HI! Paris Technical Test Notes

Tim Luka Horstmann

Code: code_TimLukaHorstmann.ipynb
Experiments: https://wandb.ai/tlh45/HIParis

Approach

This project aims to develop a computer vision (CV) model capable of detecting fire-related objects in images. The dataset consists of historical fire images taken from towers within different forests and labeled with fire locations. This project's ultimate (practical) goal is to support the (real-time) early fire detection efforts of the French firefighters.

As a starting point for this project, I created a subset of the original dataset, which only contains 10% of the original data to enable efficient training in a limited resource environment. To quickly reach strong results, I rely on the newest model of the YOLO architecture (YOLO11) - a series of real-time object detection models providing state-of-the-art (SOTA) performance on many object detection tasks. As computing hardware, Google Colab's T4 GPU is employed. Since work is conducted in the Colab cloud environment, I chose wandb (after initial experiments with MLFlow) for experiment tracking to provide a simpler integration into the cloud setting.

For most tasks, I leveraged the functions provided by the YOLO library while explainability functionality was achieved via pytorch-grad-cam. The latter required the development of a wrapper model around the fine-tuned/custom YOLO model to extract the activations from the specified target layer of the underlying YOLO model. Grad-CAM can then be used to visualize the parts of the image relevant to the specified layer during inference.

Two "yolo11n" models were fine-tuned on the provided dataset, with training for 50 and 100 epochs respectively. For this initial task, I simply saved the final model. In a real-world setting (i.e. with more time for the project), I would not only save the best model based on a validation metric but also experiment with different architectures and hyperparameters (see below). An analysis of the validation losses (box, cls, and dfl) during training as well as the performance of the model on the validation dataset after training reveals a similar performance for both models. It should be noted that the models display higher precision than recall, suggesting that they perform well in making accurate predictions once smoke is detected but struggle to identify all instances of smoke (wildfires). Increasing the dataset size would likely help improve the recall of the model by providing more diverse training samples.

What would I do if I had one month to work on the project?

If I had one month to work on this project, I would work on expanding its scope, focus on improving the performance of the object detection model, and set up an end-to-end deployment pipeline to enable usage of the model in real life (i.e. provide the French firefighters with the technical means to automatically detect fires using the YOLO model). The specific details of what should be achieved are detailed in the annex.

Annex:
Selected tasks I would work on if I had one month to work on the project

1. **Longer training:**
   Instead of limiting myself to specific training durations, I would experiment with different training regimes. For instance, extended training (i.e. more training epochs) should be evaluated.
2. **More data:**
   For this task, I have limited myself to using only 10% of the original dataset. As I expect the usage of more training data to significantly improve the performance of the model, I would work with the full dataset or maybe even introduce additional custom data augmentation to potentially improve robustness and overall performance. This also depends on the model architecture used as models like YOLO, for example, inherently execute data augmentation steps.
3. **Model architecture & improvements:**
   Instead of solely relying on YOLO, I would conduct further research into current SOTA models for (real-time) object detection. Architectures like EfficientDet, Faster R-CNN, or RetinaNet could be explored.
   Additionally, it would be important to execute a proper hyperparameter optimization (HPO). While I implemented this code roughly in the notebook, I refrained from executing a full HPO due to its computational requirements. Here, I would focus on experimenting with different hyperparameters such as batch_size, learning_rate, and others that typically have a major impact on model performance.
4. **Deployment:**
   Instead of solely developing and testing the model in a research environment, I would consider implementing an end-to-end deployment pipeline and deploying the model for actual firefighter usage. This would not only provide the real-world benefit of helping the French firefighters but also reveal the strengths and weaknesses of the model in a real-world environment. This might prompt further engineering tasks and challenges such as real-time inference optimizations, working with edge devices in forests, latency, and limited bandwidth in remote forests.