Performance Tradeoffs of a CNN Architecture with Focus on Wildfire Detection

By: Noah Cameron, Matthew Eng, Rohith Malangi, and Veda Yakkali

Objective:

- Aim is to analyze the performance tradeoffs in a Convolutional Neural Network (CNN) architecture focusing on wildfire detection
- Understand how different parameters (e.g., model, CPU/GPU usage and image size) affect detection accuracy, inference speed and especially on resource-constrained devices

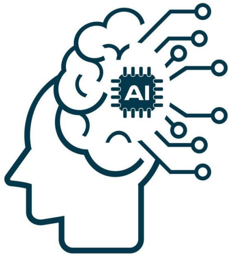
Methods of Analysis:

- Utilize the YOLOv7 (You Only Look Once) object detection framework
- Utilize Colab to do training and testing
- Train models to be able to detect fire and smoke
- The architecture and pre-trained weights remain consistent with YOLOv7
- Takes testing sets of images as Input and Outputs detections of Fire and Smoke as bounding boxes
- Test with adjustable parameters like models, CPU/GPU, and image size to analyze their effect
- Record results in a spreadsheet to look for trends and anomalies

Datasets:

- Utilized 2 Datasets
  - https://github.com/gaiasd/DFireDataset?tab=readme-ov-file
    - This was utilized for training our models
      - Consists of 17,000 training images and 4,300 testing images with labels
      - Has Fire, Smoke and Neutral images
  - https://github.com/DeepQuestAI/Fire-Smoke-Dataset?tab=readme-ov-file#reqirements
    - This was utilized for testing out models
      - Consists of 100 images for Fire, Smoke and Neutral

Training Process:



- Consistent Batch Size (16)
- Different Epochs (50 and 100)
- Utilized 2 different pre existing models (tiny-yolov7 and yolov7)
  - We Created 4 Models:
    - Tiny-Yolov7 with 50 epoch (tiny50)
    - Yolov7 with 50 epoch (reg50)
    - Tiny-Yolov7 with 100 epoch (tiny100)
    - Yolov7 with 100 epoch (reg100)

Testing Process:

- We Have:
  - 3 Testing Sets (Fire, Smoke, Neutral) each 100 images
  - 4 Models (reg50, tiny50, reg100, tiny100)
  - 4 Image Sizes (1024, 640, 480, 320)
  - 2 Runtimes (CPU, GPU)
    - **In Total 96 Experiments**
      - Tracked:
        - Time to Complete
        - Average Inference Time
        - Accuracy

Results (Fire Dataset):

Completion	1042	640	480	320
Reg50 GPU	9.713	5.66	4.727	4.023
Reg50 CPU	273.823	99.053	57.358	27.325
Tiny50 GPU	4.82	4.428	4.157	3.197
Tiny50 CPU	46.544	19.181	11.384	8.797
Reg100 GPU	9.431	5.378	5.666	4.011
Reg100 CPU	273.904	170.079	98.311	48.909
Tiny100 GPU	3.887	3.641	3.33	3.073
Tiny100 CPU	46.538	18.274	10.811	8.994

Accuracy	1042	640	480	320
Reg50 GPU	85	93	95	97
Reg50 CPU	85	93	95	97
Tiny50 GPU	86	96	97	97
Tiny50 CPU	86	97	97	97
Reg100 GPU	90	94	96	97
Reg100 CPU	90	94	96	97
Tiny100 GPU	85	95	97	95
Tiny100 CPU	85	95	97	95

Avg Inference	1042	640	480	320
Reg50 GPU	20.62	10.23	7.188	6.739
Reg50 CPU	2731.11	985.93	569.68	269.86
Tiny50 GPU	6.182	5.781	5.684	4.578
Tiny50 CPU	458.34	187.11	109.88	83.553
Reg100 GPU	20.616	9.582	8.401	6.915
Reg100 CPU	2732.005	1695.44	978.47	484.87
Tiny100 GPU	4.9109	4.4601	4.215	4.353
Tiny100 CPU	458.38	178.12	104.084	57.652


fire 0.46
fire 0.25
fire 0.67
fire 0.39


fire 0.62


fire 0.31
fire 0.67
fire 0.76
fire 0.71
fire 0.45


fire 0.67
fire 0.28

Results (Smoke Dataset):

Completion	1042	640	480	320
Reg50 GPU	8.994	5.383	4.251	4.476
Reg50 CPU	278.04	102.28	58	27.99
Tiny50 GPU	3.733	3.281	2.93	3.545
Tiny50 CPU	45.56	19.19	11.33	4.72
Reg100 GPU	9.827	5.738	4.399	4.519
Reg100 CPU	277.87	102.68	57.72	26.89
Tiny100 GPU	4.266	3.282	3.845	2.749
Tiny100 CPU	43.81	17.55	10.95	49.97

Accuracy	1042	640	480	320
Reg50 GPU	49	78	83	83
Reg50 CPU	49	78	83	83
Tiny50 GPU	55	69	85	83
Tiny50 CPU	55	69	85	83
Reg100 GPU	54	79	83	85
Reg100 CPU	54	79	83	85
Tiny100 GPU	51	67	76	79
Tiny100 CPU	51	67	76	79

Avg Inference	1042	640	480	320
Reg50 GPU	19.72	9.18	7.02	8.05
Reg50 CPU	2772.71	1018.55	576.44	276.83
Tiny50 GPU	4.78	4.18	4.06	4.83
Tiny50 CPU	449.13	187.61	109.66	44.51
Reg100 GPU	20.15	9.37	7.54	7.98
Reg100 CPU	2772.16	1022.46	573.6	265.78
Tiny100 GPU	5.48	4.52	5.14	4.22
Tiny100 CPU	431.7	171.45	106.05	5.3


smoke 0.27

Results (Neutral Dataset):

Completion	1042	640	480	320
Reg50 GPU	4.419	2.546	2.7	2.071
Reg50 CPU	513.945	190.181	116.894	53.469
Tiny50 GPU	2.331	2.043	1.823	2.411
Tiny50 CPU	81.964	32.659	19.256	10.157
Reg100 GPU	3.846	3.085	2.099	2.147
Reg100 CPU	517.153	193.576	116.199	54.606
Tiny100 GPU	2.462	1.964	1.89	1.936
Tiny100 CPU	82.291	31.62	21.441	9.703

Accuracy	1042	640	480	320
Reg50 GPU	86	92	90	87
Reg50 CPU	86	92	90	87
Tiny50 GPU	78	88	83	91
Tiny50 CPU	78	87	83	91
Reg100 GPU	93	94	92	91
Reg100 CPU	93	94	92	91
Tiny100 GPU	80	84	86	85
Tiny100 CPU	80	84	86	86

Avg Inference	1042	640	480	320
Reg50 GPU	21.058	9.898	8.505	7.064
Reg50 CPU	5124.53	1891.14	1159.06	525.59
Tiny50 GPU	6.846	4.974	4.394	5.201
Tiny50 CPU	805.049	316.003	182.84	92.734
Reg100 GPU	21.104	10.044	7.637	7.093
Reg100 CPU	5157.227	1925.055	1152.577	537.514
Tiny100 GPU	7.333	4.974	4.422	4.321
Tiny100 CPU	807.96	305.126	204.446	88.406


fire 0.55
fire 0.26

fire 0.27

Analysis:

- **Depending on the Case:**
  - **If live monitoring/speed is critical:**
    - Model: Tiny50
    - Image Size: 320
    - Hardware: GPU
    - Justification: Offers the fastest inference time while maintaining high accuracy
  - **If accuracy is the top priority:**
    - Model: Reg100
    - Image Size: 320
    - Hardware: GPU
    - Justification: Achieves the highest accuracy with reasonable inference time
  - **If working with energy constraints or limited hardware:**
    - Model: Tiny100
    - Image Size: 320
    - Hardware: CPU
    - Justification: Maintains competitive accuracy while keeping inference time manageable

What Have We Learned?

- Train and Test Data
- Adjust parameters to improve accuracy and efficiency
- Impact of Hardware on Inference
  - GPUs significantly outperform CPUs in inference speed and efficiency
- Tradeoffs Between Speed and Accuracy
  - Tiny models provide faster inference but may reduce accuracy and feature detection capabilities
  - Regular models offer higher accuracy but require more computational resources
- Optimization of Parameters
  - Depending on the situation, a different set of model, runtime, and image size may be appropriate
- Future Insight:
  - Project will provide a practical framework for optimizing CNN architectures on resource-constrained devices wildfire detection

ANY QUESTIONS?