Computer Vision - Fire detection

Hi! PARIS x Pyronear

François Soulier

Contents

1 Approaches **1**

1.1 You Only Look Once (YOLO) 1

1.1.1 YOLOv5 1

1.1.2 Data selection 1

1.1.3 Results on fire detection 1

1.2 Multi-Head detector 2

1.2.1 Description 2

1.2.2 Data preprocessing and augmentation 2

1.2.3 Performance (ResNet-50 & LSTM) 2

1.2.4 Results on fire detection 2

2 Improvements **3**

1 Approaches

1.1 You Only Look Once (YOLO)

1.1.1 YOLOv5

The YOLOv5 model is a cutting-edge object detection model known for its high accuracy in identifying objects in images. It is build upon the YOLO (You Only Look Once) architecture, a single-stage detection model designed for real-time object detection. It is a state-of-the-art model that is widely used in the computer vision community, and it was pre-trained on the COCO dataset.

Consequently, this model is well-suited for the detection of forest fires in landscape images. Indeed, in a real-world scenario, it is crucial to be able to detect fires as quickly as possible, so that it may not spread and cause massive damage. In addition, the YOLOv5 model offers high accuracy in locating objects bounds in images, which is essential to prevent false alarms (false positives predictions) and to enforce the early detection of fires.

1.1.2 Data selection

In order to run the YOLOv5 model training loop under a reasonable amount of time, and using of one T4 NVIDIA GPU, the training dataset was reduced to a subset containing only 10% of the original images.

1.1.3 Results on fire detection

The YOLOv5 model was trained on a fragment of the fire detection dataset.

Class	Images	Instances	P	R	mAP50	mAP50-95
all	152	153	0.301	0.245	0.219	0.0941

Table 1: Performance metrics of the model

- As shown in the **Table 1**, the trained model did not perform well on the fire detection task. The test was performed on 152 images containing 153 instances of fire.
- The model achieved a precision of 0.301, a recall of 0.245, a mean Average Precision (mAP) of 0.219 and a mAP of 0.0941 for the interval 50-95.

1.2 Multi-Head detector

1.2.1 Description

In this approach, the designed model is much lighter than the YOLO architecture, and is based on a ResNet-50 backbone with a multi-head detector. The model is composed of a ResNet-50 backbone, following by a regression head (based on LSTM layers) to predict object bounding boxes and a classification head (based on linear layers) to predict the presence of fire in every possible bounding box. As a result, for each sample, the label is a list of bounding boxes and an associated binary value indicating the presence of fire in each bounding box.

One main advantage of this model is that it can be trained on and smaller computational resources, and can be fine-tuned on a larger dataset to improve the performance. However, it can only support a fixed and limited number of bounding boxes per image, which can be a limitation in some cases.

1.2.2 Data preprocessing and augmentation

The images were processed according to the following steps:

- Reduce noise by applying smoothing and blurring filters (Gaussian, Median, Bilateral).
- Resizing to 224x224 pixels.
- Normalize the pixel values to the range [0, 1].
- Apply random color modifications (brightness, contrast, saturation).

1.2.3 Performance (ResNet-50 & LSTM)

- **ResNet-50 backbone** - The ResNet-50 model, pretrained on the ImageNet dataset, is a state-of-the-art convoluational encoder that is widely efficient for visual feature extraction. As it is also a lightweight model, thus suitable for fast and low-resource training, it can be used as a backbone in a transfer learning approach.
- **LSTM regression head** - The LSTM (Long Short-Term Memory) layers are used to predict the bounding boxes of the objects in the image. The LSTM layers are well-suited for this task, as they can capture the "semantic" information of the feature maps extacted by the encoder, and predict the bounding boxes with a global visual context.

1.2.4 Results on fire detection

The results are discussed directly in the notebook *fire_detection.ipynb*, with visualizations of the model predictions.

2 Improvements

In order to further improve the performance of the models, and to enhance the quality and productivity, the following improvements can be considered:

- **Hyperparameter tuning** - As mean to optimize the chosen hyperparameters, a hyperparameter search could be performed, utilizing techniques such as grid search.
- **Monitoring and logging** - Implement a monitoring and logging system to track the model performance during training and even once deployed in production. To that end, the MLFlow library can be used to log all the metrics, parameters, and artifacts of the model(s).
- **Experiment tracking** - Use a tool such as Weights & Biases to track the experiments and visualize the results in a more interactive way.
- **Model ensembling** - Combine the predictions of multiple models to improve the overall performance and prediction reliability. This can be done by averaging the predictions of the models for instance.
- **Large scale training** - Train the large YOLOv5 model on the full dataset, thus distributing over multiple GPUs or using cloud-based clusters such as AWS, GCP, or Azure.
- **Explainability** - Visualize the model predictions and the feature maps to understand the model's decision-making process. A tool such as Grad-CAM can be used for visulisation purposes.
- **Video processing** - Extend the algorithm to video processing, with and object tracking algorithm (using Kalmann filters for instance) to detect the fire in a sequence of frames.