Report on Cloud Presence Prediction using Deep Learning

Overview

This project aimed to predict cloud presence in satellite images using a deep learning model. The workflow consisted of several steps, including data preparation, model training, prediction, and integration of results into netCDF files. Below is a detailed report of the procedures followed and the results obtained.

Steps Taken

1. CSV File Creation for Cloud Intervals

- A CSV file was created to specify the intervals of cloud presence in the satellite images. This file contained the filenames of netCDF (.nc) files, the corresponding orbit numbers, and the intervals indicating the start and end frames of cloud presence.

2. Generating Training Images

- Training images were generated from the netCDF files using the cloud intervals specified in the CSV file.
- **Script**: `training_images.py`
  - This script reads the CSV file and extracts radiance data from the netCDF files.
  - Radiance data corresponding to cloud intervals were saved as images in the 'cloud' folder, while the rest were saved in the 'clear' folder.
  - Random sampling was used to decide whether to save a frame as a cloud or clear image.

3. Model Training

- A ResNet-50 model was trained using the generated images.
- **Script**: `resnet50_training.py`
  - Images were preprocessed and augmented to enhance the model's robustness.
  - The ResNet-50 model was used as the base, with additional dense layers for binary classification (cloud vs. no-cloud).
  - The model was trained on 70% of the data, validated on 20%, and tested on the remaining 10%.
  - Early stopping and TensorBoard were used to monitor training progress.

4. Prediction on New Orbits

- The trained model was used to predict cloud presence in new satellite images.
- **Script**: `predict_all_orbits.py`
  - The script loads the trained model and reads images from the specified input folder.
  - Each image is preprocessed and fed into the model to get a prediction probability.

- Running averages of predictions were computed to smooth out the results.
- Predictions were saved to CSV files, one for each orbit.

5. Adding MLCloud to netCDF Files

- Predicted cloud presence was added as a new variable ('MLCloud') in the netCDF files.
- **Script**: `add_mlcloud_to_nc_files.py`
  - This script reads the prediction CSV files and netCDF files, then adds the 'MLCloud' variable to the netCDF files.
  - The new netCDF files with the 'MLCloud' variable were saved to a separate output folder.

Detailed Script Descriptions

`training_images.py`

- **Purpose**: To generate training images from netCDF files based on cloud intervals specified in a CSV file.
- **Process**:
  - Read the CSV file to get the intervals of cloud presence.
  - Extract radiance data from netCDF files.
  - Save radiance data as images in 'cloud' or 'clear' folders based on the intervals.

`resnet50_training.py`

- **Purpose**: To train a ResNet-50 model for cloud presence prediction.
- **Process**:
  - Load and preprocess images.
  - Augment data and prepare it for training.
  - Train the ResNet-50 model with early stopping and TensorBoard monitoring.
  - Save the trained model and performance metrics.

`predict_all_orbits.py`

- **Purpose**: To predict cloud presence in new satellite images using the trained model.
- **Process**:
  - Load the trained model and preprocess images.
  - Predict cloud presence and compute running averages of predictions.
  - Save the predictions to CSV files.

`add_mlcloud_to_nc_files.py`

- **Purpose**: To add the 'MLCloud' variable to netCDF files based on model predictions.
- **Process**:
  - Read the prediction CSV files and netCDF files.
  - Add the 'MLCloud' variable to the netCDF files.
  - Save the modified netCDF files to a new folder.

Results and Performance

- **Training Time**: The model was trained over 80 epochs using 6177 labeled images, which took approximately 9617 seconds (2 hours 40 minutes) on a powerful computer.
- **Prediction Time**: Predictions for all 6 new orbits were completed in approximately 849 seconds.
- **Model Performance**: The trained ResNet-50 model achieved >80% accuracy.

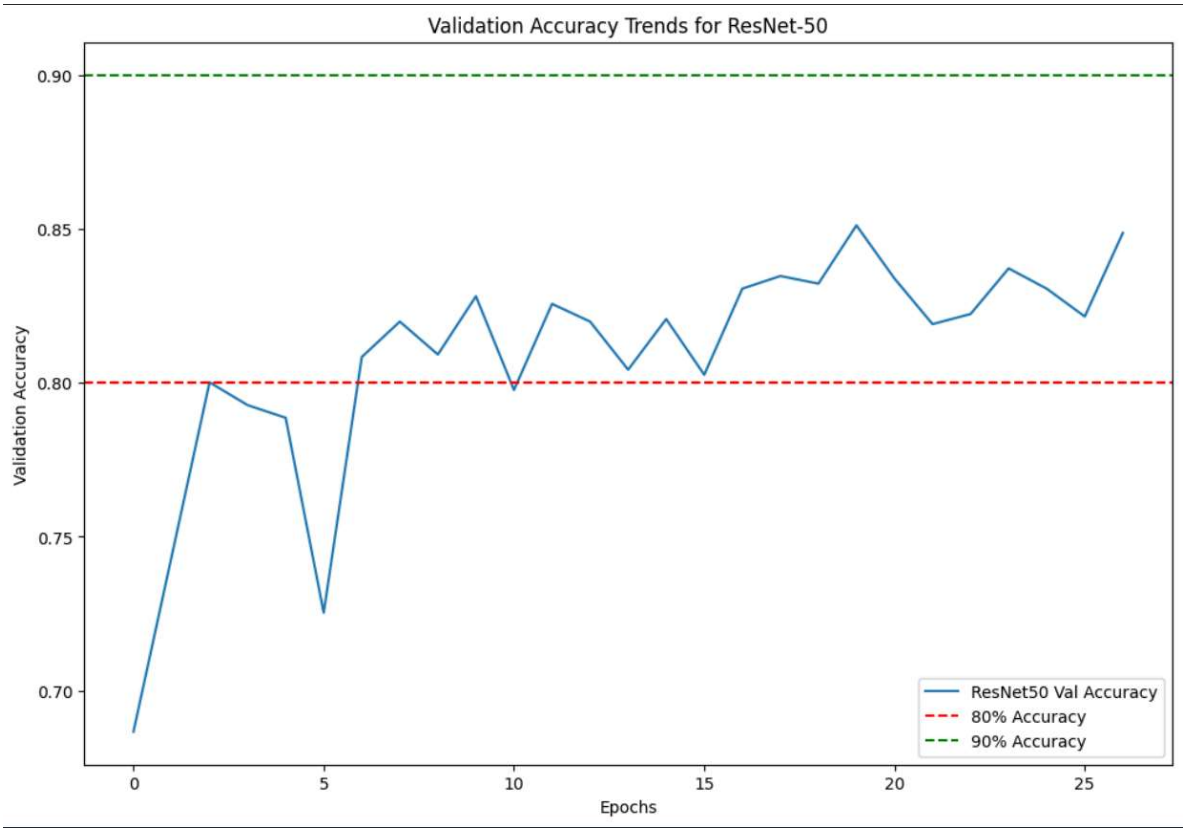
Validation Accuracy Trends for ResNet-50

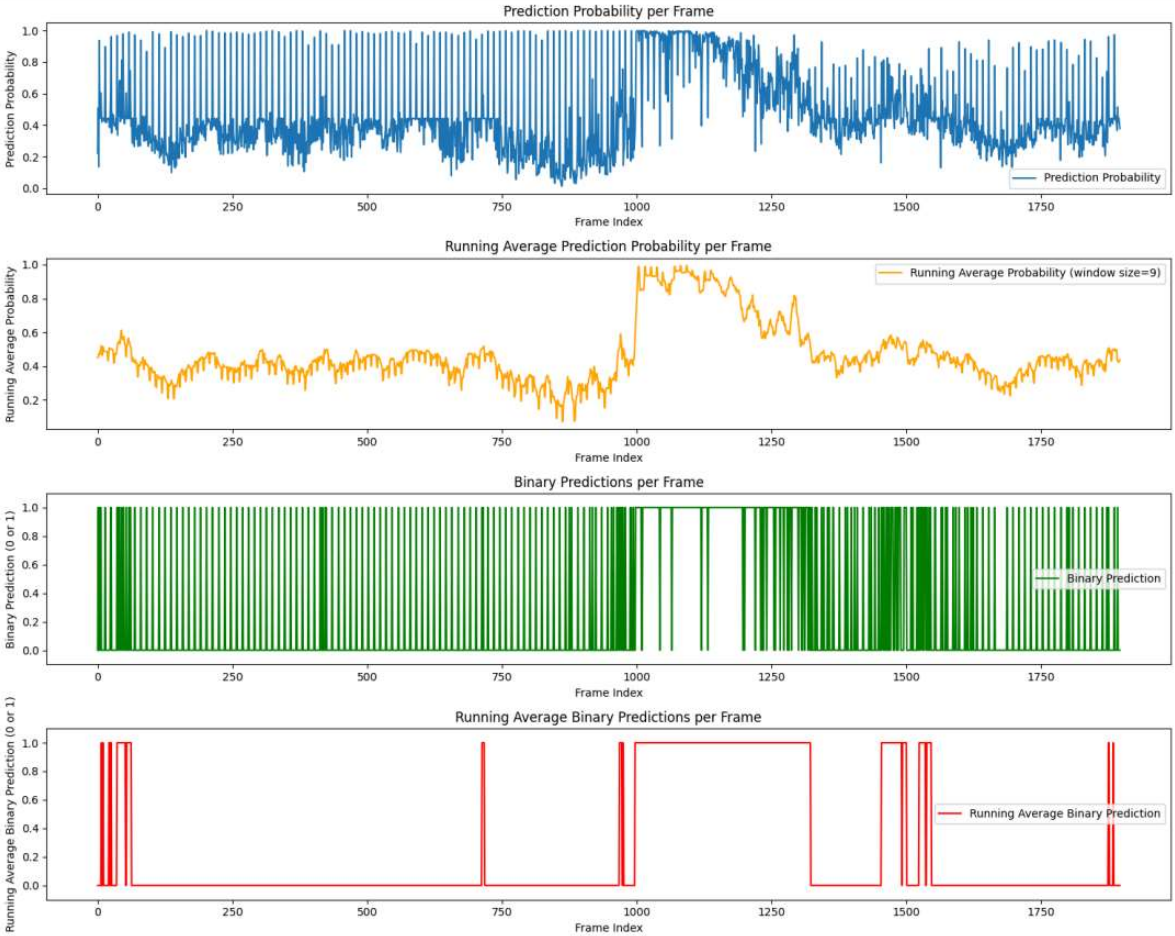
Predictions for Orbit 661

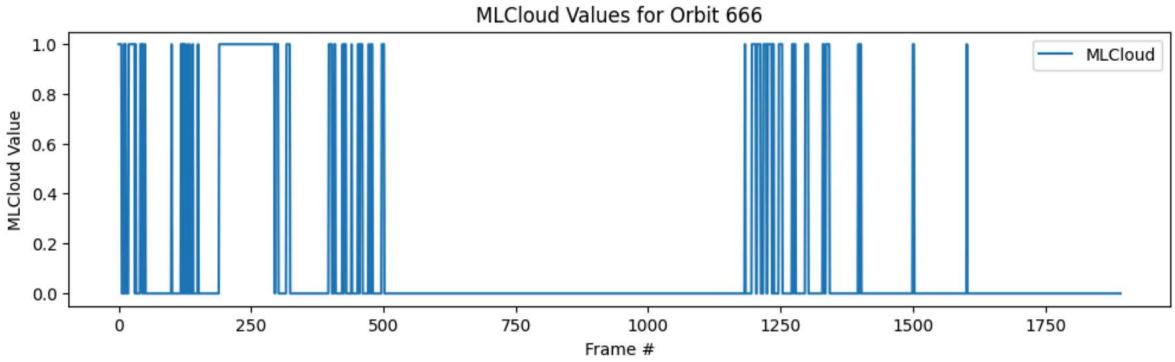
MLCloud Variable from the Newly Created .nc Files

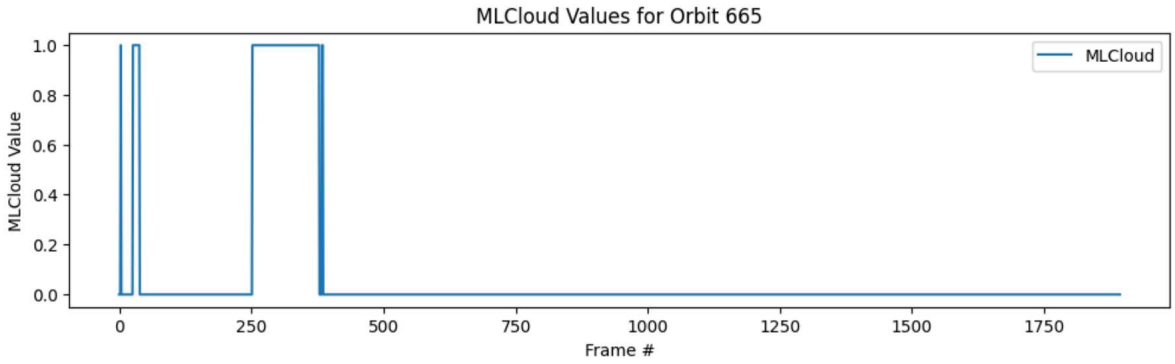
Conclusion

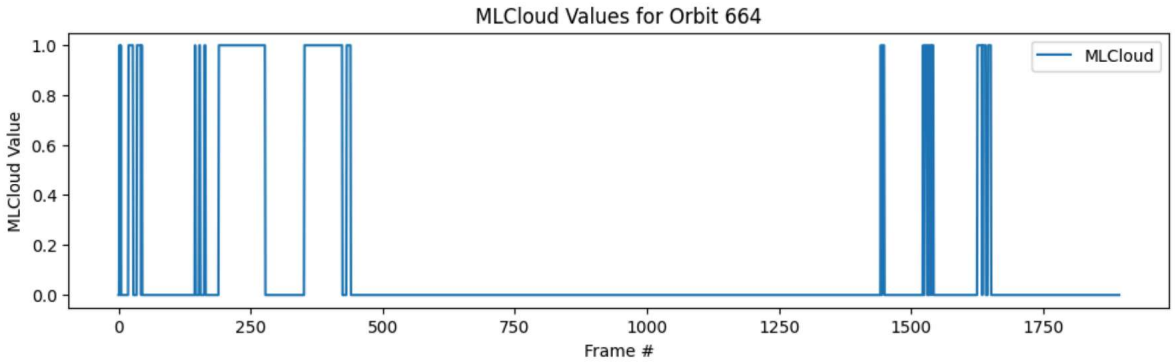
This project demonstrated the use of deep learning for cloud presence prediction in satellite images. The comprehensive pipeline, from data preparation to model training and prediction integration, proved effective and reliable. Future work could explore enhancing the model's accuracy further.

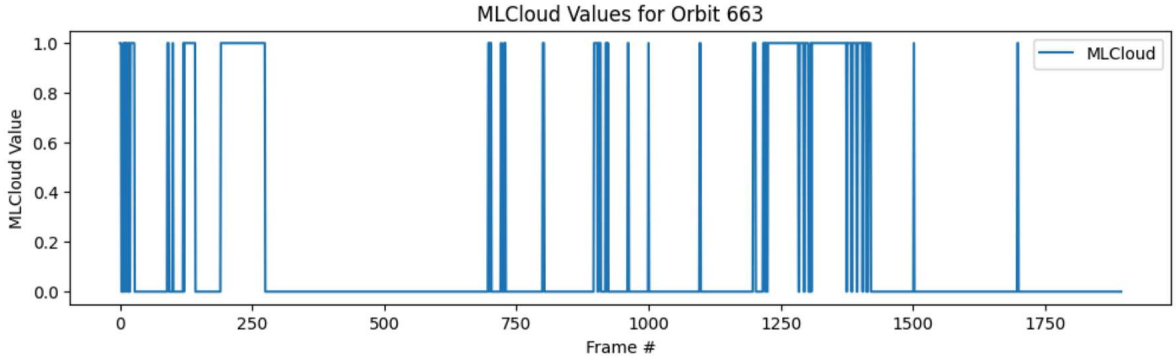


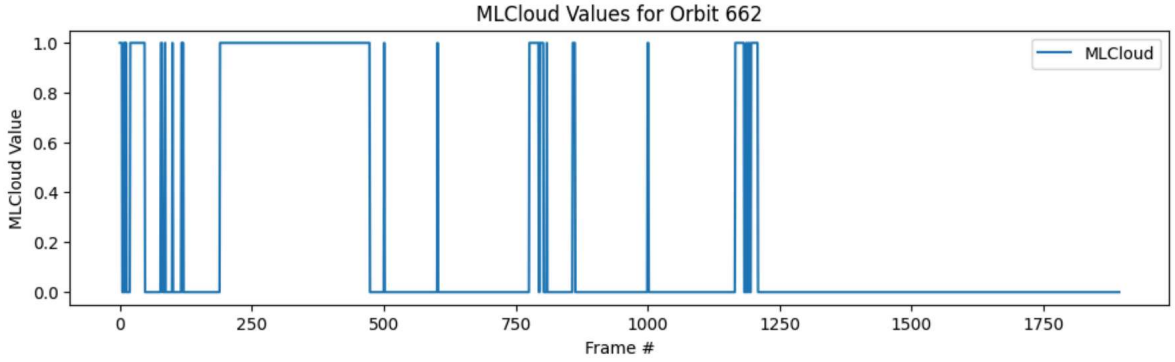

Prediction Probability per Frame
Prediction Probability
Frame Index
Running Average Prediction Probability per Frame
Running Average Probability (window size=9)
Running Average Probability
Frame Index
Binary Predictions per Frame
Binary Prediction
Binary Prediction (0 or 1)
Frame Index
Running Average Binary Predictions per Frame
Running Average Binary Prediction
Running Average Binary Prediction (0 or 1)
Frame Index

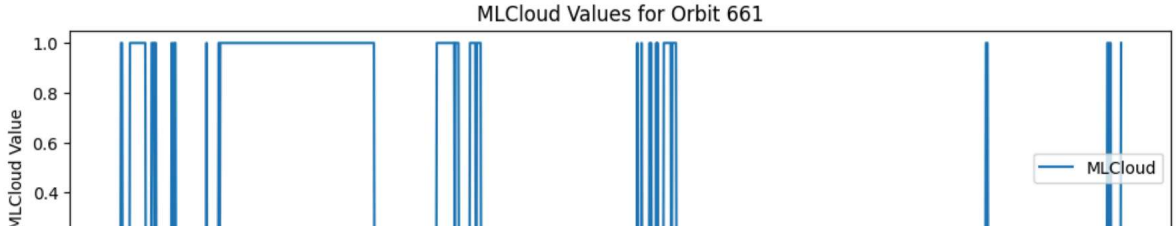

MLCloud Values for Orbit 666
MLCloud
MLCloud Value
Frame #


MLCloud Values for Orbit 665
MLCloud
MLCloud Value
Frame #


MLCloud Values for Orbit 664
MLCloud Value
Frame #
MLCloud


MLCloud Values for Orbit 663
MLCloud Value
Frame #
MLCloud


MLCloud Values for Orbit 662
MLCloud Value
Frame #
MLCloud


MLCloud Values for Orbit 661
MLCloud Value
MLCloud

In [ ]: