Diffusion for Object Detection

`Smartathon: The Smart Cities Challenge` : `Theme 1` Denoise predictions from Object Detection models using **diffusion-based** sequential denoising. Project Link: https://github.com/sahagar/SDAIA_Hack_Theme1

Project Overview

Large-Scale real-world datasets often contain significantly more noise than academic benchmark datasets. Diffusion can be used as a denoising algorithm to further align and generalize model performance.

- Use `Yolov6` as the base *object detection* framework and train it on processed Theme 1 dataset.
- Predictions from `Yolov6` are used as the initial proposals for the diffusion model which uses a `Resnet-50` backbone for feature extraction.
- Train the diffusion model to generate **corrected** proposals by denoising the initial proposals over several time-stamps.

Generating Submission File

- Build and run docker using provided DockerFile
  - `docker build -f DockerFile -t sdaiahack:latest .`
  - `docker run --rm --gpus all -v <path-to-dataset-directory>:/data -v <path-to-model-directory>:/output --name sdaiahack-container -it sdaiahack:latest`
  - Assumes `<path-to-dataset-directory>` contains unzipped `Theme1_dataset` and is mounted read/write

    ```
    ├── <PATH-TO-DATASET-DIRECTORY>/
    │   ├── THEME1_DATASET/
    │   │   │   ├── IMAGES/
    │   │   │   ├── 0A1EA4614A9DF912EEB8D1B40BFFEE74`.JPG
    │   │   │   ├── 0A2BC0DC2371794509F4B776AFF0DD88.JPG
    │   │   │   ├── ...
    │   │   │   └── 0A82E45ED11FB9EF1620A0B40CD9F6D8.JPG
    │   │   ├── SAMPLE_SUBMISSION.CSV
    │   │   ├── TEST.CSV
    │   │   └── TRAIN.CSV
    ```
  - Assumes `<path-to-model-directory>` is mounted read/write

- Download trained model from checkpoint to `<path-to-model-directory>/yolov6_train_output/exp/weights/best_ckpt.pt`
  - `mkdir -p <path-to-model-directory>/yolov6_train_output/exp/weights`
  - md5 checksum: `8f379b2e998ffc80c51e625496135cb0`
- Run `evaluate_yolov6.sh` from `/workspace` inside docker. `submission.csv` will be generated in `/output/yolov6_evaluation_output`
- By default, it evaluates `test` images

Training

NOTE: All scripts executed from `/workspace` inside docker

```
# Preprocess Theme 1 dataset
bash preprocess_data.sh

# Train Yolo
bash train_yolov6.sh

# Evaluate Yolo on train images
bash evaluate_yolov6_for_diffusion.sh

# Train Diffusion
bash train_diffusion.sh
```

Challenges during Data preparation

- Annotations were noisy by design which made the Object Detection process very challenging.
- Adding TTA (Test-Time Augmentation) to both Yolo and Diffusion model evalutaion and image transformations to training reduced the error rate.

Towards Future Scalability

- The proposed framework is automated and scalable to large datasets.
- Optimizing the inference stage for both Yolov6 and Diffusion using acceleration frameworks like `TensorRT` and `ONNX` would be required to make this approach deployable.
- Distillation to smaller model sizes would also be crucial in deployment.

Future direction

I would focus on the following areas of improvement:

- Adding diffusion denoising to Yolo training directly. My current setup is a 2-stage approach which likely causes performance degradation by not optimizing end-2-end.
- RLHF (Reinforcement Learning from Human Feedback) has shown great promise in language and image generation to align model outputs with human preference. This approach could be useful in aligning the object detection models for noisy datasets. A reward model can be created using only a small amount of Feedback data and can scale to large datasets and help with model generality.

Open Source Software

- DiffusionDet
- Yolov6
- Detectron2
- cv2
- docker