Arvind Vinod
Stubbs
COE 379L

Project 3 Report

The goal of this project was to classify satellite images from Texas post-Hurricane Harvey as containing buildings with damage or no damage using neural networks. The dataset included 14,170 damaged and 7,152 non-damaged images. The project involved data preprocessing, designing and training multiple neural network architectures (Dense ANN, LeNet-5, and a custom CNN described in a research paper), evaluating model performance, and deploying the best model as an inference server via Docker. This report details the methodologies, challenges, and outcomes of each stage, providing insights into the decision-making process and final results.

The dataset was analyzed and preprocessed to ensure compatibility with neural network training. The raw dataset exhibited a significant class imbalance, with the damage class containing nearly double the images of the no_damage class (14,170 vs. 7,152 images). This imbalance posed a risk of model bias toward the majority class. To address this issue, class weights were applied during training, assigning higher penalties to misclassifications in the minority class which meant that the model did not disproportionately favor predicting the damage class. The dataset was split into 70% training, 15% validation, and 15% testing subsets as shown in class. Random shuffling was applied to eliminate biases, ensuring that geographic patterns did not skew the model's learning. Preprocessing steps included rescaling pixel values to the range [0, 1] using Rescaling(1./255) to normalize inputs for neural network compatibility. To enhance generalization and reduce overfitting, data augmentation techniques were applied during training. Transformations included horizontal flips to simulate mirrored perspectives, random rotations of 10 degrees, zoom-ins and outs of 10%, and translations of 10%. These steps artificially expanded the dataset's diversity, exposing the model to varied orientations and scales of damage patterns. I note that this was only applied to training data, to preserve the test data.

Three neural network architectures were implemented and rigorously evaluated to identify the optimal solution for this classification task. The first model was a Dense Artificial Neural Network (ANN) designed to establish a performance baseline. The architecture began with an input layer that flattened the images into a dimensional vector. This was followed by two hidden

Dense layers with 256 and 128 units, both using ReLU activation to introduce non-linearity. Dropout layers were added after each Dense layer to mitigate overfitting. The output layer used a sigmoid activation function to produce binary predictions. The model achieved a test accuracy of 33.56% and an AUC of 0.5, equivalent to random guessing. This poor performance underscored the unsuitability of fully connected layers for image data.

The second model implemented the LeNet-5 CNN architecture, a convolutional neural network designed for image recognition. The architecture included two Conv2D layers with 6 and 16 filters using 5x5 kernels, followed by MaxPooling layers to downsample feature maps. The output was flattened and passed through two Dense layers. This model achieved a test accuracy of 89.47% and an AUC of 0.9628, demonstrating the superiority of CNNs over fully connected networks for image tasks. The convolutional layers effectively extracted spatial features like edges and textures, while MaxPooling reduced dimensionality and computational complexity.

The third model, a Custom CNN inspired by a research paper, was designed to address the limitations of LeNet-5. The architecture included four Conv2D layers with 32, 64, 128, and 128 filters using 3x3 kernels, each followed by MaxPooling to progressively abstract features. A 50% dropout layer was added after flattening to prevent overfitting, followed by a Dense layer with 512 ReLU units and a sigmoid output layer. This deeper architecture achieved best performance: 96.69% test accuracy, 0.9953 AUC, 93.37% precision, and 97.02% recall. The model's success stemmed from its ability to learn hierarchical features. Early layers detected edges and corners, while deeper layers identified complex structures like collapsed walls or intact roofs. Dropout regularization ensured robustness against overfitting, even with the imbalanced dataset.

Models were evaluated using accuracy, AUC-ROC, precision, and recall. The AUC-ROC score was prioritized due to its robustness to class imbalance, as it measures the model's ability to distinguish between classes across all classification thresholds. The Dense ANN failed pretty badly with an AUC of 0.5, highlighting the inadequacy of fully connected architectures for image classification. The LeNet-5 CNN performed well with an AUC of 0.9628 but was constrained by its shallow design. The Custom CNN achieved near-perfect performance with an AUC of 0.9953. The custom CNN was validated using the grader script, which tested the deployed model's API endpoints against new images. The model achieved 100% accuracy on the grader's test suite, confirming its reliability in real-world scenarios. Analysis of the confusion

matrix revealed that the model misclassified 2.6% of no_damage images as damaged, likely due to shadows or vegetation obscuring roofs. Only 1.3% of damage images were misclassified, a critical achievement for disaster response where missing damage could delay recovery efforts. The best model (Custom CNN) was serialized as final_best_model.h5 and deployed using Docker for scalability. A Docker image was built and pushed to Docker Hub as a public repository under the tag arvindvinod/hurricane-damage-classifier:latest. This allows anyone to pull the image using docker pull arvindvinod/hurricane-damage-classifier:latest. The deployment included two API endpoints: GET /summary for model metadata and POST /inference for image classification. Users can start the server using docker-compose up, which relies on a docker-compose.yml file configuring port mapping (5000:5000).

In short, the Custom CNN emerged as the optimal model, achieving 96.69% accuracy and near-perfect AUC (0.9953). Its balance of precision (93.37%) and recall (97.02%) minimizes false negatives, a critical requirement for disaster response applications. The model's deployment via Docker ensures scalability, enabling integration with real-time satellite imagery pipelines, allowing for disaster response to be much quicker.