Performance Tradeoffs of a CNN Architecture with Focus on Wildfire Detection

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

The project aimed to explore performance tradeoffs of a Convolution Neural Network (CNN) to detect wildfires. This topic is relevant with the increasing number of wildfires going on throughout the world. From the constant fires of California, to the bushfires of Australia, to the burning of the Amazon, there is no shortage of these disasters. The sight of flames or smoke in the woods is an indicator that a wildfire is present or starting to form. The model could be installed in forest-facing cameras, as well as other locations where it would be most useful.

The project focuses on examining performance tradeoffs in a Convolutional Neural Network (CNN) architecture for object detection, specifically tailored for analyzing images of wildfires. We utilized YOLOv7 (You Only Look Once), which is a real-time object detection algorithm built on a convolutional neural network (CNN) that simultaneously predicts bounding boxes and class probabilities for objects in an image, offering high speed and accuracy.

To implement this, we trained 4 models that were trained on a fire and smoke dataset with labels that are taken from this dataset. All of the 4 models that were created from the training, utilized the 2 existing models, YOLOv7 and Tiny YOLOv7 as the basis for the training. YOLOv7 uses a deeper and more complex neural network with more layers and parameters, enabling higher accuracy and feature extraction, while Tiny YOLOv7 simplifies the architecture with fewer layers and parameters, prioritizing speed and efficiency for resource-constrained environments. Each model is a 50 and 100 epoch tiny model and 50 and 100 epoch regular model, where the batch size and all other parameters were kept constant across the training.

After the training stage, the focus shifts to evaluating and optimizing the inference process. This involves analyzing the tradeoffs between accuracy, computational speed, and efficiency. By understanding these tradeoffs, performance can be balanced with resource utilization, ensuring the model operates effectively in real-world scenarios where quick and reliable predictions are critical.

The parameters modified are image size, CPU vs GPU, and the model used. For each set of different parameters, the inference program is run on the dataset which has 3 different sets of images: fire, smoke, and neutral (no fire or smoke), where each set contains 100 images.. Completion time, accuracy, and average inference time was measured for each set of parameters and class. Given the presence of noise in many of the inputs and sometimes differing utilization of the colab runtimes, we will execute the program multiple times and take an average to obtain more accurate and reliable results.

The detailed final results have been recorded and can be accessed for review here. The column headers referred to the image size, where the completion table is recorded in seconds, and the average inference is recorded in milliseconds. The fire set reported the highest accuracies out of all 3 groups, where the smoke set saw much lower accuracies, and the neutral category had better accuracy (an image correctly classified if no smoke or fire was detected).

After our presentation, we received feedback that the GPU completion times did not always trend downwards with decreasing image sizes, as should be the case. We therefore reran those experiments that produced increasing completion times to ensure that the GPU was properly being utilized. After the reruns were completed, we were able to obtain more consistent results, but we believe the inconsistency in the readings could have been due to several factors.

Initially, the GPU may not have been fully utilized due to Colab environment limitations, background processes, or system throttling, leading to inconsistent results. It's also possible that background processes or system throttling affected performance during the initial tests. After rerunning the experiments, the GPU was likely allocated more resources, allowing for proper utilization and the expected decrease in completion times with smaller image sizes.

Several interesting trends were identified from the collected data. Across all experiments, the rate of decrease in CPU completion times with respect to image size was significantly greater than the rate of decrease of GPU completion times with respect to image size. For example, the CPU rate of decrease was around 50% for each smaller image size while the gpu rate was usually 10% or less. An explanation for this could be the increased parallel processing capabilities of the GPU versus CPU. Since the GPU is more parallelizable, it could be better at handling larger workloads (image size) and therefore was not as affected by the decrease in image size.

Another trend is that accuracy tends to increase with lower image resolution. This is slightly counterintuitive as one would expect a higher resolution image (with more detail) to offer better inference. One reason lower image resolution images were more accurate could be because there tends to be more noise in higher resolution images, which could make feature detection more difficult for the CNN model, thus reducing accuracy.

Lastly, there was a stark difference in accuracy between the smoke testing set and the fire & neutral testing sets. As mentioned earlier, due to the nature of smoke in an image (it spreads out more and is a less well defined object than fire), this may make it harder for a CNN to detect its features and place a bounding box around it. As smoke is generally a grey color, it is more likely to blend in with surrounding objects or elements in the image, whereas fire stands out more as it is usually a red or orange color. This makes smoke harder to detect than fire in most cases. But compared to the fire set, which had less false negatives compared to the smoke set, it still had some false negatives that mainly depended on the image size as stated before. The neutral set had a couple false positives, where some pictures show a bounding box of either fire or smoke which in reality it shouldn't show any bounding box. The reason that these bounding boxes showed up is due to the fact that these pictures had similar color palettes to fire and smoke which caused the model to falsely identify it as fire or smoke. But overall, the models had good accuracy ratings on the fire, smoke and neutral dataset.

From the project overall, we learned a variety of useful skills in the context of deep learning. We learned about the importance of updating and varying parameters to observe the results that the machine produced. This led to us learning about the impact of hardware on the performance of a model as well. In addition, we learned about the tradeoffs between speed versus accuracy, and how Tiny models provide faster inference but may reduce accuracy and feature detection capabilities. Meanwhile, Regular models offer higher accuracy but require more computational resources. As for future insight, more time could be spent on the training aspect of the project to better optimize each model based on the epoch count and also the batch size to increase precision and recall numbers. But the main takeaway from our project is that it will provide a practical framework for optimizing CNN architectures on resource-constrained devices for wildfire detection.