. (n.d.). *Oxford-IIIT Pet Dataset*. Retrieved from https://www.tensorflow.org/datasets/catalog/oxford_iiit_pet
- Parkhi, O. M., Vedaldi, A., Zisserman, A., & Jawahar, C. V. (2012). Cats and Dogs. *IEEE Conference on Computer Vision and Pattern Recognition*.


HCC