PRACTICAL WORK : FIRE DETECTION

IA323

Lefebvre Alexandre, Marguet Emeric, Cherel Vincent, Gossec Yaël

1 DATASET DESCRIPTION

This project utilizes a dataset derived from Canada's wildfire data. The dataset consists of satellite images, each measuring 350x350 pixels, categorized into two classes:

- **Wildfire**: 22,710 images
- **No Wildfire**: 20,140 images

The dataset is split into three subsets:

- **Training set**: ∼70%
- **Test set**: ∼15%
- **Validation set**: ∼15%

(a) Example Image from the Wildfire Class.

(b) Example Image from the No Wildfire Class.

Figure 1: Comparison of Wildfire and No Wildfire Images.

2 DATA PROCESSING

As instructed, we divided the validation set (∼6000 images) into two equal parts: a new training set and a new validation set. This allowed us to refine the model's generalization ability without relying on the original training dataset.

In order to leverage the pretrained resnet, we resized the images from 350 x 350 to the 224 x 224 format, on which resnet50 is pretrained.

3 MODEL SELECTION AND ARCHITECTURE

Considering computational constraints, we opted for a pre-trained deep learning model to reduce training time. The chosen architecture is **ResNet-50**, a widely-used convolutional neural network (CNN) known for its balance between depth and efficiency. ResNet-50 is equipped with residual connections, which help mitigate the vanishing gradient problem in deep networks, allowing for a deep architecture (174 layers).

We initialized the model with pre-trained weights provided in the `torchvision` library. The default pre-trained weights were chosen for convenience and reproducibility. The final fully connected layer of ResNet-50 was replaced with a single-node output layer to suit our binary classification task. The architecture is depicted in Figure 2.

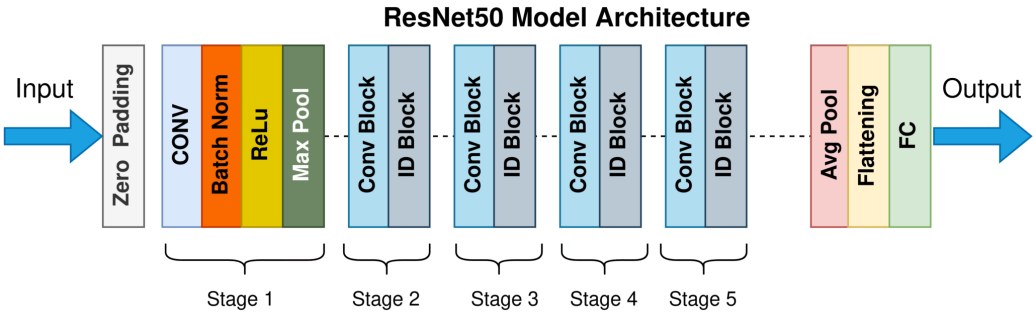

Figure 2: ResNet-50 Architecture.

4 TRAINING AND HYPERPARAMETERS

We trained the model for 10 epochs, ith Specifically, we used:

- **Loss Function**: Binary Cross-Entropy (BCE) Loss
- **Optimizer**: Adam optimizer with default hyperparameters (learning rate = 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$)
- **Batch Size**: Default batch size from PyTorch's DataLoader

Using the default optimizer settings meant that the training was stable and required minimal hyperparameter tuning. However, adjusting the learning rate or batch size could potentially improve convergence.

5 RESULTS AND ANALYSIS

After only 10 epochs, the model achieved **96% accuracy** on the new validation set. To analyze model failures, we added a section in the notebook to visualize incorrect predictions. It is rather tough to find a visual pattern. It seems that it's hard to predict wildfire in urban area, fields and areas with a body of water. Further investigation could be made on this topic, perhaps one could find cornercases on which the model performs well, and perform data augmentation on these cases to further finetune the model.