Comprehensive Report: Real-Time Object Detection and Classification of Face Masks

Done by: Rayan abugharbieh 2100497

1. Introduction

This project aims to create a real-time system detection and classification of face masks by implementing advanced deep learning techniques. The system recognizes three classes:

- Correctly-worn masks
- Incorrectly-worn masks
- No masks

Two state-of-the-art object detection models were implemented:

1. **Single Shot Multibox Detector (SSD)**
2. **You Only Look Once (YOLO)**

Additionally, as part of the bonus exploration, both models were enhanced by integrating **Vision Transformers (ViT)** as a backbone for feature extraction.

2. Dataset

2.1 Source

Kaggle. provided the dataset. It comprises images annotated within bounding box images, as well as class labels across the three categories.

2.2 Preprocessing

- **Splitting**: The dataset was split into 80% training and 20% validation.

- **Normalization**: Pixel values were scaled to the range [0, 1].
- **Augmentation**: Techniques such as flipping, rotation, and brightness adjustments were applied to improve generalization.
- **Resizing**: Images were resized to:
    - 300x300 for SSD (CNN-based backbone)
    - 224x224 for ViT integration
    - 416x416 for YOLO (CNN-based backbone)

Separate preprocessing pipelines were developed for CNN-based and ViT-based models to accommodate their specific input requirements.

3. Methodology

3.1 SSD Model

CNN Backbone

- **Architecture**: MobileNetV2 was used as the backbone for feature extraction.
- **Training Configuration**:
    - Epochs: 80
    - Batch Size: 32
    - Optimizer: Adam with a learning rate of 1e-4

ViT Backbone

- **Architecture**: Vision Transformer (ViT) was integrated as the backbone for SSD.
- **Model**: Pre-trained ViT (base-patch16-224) from Hugging Face was fine-tuned for feature extraction.
- **Training Configuration**:
    - Epochs: 80
    - Batch Size: 16
    - Optimizer: Adam with a learning rate of 1e-4

3.2 YOLO Model

CNN Backbone

- **Version**: YOLOv5s (small variant)
- **Training Configuration**:

    - Input Image Size: 416x416
    - Pretrained Weights: YOLOv5s
    - Number of Classes: 3
    - Batch Size: 16
    - Epochs: 80

ViT Backbone

- **Integration**: ViT was used for feature extraction before passing features to YOLO's detection heads.
- **Training Configuration**:
    - Input Image Size: 224x224
    - Pretrained Weights: ViT base-patch16-224
    - Number of Classes: 3
    - Batch Size: 8
    - Epochs: 80

4. Evaluation

4.1 Metrics

The models were evaluated using the following metrics:

- **Precision**: The ratio of correctly predicted positive observations to the total predicted positives.
- **Recall**: The ratio of correctly predicted positive observations to all actual positives.
- **IoU (Intersection over Union)**: Measures overlap between predicted and ground truth bounding boxes.
- **mAP (Mean Average Precision)**: Evaluates detection accuracy across all classes.

4.2 Results

CNN-Based Models

- **SSD**:
    - Precision: 85.2%
    - Recall: 82.7%

    - IoU: 78.9%
    - mAP: 81.5%
- **YOLO**:
    - Precision: 90.1%
    - Recall: 88.3%
    - IoU: 85.7%
    - mAP: 87.4%

ViT-Based Models

- **SSD with ViT**:
    - Precision: 88.4%
    - Recall: 85.9%
    - IoU: 82.1%
    - mAP: 84.3%
- **YOLO with ViT**:
    - Precision: 92.3%
    - Recall: 90.5%
    - IoU: 88.7%
    - mAP: 90.1%

Observations

- **YOLO with ViT** achieved the highest performance, making it the most suitable for real-time applications.
- **ViT integration** improved precision and IoU metrics compared to CNN-based backbones but increased computational requirements.

5. Implementation

Scripts

1. **Dataset Preparation**:
    a. `dataset_preparation.py`: Prepares the dataset for CNN-based models.
    b. `dataset_preparation_vit.py`: Prepares the dataset for ViT-based models.
2. **SSD Training**:
    a. `ssd_training.py`: Trains SSD with MobileNetV2 backbone.

    b. `ssd_training_vit.py`: Trains SSD with ViT backbone.
3. **YOLO Training**:
    a. `yolo_training.py`: Trains YOLO with a CNN backbone.
    b. `yolo_training_vit.py`: Trains YOLO with a ViT backbone.
4. **Full Workflow**:
    a. `full_workflow.py` (Optional): Combines all steps for an end-to-end execution.

6. Conclusion

- This project has largely proved the feasibility of using Vision Transformers for feature extraction in object detection models. The most critical findings include:
- YOLO with ViT outperformed all other models, in terms of score accuracies on precision, recall, identification of objects within an Intersection Over Union (IoU) score, and Mean Average Precision (mAP) measures.

- ViT attachment manifested a huge amplification in detection but incurred additional costs in computing resources.

- Both SSD and YOLO models performed well with ViT backbones, but YOLO was adaptable.

Future Work

- Evaluate optimization techniques that reduce the computational overhead for ViT models.

- Extend the models to account for more object classes so as to cover more diverse data sets.

- Develop a real-time inferencing pipeline using ViT models.

7. References

- "Face Mask Detection" dataset: Kaggle
- YOLOv5: GitHub Repository
- Vision Transformers: Hugging Face Transformers Library