- Transformer "VIT"
    - Images were resized to 224×224.
    - Augmentations included:
        - Random horizontal and vertical flips
        - Color jittering
        - Random affine transformations

Include a screenshot or example of the dataset before and after augmentation if possible.

3. Model Description

- **Model Used**: Vision Transformer (ViT)
    - **Base Architecture**: vit_base_patch16_224 pretrained on ImageNet.
    - **Customizations**:
        - Replaced the classification head to match 18 animal classes.
        - Fine-tuned layers for improved performance.
- **Key Components**:
    - Patch Embedding
    - Multi-Head Attention
    - MLP Layers
    - Classification Head

4. Training and Evaluation

- **Data Splits**:
    - 80% for training, 20% for testing.
- **Hyperparameters**:
    - Optimizer: Adam
    - Learning Rate:
        - 1×10^−3 for the classification head
        - 1×10^−5 for other layers
    - Scheduler: CosineAnnealingLR with Tmax=10
    - Batch Size: 4
    - Epochs: 5
    - Gradient Clipping: 1.0
- **Training Process**:
  The model was trained for 5 epochs with early stopping based on validation accuracy.

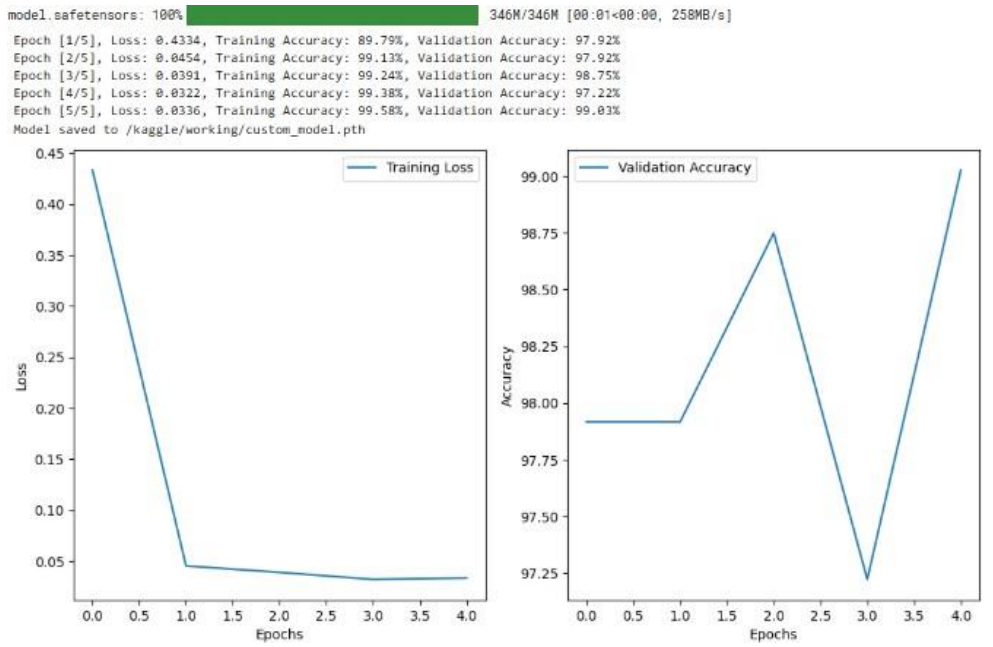
Include training and validation accuracy plots, as well as any screenshots of the process.

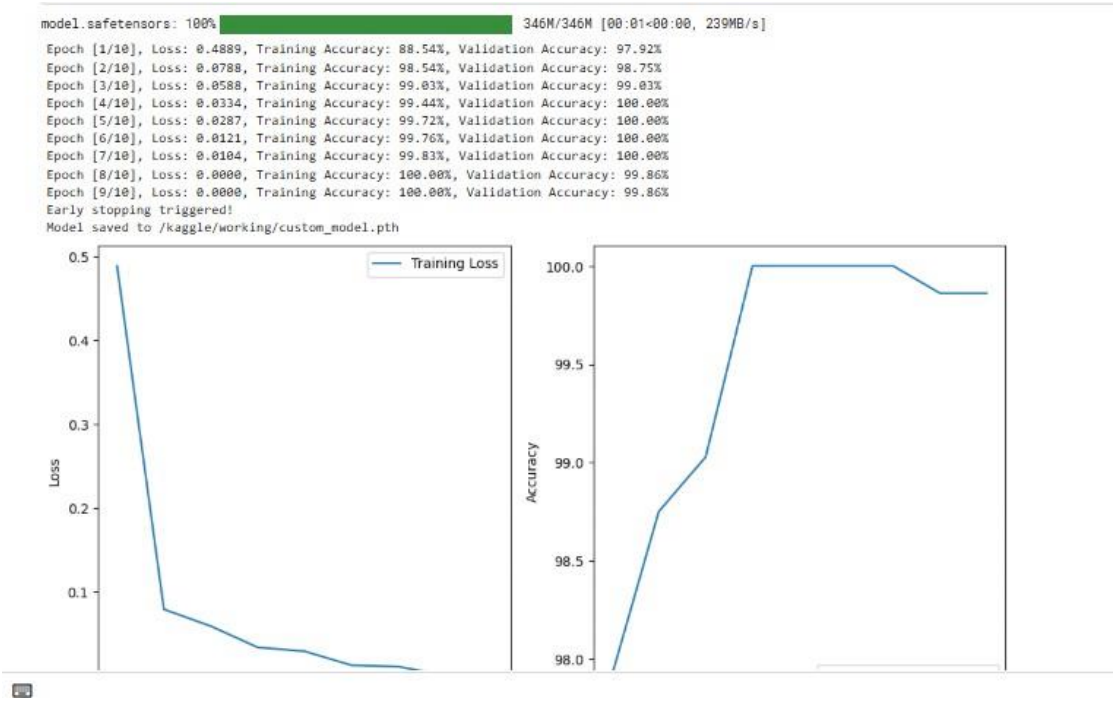
---

5. Results

- **Performance Metrics**:
    - Final Training Accuracy: 99.58%
    - Final Validation Accuracy: 99.86%
    - Best Validation Accuracy Achieved: 99.86%
- **Confusion Matrix**:
  Optional: Include a confusion matrix to show class-wise performance.
- **Training Curves**:

- Loss vs Epoch
- Accuracy vs Epoch





6. Challenges and Solutions

- **Class Imbalance:**
  Balanced the dataset by augmenting underrepresented classes.

- **Overfitting**:
    - Used dropout layers.
    - Applied data augmentations.
- **Computational Limitations**:
    - Adjusted batch size to fit within memory constraints.

---

7. **Conclusion**

- **Summary**:
  The project successfully classified animal species using a Vision Transformer model, achieving a validation accuracy of 99.86%.

Architecture

Vision Transformer (ViT) Architecture

1. Input Features:
   Images ((batch_size,H,W,C)are divided into P×PP \times PP×P patches. Each patch is flattened and projected to a vector of dimension dmodel, resulting in shape:(batch_size,seq_len,dmodel).
2. Positional Encoding:
   Adds positional information to the patch embeddings. Output retains shape: (batch_size,seq_len,dmodel).
3. Transformer Encoder Layers:
   Passes through #layers consisting of:
    - Multi-Head Self-Attention (MHSA)
    - Feedforward Neural Network (MLP)
    - Residual connections and Layer Normalization (batch_size,seq_len,dmodel).

4. Global Average Pooling:
   Aggregates sequence representations into a single(batch_size,dmodel).
5. Fully Connected Layer:
   Maps pooled features to class logits:(batch_size,#classes)

2-Resnet

Dataset

- **Input Data**: The dataset includes labeled images of animals categorized into 18 classes such as "beaver," "butterfly," and "elephant."
- **Augmentation**: To handle class imbalance, the dataset was augmented by oversampling underrepresented classes.
- **Transformation**: Applied the following:
  - Random horizontal and vertical flips.
  - Random rotation, color jitter, and perspective distortion.
  - Resizing images to 224×224224 \times 224224×224, followed by normalization.
- **Split**: 80% of the data was used for training, and 20% for validation.

Model Architecture

1. **Base Architecture**: ResNet50.
   - Pretrained weights on ImageNet were used to initialize the network.
2. **Modifications**:
   - The final fully connected layer was replaced with one tailored to predict 18 classes.

- Custom ResNet blocks were implemented using a bottleneck residual block structure.

Input Layer

- **Input Shape**:
  Images resized to 224×224 with 3 channels (RGB).
  Input tensor shape (batch size,3,224,224).

---

2. Initial Convolution Block

- **Convolutional Layer**:
  A7×7 kernel with 64 filters, stride 2, and padding, extracting basic features from input images.
- **Batch Normalization**:
  Applied to normalize feature maps for stable training.
- **Activation**:
  ReLU activation introduces non-linearity.
- **Max Pooling**:
  A3×3 max pooling operation with stride 2 reduces spatial dimensions.

---

3. Residual Blocks

ResNet50 is built using **bottleneck residual blocks**, which allow the model to learn identity mappings efficiently, mitigating the vanishing gradient problem. Each block comprises:

1. **Bottleneck Design**:
   - **1x1 Convolution**: Reduces the feature dimensionality.
   - **3x3 Convolution**: Extracts features with a spatial focus.
   - **1x1 Convolution**: Restores the dimensionality for compatibility with residual connections.

.Batch Normalization:

- Follows each convolution to stabilize learning.

.ReLU Activation:

- Adds non-linearity after each convolutional layer.

.Residual Connection:

- Adds the input (identity) to the block's output if dimensions align.
- For mismatched dimensions, a **downsample layer** is used to align input and output shapes.

4. Block Configuration

The ResNet50 model has four main stages, each with an increasing number of filters and decreasing spatial dimensions:

5. Global Average Pooling

- Reduces each feature map into a single value by averaging its spatial dimensions.
- Output shape: (batch size,512×4=2048)(\text{batch size}, 512 \times 4 = 2048)(batch size,512×4=2048).

6. Fully Connected Layer

- The final layer maps the 2048-dimensional feature vector to 18 logits (one for each class).
- Softmax activation is implicitly applied during training via cross-entropy loss.

Training and Evaluation

- **Hyperparameters**:
    - Optimizer: Adam with a learning rate of 1×10−51 \times 10^{-5}1×10−5.
    - Loss Function: Cross-Entropy Loss.
    - Scheduler: Learning rate decreased by a factor of 0.5 every 7 epochs.
    - Batch Size: 4.
- **Training**:
    - Model trained for a maximum of 10 epochs.
    - Early stopping was implemented based on validation accuracy and overfitting checks.
- **Metrics**:
    - Training loss and accuracy.
    - Validation accuracy to monitor generalization.

Results

- **Training Performance**:
    - Training loss consistently decreased, indicating proper learning by the model.
    - Training accuracy steadily increased over epochs.
- **Validation Performance**:
    - Best validation accuracy achieved: **GG.44%**
    - Average validation accuracy across all epochs: **G7.47%**
- **Early Stopping**:
    - Training was stopped early due to no improvement in validation accuracy for 7 consecutive epochs or detection of overfitting.

```
Epoch [1/10], Loss: 1.9643, Training Accuracy: 62.74%, Validation Accuracy: 91.39%
Epoch [2/10], Loss: 0.8478, Training Accuracy: 86.01%, Validation Accuracy: 95.83%
Epoch [3/10], Loss: 0.4755, Training Accuracy: 92.19%, Validation Accuracy: 97.78%
Epoch [4/10], Loss: 0.3268, Training Accuracy: 94.41%, Validation Accuracy: 97.64%
Epoch [5/10], Loss: 0.2387, Training Accuracy: 95.56%, Validation Accuracy: 98.61%
Epoch [6/10], Loss: 0.1855, Training Accuracy: 96.70%, Validation Accuracy: 99.17%
Epoch [7/10], Loss: 0.1665, Training Accuracy: 97.40%, Validation Accuracy: 99.03%
Epoch [8/10], Loss: 0.1427, Training Accuracy: 97.50%, Validation Accuracy: 99.03%
Epoch [9/10], Loss: 0.1250, Training Accuracy: 97.88%, Validation Accuracy: 99.44%
Epoch [10/10], Loss: 0.1064, Training Accuracy: 98.19%, Validation Accuracy: 99.44%
Average Validation Accuracy: 97.74%
```

-
  -

Key Observations

1. **Class Balancing**: Augmentation effectively balanced the dataset, leading to improved performance across all classes.
2. **Data Augmentation**: Improved robustness to transformations and prevented overfitting.
3. **Early Stopping**: Prevented overfitting, ensuring the model generalized well to unseen data.

Challenges

1. **Limited Batch Size**: A batch size of 4 may have led to slower convergence due to gradient noise.
2. **Overfitting Risk**: Overfitting was observed in some epochs, as indicated by the gap between training and validation accuracy.

```
Epoch 1/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 12s 73ms/step - accuracy: 0.0516 - loss: 2.8567 - val_accuracy: 0.0379 - val_loss: 2.9001 - learning_rate: 1.0000e-05
Epoch 2/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 61ms/step - accuracy: 0.0586 - loss: 2.8533 - val_accuracy: 0.1034 - val_loss: 2.8828 - learning_rate: 1.0000e-05
Epoch 3/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 61ms/step - accuracy: 0.0754 - loss: 2.8636 - val_accuracy: 0.0517 - val_loss: 2.8671 - learning_rate: 1.0000e-05
Epoch 4/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 61ms/step - accuracy: 0.0851 - loss: 2.7819 - val_accuracy: 0.0793 - val_loss: 2.8319 - learning_rate: 1.0000e-05
Epoch 5/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 61ms/step - accuracy: 0.0936 - loss: 2.7803 - val_accuracy: 0.1103 - val_loss: 2.7848 - learning_rate: 1.0000e-05
Epoch 6/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 61ms/step - accuracy: 0.1206 - loss: 2.7364 - val_accuracy: 0.0828 - val_loss: 2.7578 - learning_rate: 1.0000e-05
Epoch 7/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.1059 - loss: 2.6729 - val_accuracy: 0.1448 - val_loss: 2.7270 - learning_rate: 1.0000e-05
Epoch 8/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.1688 - loss: 2.6415 - val_accuracy: 0.2000 - val_loss: 2.6915 - learning_rate: 1.0000e-05
Epoch 9/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 61ms/step - accuracy: 0.1906 - loss: 2.6009 - val_accuracy: 0.2310 - val_loss: 2.6748 - learning_rate: 1.0000e-05
Epoch 10/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 59ms/step - accuracy: 0.1758 - loss: 2.5965 - val_accuracy: 0.1966 - val_loss: 2.6010 - learning_rate: 1.0000e-05
Epoch 11/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.2310 - loss: 2.5518 - val_accuracy: 0.2172 - val_loss: 2.5993 - learning_rate: 1.0000e-05
Epoch 12/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.2113 - loss: 2.6190 - val_accuracy: 0.2448 - val_loss: 2.5655 - learning_rate: 1.0000e-05
Epoch 13/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.2112 - loss: 2.5817 - val_accuracy: 0.2241 - val_loss: 2.5482 - learning_rate: 1.0000e-05
Epoch 14/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.2101 - loss: 2.5911 - val_accuracy: 0.2207 - val_loss: 2.5042 - learning_rate: 1.0000e-05
Epoch 15/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 2s 34ms/step - accuracy: 0.2555 - loss: 2.5866 - val_accuracy: 0.2483 - val_loss: 2.5340 - learning_rate: 1.0000e-05
Epoch 16/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 61ms/step - accuracy: 0.2470 - loss: 2.4181 - val_accuracy: 0.2655 - val_loss: 2.4675 - learning_rate: 1.0000e-05
Epoch 17/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 5s 62ms/step - accuracy: 0.2416 - loss: 2.4878 - val_accuracy: 0.2586 - val_loss: 2.4561 - learning_rate: 1.0000e-05
Epoch 18/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.2057 - loss: 2.5763 - val_accuracy: 0.2552 - val_loss: 2.4386 - learning_rate: 1.0000e-05
Epoch 19/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.2763 - loss: 2.4464 - val_accuracy: 0.2655 - val_loss: 2.4145 - learning_rate: 1.0000e-05
Epoch 20/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 61ms/step - accuracy: 0.2462 - loss: 2.5223 - val_accuracy: 0.2586 - val_loss: 2.4076 - learning_rate: 1.0000e-05
Epoch 21/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.2362 - loss: 2.4940 - val_accuracy: 0.2552 - val_loss: 2.3858 - learning_rate: 1.0000e-05
Epoch 22/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 2s 33ms/step - accuracy: 0.2708 - loss: 2.4287 - val_accuracy: 0.2793 - val_loss: 2.3925 - learning_rate: 1.0000e-05
Epoch 23/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.2618 - loss: 2.4522 - val_accuracy: 0.2759 - val_loss: 2.3687 - learning_rate: 1.0000e-05
Epoch 24/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 2s 33ms/step - accuracy: 0.2643 - loss: 2.3594 - val_accuracy: 0.2552 - val_loss: 2.3945 - learning_rate: 1.0000e-05
Epoch 25/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 2s 34ms/step - accuracy: 0.2661 - loss: 2.4544 - val_accuracy: 0.2690 - val_loss: 2.4061 - learning_rate: 1.0000e-05
Epoch 26/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 61ms/step - accuracy: 0.2552 - loss: 2.4519 - val_accuracy: 0.2793 - val_loss: 2.3275 - learning_rate: 1.0000e-05
Epoch 27/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 2s 33ms/step - accuracy: 0.2635 - loss: 2.4130 - val_accuracy: 0.2897 - val_loss: 2.3281 - learning_rate: 1.0000e-05
Epoch 28/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.3160 - loss: 2.2987 - val_accuracy: 0.2966 - val_loss: 2.3196 - learning_rate: 1.0000e-05
Epoch 29/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 4s 60ms/step - accuracy: 0.2932 - loss: 2.3586 - val_accuracy: 0.2862 - val_loss: 2.2905 - learning_rate: 1.0000e-05
Epoch 30/30
73/73 ━━━━━━━━━━━━━━━━━━━━ 2s 34ms/step - accuracy: 0.2897 - loss: 2.3475 - val_accuracy: 0.2862 - val_loss: 2.3505 - learning_rate: 1.0000e-05
Model training complete and saved.
```