André Ramolivaz
NetID: 3933665317

HMW 6: CNN Image Classification on Crack Dataset

This report presents an analysis of a Convolutional Neural Network (CNN) applied to classify images of concrete as either containing cracks or not. The dataset consists of 4000 images (2000 positive, 2000 negative). The CNN model was developed using the **tensorflow.keras** library, trained and tested on an HPC (High-Performance Computing) cluster.

CNN Architecture

The implemented CNN architecture includes:

- **Input Layer**: Images were standardized to the size (150,150,3) to provide consistency and reduce computational load without significantly losing image quality.
- **Rescaling Layer**: Pixel values were normalized to the range [0, 1] to ensure numerical stability and faster convergence during training.
- **Convolutional Layers**: Three convolutional layers with increasing filter numbers (32, 64, and 128) were chosen to incrementally capture low-level to high-level image features, respectively. Each layer uses a (3x3) kernel, which effectively captures local features. Batch Normalization and MaxPooling were employed after each convolution to reduce internal covariance shift and spatial dimensionality, thus improving convergence speed and performance.
- **Fully Connected Layers**: A dense layer of 128 neurons with ReLU activation was implemented to synthesize the learned spatial features. A dropout rate of 30% was added to reduce the likelihood of overfitting by randomly disabling neurons during training.
- **Output Layer**: A sigmoid activation function was chosen for binary classification due to its suitability in distinguishing between two classes.

Performance Metrics

Confusion Matrices

The model performance is evaluated via confusion matrices for both training (Fig. 1) and testing (Fig. 2) datasets.

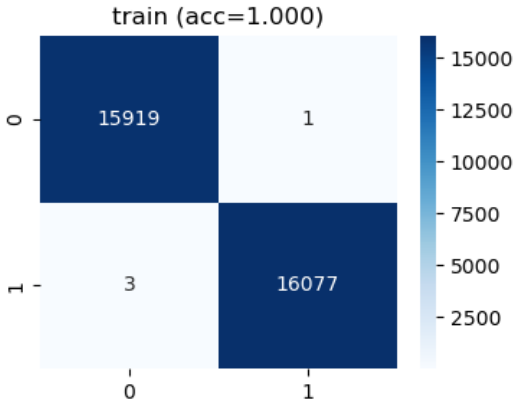

Figure 1: Training Confusion Matrix (Accuracy: 99.99%)

André Ramolivaz
NetID: 3933665317

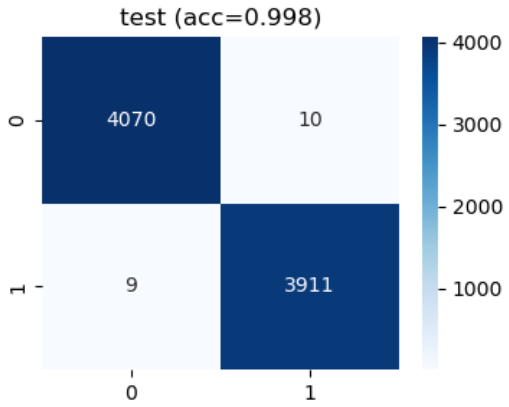

Figure 2: Testing Confusion Matrix (Accuracy: 99.76%)

From the matrices:

- Training set accuracy is 99.99%, with only 4 misclassifications out of 32,000 samples.
- Testing set accuracy is 99.76%, with only 19 misclassifications out of 8,000 samples.

Loss Curves Analysis

The loss curves for training and validation sets (Fig. 3) provide insights into model training dynamics:

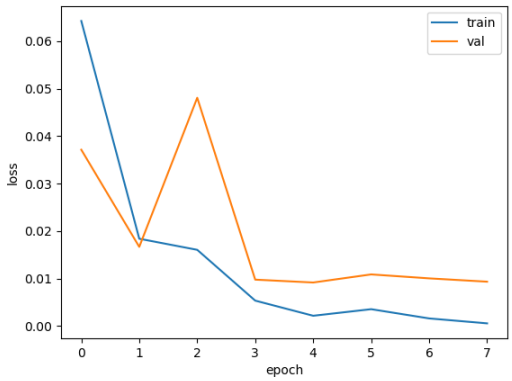

Figure 3: Training and Validation Loss Curves

- Both curves show a rapid initial decrease, indicating effective learning.
- After 3 epochs, the curves stabilize and closely converge, reflecting minimal risk of overfitting or underfitting.
- Validation loss does not significantly diverge from training loss, supporting robust model performance.

Implementation Decisions

Python Script

The Python training script follows a modular and reproducible design, using the TensorFlow high-level API for dataset loading, model construction, and training. Key choices include:

- Use of image_dataset_from_directory to automatically label images and create validation splits, which streamlines data preparation.
- Use of callbacks such as `EarlyStopping`, `ReduceLROnPlateau`, and `ModelCheckpoint` to improve training efficiency, prevent overfitting, and retain the best-performing model.

André Ramolivaz
NetID: 3933665317

- Saving key output artifacts (confusion matrices, loss curves, metrics) for reproducibility and later analysis.

SLURM Job Script

The SLURM submission script is tailored for resource efficiency on the HPC cluster. Main decisions include:

- Requesting 4 CPUs and 16GB memory, which ensures adequate capacity for loading image batches and training on CPU.
- Setting a timeout of 4 hours to accommodate model training with sufficient headroom for longer runs.
- Dynamically assigning output folders using `$SLURM_JOB_ID` to avoid overwriting results and enable job-specific logging.

Folder Structure on HPC

The directory structure on the CARC-HPC cluster is organized as follows:

```
cnn_project/
|-- src/
|   `-- train_cnn.py   (script containing CNN model and training procedure)
|-- out/
|   `-- <SLURM_JOB_ID>/ (output directory containing model outputs)
|       |-- confusion_train.png
|       |-- confusion_test.png
|       |-- history_loss.png
|       |-- metrics.txt
|       `-- best.h5      (saved best-performing model)
|-- datasets/
|   `-- concrete/
|       |-- positive/  (cracked concrete images)
|       `-- negative/  (non-cracked concrete images)
|-- cnn_crack_<SLURM_JOB_ID>.out (standard output log)
`-- cnn_crack_<SLURM_JOB_ID>.err (standard error log)
```