Proposal: A Tutorial for Fire and Smoke Detection Using Deep Learning on Resource Constrained Devices

Aryan Naim
University of New Mexico
Electrical and Computer Engineering
Albuquerque, New Mexico, USA
anaim@unm.edu

***Abstract*— In recent years, early fire detection has become vital due to large-scale fires causing extensive financial damage and loss of life in such places as the United States, Canada, and Europe. This tutorial introduces how to train deep neural networks for detecting fire and smoke, leveraging the inference of high-speed YOLO architecture. YOLO can be optimized for deployment on resource-limited edge hardware like the Raspberry Pi. The tutorial covers the entire process, including data preprocessing, model training, hyperparameter tuning, evaluating model metrics, conducting model inference in a cloud environment, and final deployment on a RasperryPi W Zero.**

Keywords*— *Object Detection, Machine Learning, Resource Constrained Devices, MLOps (Machine Learning Operations)

I. Motivation

The motivation is to develop a highly accurate, cost-effective, COTS-based, early warning network for fire and smoke detection inspired by recent developments and deep learning-based algorithms that are suited for embedded resource constraint devices.

II. Unique Contribution

During the implementation of the fire detection pipeline, we will adapt already developed code based on the YOLO family of object detection algorithms, however, we will be improving the code quality for the production environment by considering training and optimization of the neural network for resource-constrained devices and model tracking using frameworks such as MLFlow [1][2][3][7]. Furthermore, previous authors have deployed YOLOv4 for fire detection on regular Raspberry Pi's however we will be taking the deployment one step further by deploying to the smaller and more resource-constrained Raspberry Pi W Zero. We will be providing a GitHub repo with all the source code, step-by-step setup, and references.

III. Expected Contribution Against Competitive Methods

During the training of the fire and smoke detection model based on the YOLOv4 architecture we will use the following performance metrics mAP, F1-score, and average IoU as listed in Table 1.

Performance metrics on the test set

Network	mAP@0.50 (%)	F_1-score	avg IoU (%)
Tiny YOLOv4	61.4740 ± 0.6488	0.6300 ± 0.0071	51.8880 ± 0.5865
YOLOv4	73.9840 ± 0.4029	0.7240 ± 0.0055	60.5080 ± 0.3136

Table 1. Model performance metrics on the test set [1].

We will be attempting to match or exceed these metrics primarily while attempting to decrease detection time.

IV. Draft Flowchart or Pseudo-code for a YOLO Training Pipeline

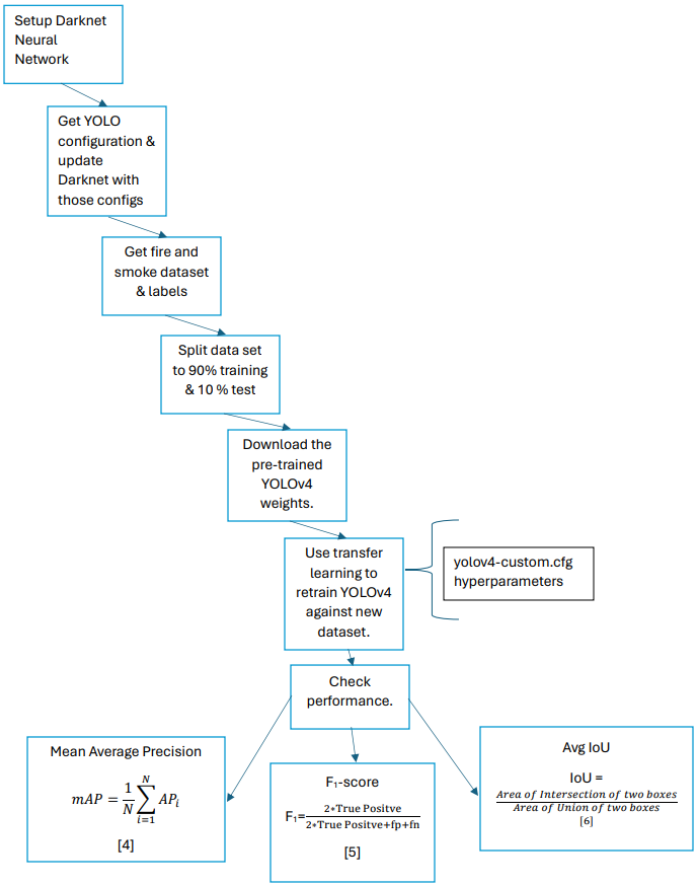

Figure 1. High Level YOLO training pipeline

V. Code and Implementation

The primary language that you will be using is Python. The neural network training frameworks will be TensorFlow, Keras, and Darknet. Finally, all the software will be implemented on the cloud using Google Colab's online environment. The source code will be adapted from an online tutorial on training YOLOv4 using Python [8]. The architecture and tuning methods were influenced by several previous projects that attempted fire detection using YOLO and other deep-learning object detection techniques [1][2][3].

VI. Dataset

Category	Description	Number Of Images
Fire	Images that contain only fire	1,164
Smoke	Images that contain only smoke	5,867
Smoke & fire	Images that contain both smoke & fire.	4,658
None	Images containing neither fire nor smoke	9,838
Total		21,527

Table 2. Dataset number of images per class [1].

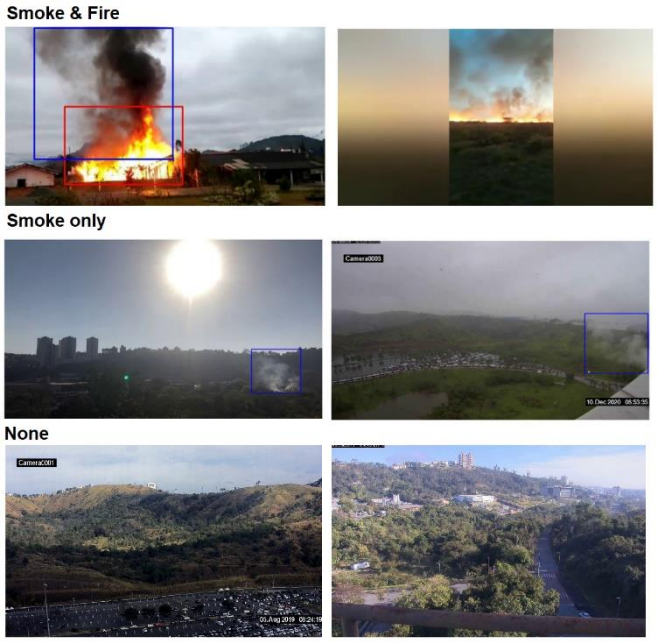

Figure 2. Sample images for different classes [4].

The primary dataset will be images of fire, smoke, and labels from: https://gaiasd.github.io/DFireDataset/. There are a total of 21,527 images. This dataset is used to train and study the models in the paper titled "An automatic fire detection system based on deep convolutional neural networks for low-power, resource-constrained devices." Which will serve as our main guide [1]. Will be attempting to classify 2 different classes which are fire and smoke.
Based on a quick inspection of the images in the dataset one can notice that these images are from a variety of sources such as surveillance cameras, smartphones, television screen cameras, and aerial images. Therefore, there is a range of different angles and image qualities.

VII. Data Training

The data will be trained and validated using K-Fold cross-validation and using metrics mAP, F1-score, and Avg IoU (Intersection over Union). Cross-validation is a statistical method used to protect against overfitting in a predictive model, particularly when the amount of data is limited. In cross-validation, the dataset is partitioned into subsets, and the analysis is carried out repeatedly. We will be using 5-fold cross-validation. The dataset is divided into k=5 equal parts also known as folds, and the model is trained on 'k-1' folds while using the remaining fold as the test set. This process is repeated 5 times, with each fold used exactly once as the test set. The model's performance (mAP, F1-score, and Avg IoU) is then averaged over the 'k' trials to provide an overall effectiveness measure.

VIII. Main Reference Paper

The main reference we will be reproducing will be "An automatic fire detection system based on deep convolutional neural networks for low-power, resource-constrained devices."[1] this paper is published in Neural Computing and Applications Springer with an impact factor of 6.0 as of 2022.

References

[1] P. V. A. B. de Venâncio, A. C. Lisboa, and A. V. Barbosa, "An automatic fire detection system based on deep convolutional neural networks for low-power, resource-constrained devices," Neural Computing and Applications, vol. 34, no. 18, pp. 15349–15368, Jun. 2022, doi: https://doi.org/10.1007/s00521-022-07467-z.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] X. Geng, Y. Su, X. Cao, H. Li, and L. Liu, "YOLOFM: an improved fire and smoke object detection algorithm based on YOLOv5n," Scientific Reports, vol. 14, no. 1, Feb. 2024, doi: https://doi.org/10.1038/s41598-024-55232-0.

[3] Techzizou, "TRAIN A CUSTOM YOLOv4 OBJECT DETECTOR (Using Google Colab)," Analytics Vidhya, Mar. 17, 2024. https://medium.com/analytics-vidhya/train-a-custom-yolov4-object-detector-using-google-colab-61a659d4868#4be1 (accessed Mar. 24, 2024).

[4] https://gaiasd.github.io/DFireDataset/

[5] https://towardsdatascience.com/breaking-down-mean-average-precision-map-ae462f623a52

[6] https://en.wikipedia.org/wiki/F-score

[7] https://medium.com/analytics-vidhya/iou-intersection-over-union-705a39e7acef

[8] https://medium.com/analytics-vidhya/train-a-custom-yolov4-object-detector-using-google-colab-61a659d