Hack4dev
hack4dev.org

Regional Hackathon Pipeline Submission Template

Team Number:2
Regional Hackathon Location: Brazilian Center for Physics Research
Date of Submission: 26/03/2025
Team Members: Giovanna Bertoletti, Luís Guedes, Omar Mesquita

---

1. Pipeline Description and Evaluation (200-350 words)

A. Preprocessing & ML Methods

The preprocessing pipeline involved the following actions:

1. Combination of the training and validation sets
2. Data augmentation through random oversampling to ensure label balance
3. Distribution of the combined oversampled set: 80% of the data for training and 20% for validation
4. Conversion of each image in the dataset into a histogram representing its pixel intensity distribution in each channel (R, G, B)

Initially, the model was defined as a fully connected neural network (FCNN):

- One dense layer with 128 neurons and ReLU activation
- One dense layer with 64 neurons and ReLU activation
- One dense layer with 5 neurons (number of labels in data) with Softmax activation

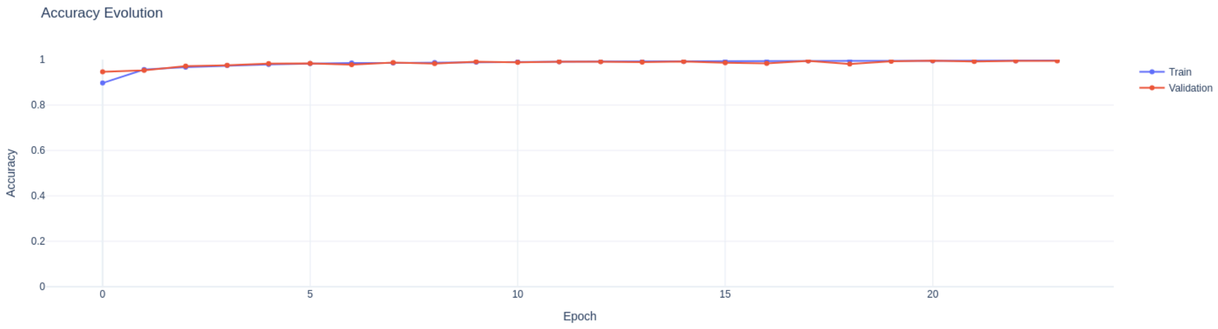
Afterwards, an improved model was created through Keras' tuner RandomSearch in search of optimal model hyperparameters, and immediately retrained for 50 epochs to ensure the model converged. At the end of the pipeline, the final architecture was defined as:

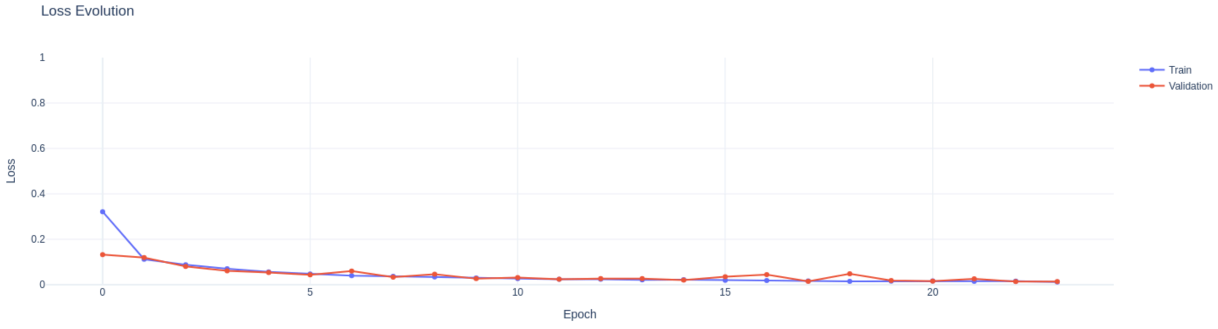
- One dense layer with 96 neurons and ReLU activation, followed by Dropout for regularization.
- One dense layer with 64 neurons and tanh activation, followed by Dropout.
- One dense layer with 5 neurons (matching class labels) and Softmax activation.

B. Results and Discussion

The histogram based FCNN achieved an accuracy of 99.48% on the validation set, underperforming CubeSat CNN's accuracy by less than 1% [1]. The use of histograms (flattened, lower dimensional data) and the adoption of a simpler, non convolutional architecture reduced training time by approximately 98%.

The training ran for 15.4 seconds on a workstation PC equipped with an AMD Ryzen 5 over 30 epochs. The resulting keras model size was 135.72 KB, representing a 65% reduction in model size compared to CubeSat CNN. The graphs below demonstrate the accuracy and loss function (Categorical Cross Entropy) evolution over epochs. Similarly to the article's graphs, an initial steep increase in accuracy is accompanied by a plummet in the loss function. Over epochs, both metrics tend to stabilize, i.e., the model converges.

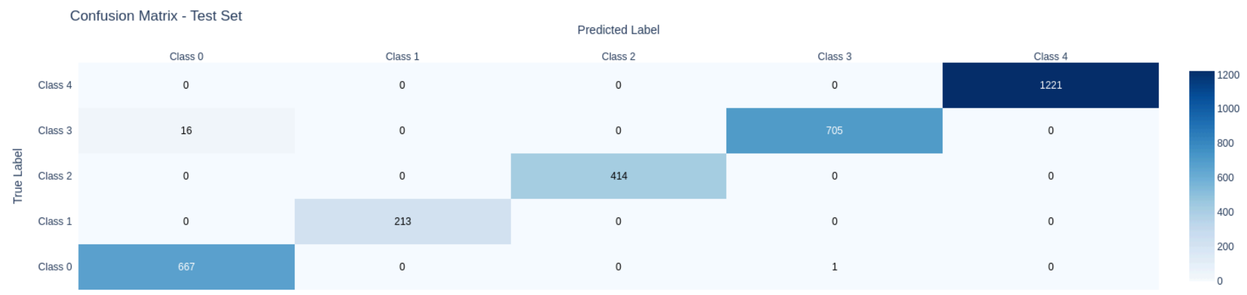




In the classification report below, an important aspect is shown: the accuracy for Class 4 (Priority) remains at 100% which is critical to the CubeSat's main goal of enqueuing relevant astronomical data first to downlink. In this context, the misclassifications between classes 0 (Blurry) and 3 (Noisy) constitute a reasonable tradeoff for drastically reduced computation on ground stations and smaller model size in space.

Labels	**Precision**	**Recall**	**F1-Score**
Blurry	0.9766	0.9985	0.9874
Corrupt	1.0000	1.0000	1.0000
Missing Data	1.0000	1.0000	1.0000
Noisy	0.9986	0.9778	0.9881
Priority	1.0000	1.0000	1.0000
Accuracy			0.9947
Macro Average	0.9950	0.9953	0.9951
Weighted Average	0.9949	0.9947	0.9947

The model's confusion matrix on novel data in the test set is available below:



Confusion Matrix - Test Set

Predicted Label

True Label	Class 0	Class 1	Class 2	Class 3	Class 4
Class 4	0	0	0	0	1221
Class 3	16	0	0	705	0
Class 2	0	0	414	0	0
Class 1	0	213	0	0	0
Class 0	667	0	0	1	0

C. Challenges, Reflections, and Time Constraints

Initially, following advice from Professor Thron's video, we attempted to improve the preprocessing of the model in Notebook 3 - a traditional ML model. Our attempts at reducing the downscaling of the original 512x512 images proved ineffective: downscaling the images to half (256x256) and to a fourth (128x128) did not improve the metrics nor the confusion matrix, actually worsening them in some tests. Given that the preprocessing in that notebook was quite optimal, we pivoted to Notebook 4.

There, a convolutional neural network was presented. CubeSat CNN had perfect and almost perfect (99%) scores across all labels, yet its main issue lay in the extremely lengthy training duration (up to 2 hours on the Ilifu VMs). Our approach then was to use a similar model (a deep learning one) that scored at least nearly as good but was more lightweight.

Deep learning models, despite their complexity, can yield strong results even with simpler data representations. After brainstorming different approaches (RESNET, Visual Transformers, etc), we opted for a deep learning model called a fully connected neural network (FCNN).

Its main strength consists of a simple architecture with 3 fully connected (also called dense) layers which takes a batch of 1D arrays with 30 features as input. This is a drastic decrease from the original input size in the CNN (raw 512x512 image data), but requires careful preprocessing along with the training.

Once the training time was successfully decreased (taking at most 30 seconds on a workstation PC), we attempted to improve the model's hyperparameters through GridSearch, and later through Keras's RandomSearch, achieving a setup which converts unseen, test image data into histograms and predicts the labels in less than 40 seconds. The metrics below summarize this evaluation.

```
Evaluation Time:        38.1054 seconds

Peak Memory Usage:      5002.89 MB

Average CPU Usage:      9.40%

Model Code Size:    135.72 KB

Accuracy:               0.9947

F1 Score:               0.9947
```

Due to its simple metrics and the limited hardware of CubeSat satellites, an interesting reflection was brought up by the team: since models tend to lose their performance over time [2], this novel architecture allows the retraining of the model in space, ensuring it can effectively classify photos taken over an extended period of time. This reduces mission maintenance costs since there will be no need for frequent CubeSat launches, as the model intelligently updates itself with new data.

The tight time constraints of the hackathon were the major challenge to our group: coming up with new ideas, testing them, and presenting a unique solution at the end of the week were very challenging, yet fruitful. During the last hours of the hackathon, we uploaded an outdated model and notebook to Google Drive, and because of this, we decided to link a team member's GitHub repository with the latest notebook and model, which can be accessed here.

2. References

- **[1] K. A. A. Chatara, E. Fielding, K. Sano, and K. Kitamura, "Data downlink prioritization using image classification on-board a 6u cubesat", 2024**
- **[2] F. Almeida, "Automatic retraining for machine learning models: tips and lessons learned", 2022**