Satellite Orbit Classification through Machine Learning

AE 8900 MS Special Problems Report
Space Systems Design Laboratory (SSDL)
Guggenheim School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, GA

Author:
Lakshmi Kundana Kalidindi

Advisor:
Prof. Brian C. Gunter

August 4, 2023

Satellite Orbit Classification through Machine Learning

Lakshmi Kundana Kalidindi * and Dr. Brian Gunter†
Georgia Institute of Technology, Atlanta, GA, 30313

This project focuses on orbit classification of resident space objects (RSOs) using novel machine learning techniques. The goal is to classify satellite orbits into Low Earth Orbit (LEO), Medium Earth Orbit (MEO), and Geostationary Orbit (GEO) based on images taken from a sensor on the Earth. The implementation involves generating satellite images over various orbits, extracting orbital parameters from Two-Line Element (TLE) data, and developing various machine learning models to classify these orbits. Techniques such as Convolutional Neural Networks (CNNs), transfer learning, and ensemble learning have been used for accurate orbit classification. The project emphasizes preprocessing TLE data, image generation, model training, and evaluation to achieve efficient classification accuracy, thereby contributing to satellite tracking and space exploration methodologies.

Nomenclature

a	=	semi-major axis
CNN	=	convolutional neural network
Dec	=	declination
e	=	eccentricity
ECI	=	Earth Centered Inertial
FOV	=	Field of View
GCRF	=	Geocentric Celestial Reference System
GEO	=	Geosynchronous Earth Orbit (GEO)
i	=	inclination
ICRS	=	International Celestial Reference System
KRR	=	Kernel Ridge Regression
LEO	=	Low Earth Orbit
MAE	=	Median Absolute Error
MEO	=	Medium Earth Orbit

*Graduate Student, Daniel Guggenheim School of Aerospace Engineering.
†Professor, Daniel Guggenheim School of Aerospace Engineering,.

ML	=	Machine Leaning
Ra	=	right ascension
ReLU	=	rectified linear unit
RMSE	=	root mean squared error
RSO	=	resident space object
sgp4	=	Simplified General Perturbations Four
TLE	=	Two Line Element
v	=	true anomaly
Ω	=	right ascension of the ascending node
ω	=	argument of periapsis

I. Introduction

Satellite orbit classification is a pivotal aspect of space research and satellite navigation systems. Understanding the distinct characteristics and trajectories of satellites in various orbital paths—Low Earth Orbit (LEO), Medium Earth Orbit (MEO), and Geosynchronous Earth Orbit (GEO)—is crucial for effective satellite deployment, tracking, and data transmission. This technical paper presents an innovative approach to satellite orbit classification utilizing machine learning (ML) techniques.

The project involves generating synthetic satellite images based on Two-Line Element (TLE) data, which encapsulates key orbital parameters of satellites over time. These synthetic images simulate satellite observations and serve as essential inputs for training ML models. The implementation focuses on preprocessing TLE data, image generation based on orbital parameters, and employing Convolutional Neural Networks (CNNs), transfer learning, and ensemble learning methods for accurate orbit classification.

This research emphasizes the fusion of orbital mechanics, data science, and image processing to create a robust framework for classifying satellite orbits. The proposed methodology aims to contribute significantly to satellite monitoring, space exploration, and the development of efficient satellite navigation systems.

II. Synthetic data for training

A good machine learning algorithm needs a large enough dataset to train on. Since we do not have that much of ground imaging data readily available, we prepare the data synthetically. The code developed for this project is designed to generate synthetic satellite images and corresponding labels for training the ML models. The process can be divided into several steps, involving celestial data retrieval, synthetic image creation, and dataset preparation.

A. Celestial Data Retrieval

To generate synthetic images of satellites, we need their orbital elements at a certain point of time so that we can propagate them to plot their orbit. Two Line Elements (TLEs)[1] are data formats encoding a list of orbital elements of an Earth-orbiting object for a given point in time, the epoch. Each TLE consists of two text lines which include information such as the satellite name, epoch date, orbital inclinations, right ascension of ascending node, eccentricity, argument of perigee, mean motion, and several other parameters to define the satellite's orbit uniquely. The data is typically updated every few days due to natural perturbations affecting the satellite's orbit. To propagate the orbit, we chose the python skyfield[2] library that uses the sgp4 orbit propagator.

B. Synthetic Image Generation

In the preparation of the dataset, we propagated 200 LEO3, 80 MEO5 and 80 GEO7 satellites. We begin with the assumption of an Earth based sensor with coordinates: '33.75 N', '-84.39 E' in Atlanta. We use an intertial frame of reference to make sure that the synthetic images simulate the output of the earth based sensor. At each time step, the angle of satellites relative to the sensor are calculated using the dot product of the vectors from the sensor to the reference log and from the sensor to the object. We determine the distance from the sensor to the center of the FOV plane contained the observed object. The RSOs are propagated and observed for 10 minutes and a total of 16,000 images are generated. The Python package matplotlib[3] was used to plot the Ra,Dec values plotted in equi-rectangular projection. To make the training data more robust, OpenCV[4] was leveraged to add Gaussian noise to the images2468. Since it is such a short period of time, the RSOs in the images are indiscernible by human eye. That is where ML algorithms step in.

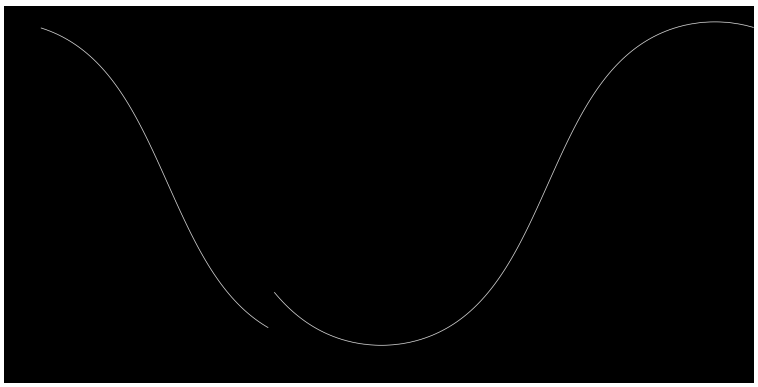

Fig. 1 Synthetic image of ISS ground track

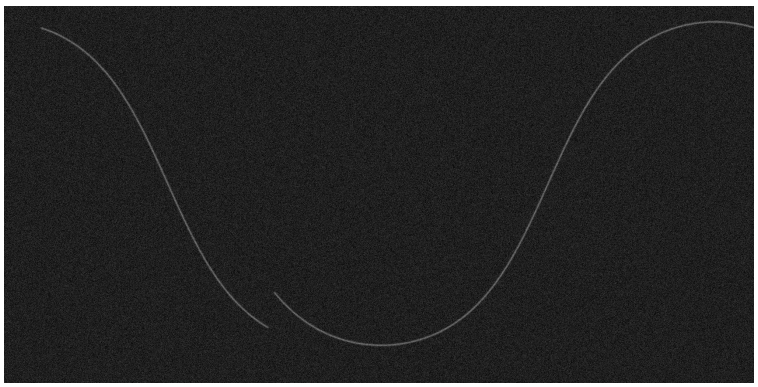

Fig. 2 Synthetic image of ISS ground track with noise

Fig. 3 COSMOS 2024 (ETALON 2)(LEO)

Fig. 4 COSMOS 2024 (ETALON 2) with noise

Fig. 5 NAVSTAR 59 (USA 192)(MEO)

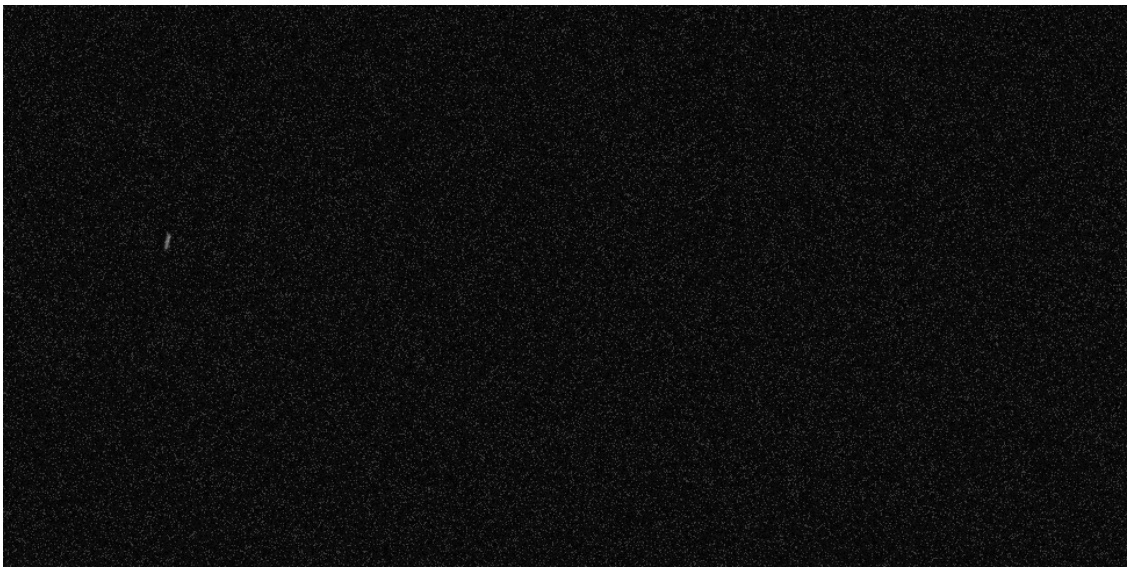

Fig. 6 NAVSTAR 59 (USA 192) with noise

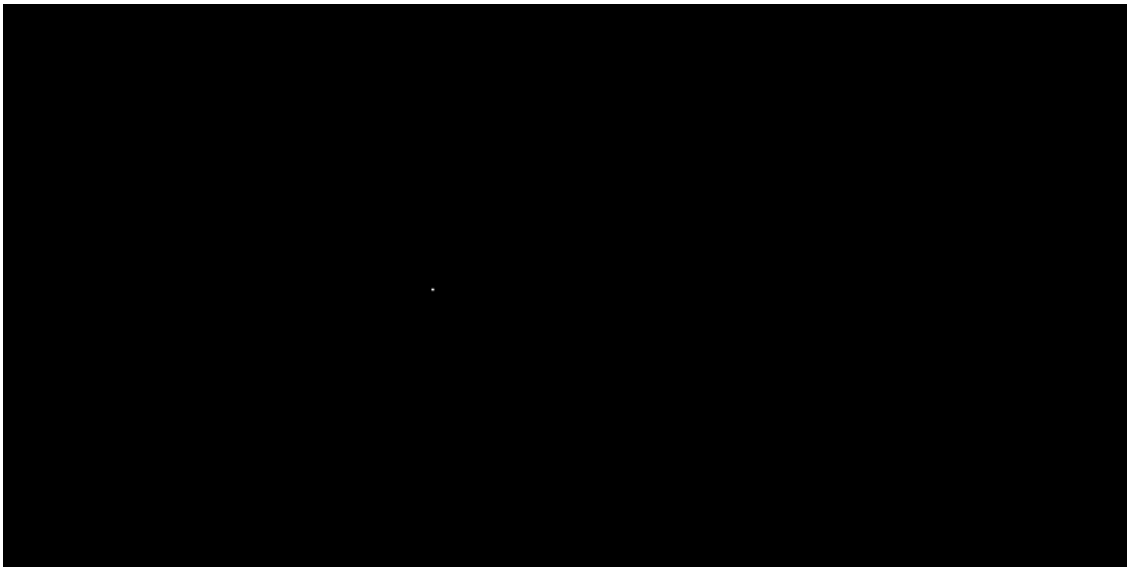

Fig. 7 INTELSAT 5 (IS-5) (GEO)

Fig. 8 INTELSAT 5 (IS-5) with noise

III. ML methods to classify satellite orbits

Since we are trying to classify the images into 3 categories, namely LEO, MEO and GEO, we need Supervised Machine Learning algorithms. From the various available ML algorithms Convolutional Neural Network (CNN)[5] has been chosen for satellite classification because of its well proven ability in multi-label classification[6]. TensorFlow/Keras[7] libraries are used to convert the images into tensors and apply machine learning models. The generated synthetic grayscale images and labels are read using OpenCV.

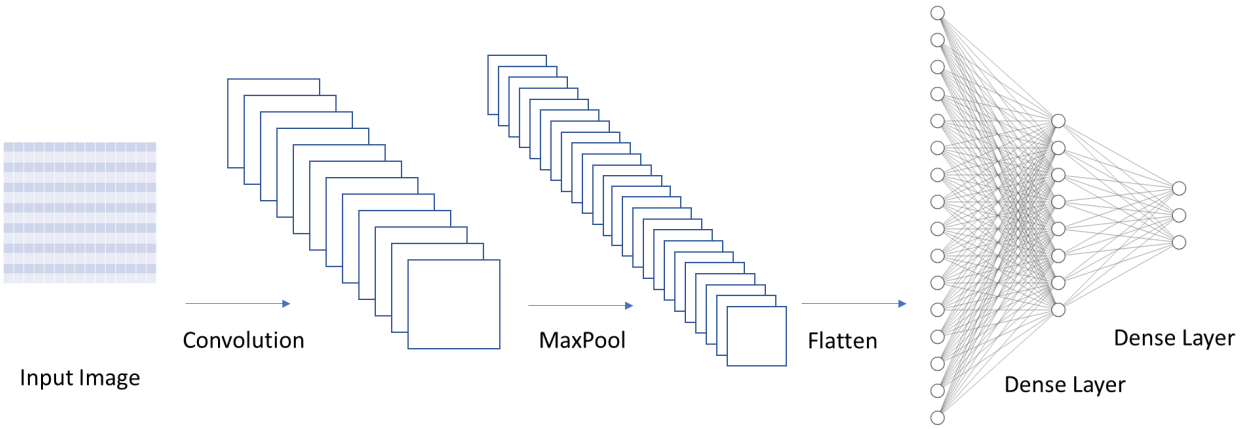

Fig. 9 CNN Architecture

The CNN architecture is built within a Sequential model from Keras. It comprises layers like Conv2D (convolutional), MaxPooling2D (pooling), Flatten (flattening data), and Dense (fully connected) layers. The last dense layer should be of the size of the number of labels required, which is 3 in our case. The model is compiled using the Adam optimizer and 'sparse-categorical-crossentropy' as the loss function for multiclass classification.

A. Model Training and Evaluation

It is important that we reshape the prepared data to fit the CNN input shape. Since the are over 16000 synthetic images in the dataset, there can be out of memory errors that arise while training the model. The model is trained by using a simple fit function and evaluated on the test set to assess its performance. The following metrics we generated from the evaluation of this model:

$$accuracy = 0.9922$$

$$Confusion\ Matrix = \begin{vmatrix} 247 & 0 & 0 \\ 1 & 628 & 0 \\ 2 & 1 & 286 \end{vmatrix}$$

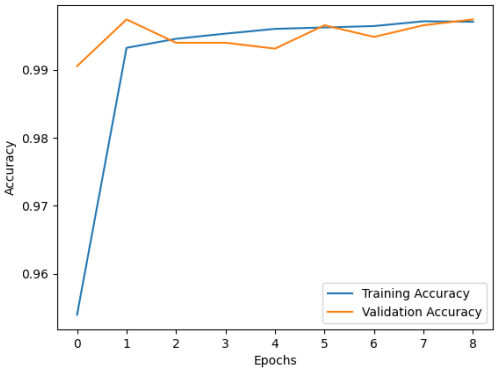

Fig. 10 Accuracy plot of CNN

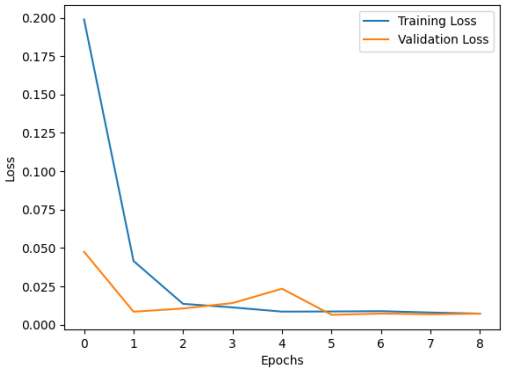

Fig. 11 Loss plot of CNN

IV. Transfer learning and ensemble methods

Training the model on a CNN for a huge dataset takes a substantial amount of time. In order to reduce the time taken to train the model and make the model more robust, there are many techniques available. In this paper, we explore the concepts of Transfer Learning[8] and Ensemble Learning[9] to classify our satellite image dataset.

A. Transfer Learning

The transfer learning technique involves leveraging pre-trained models on large image datasets (like ImageNet) and reusing their learned features to solve similar problems with smaller datasets. Utilizing pre-trained networks like VGG16[10] and ResNet50, which have been trained on vast datasets like ImageNet, provides a powerful feature-extracting base due to their deep and complex architectures. The early layers of a pre-trained CNN, which capture basic and generic features (like edges, textures, etc.), are reused. These features are often relevant across different image recognition tasks and provide a strong and robust starting point for new tasks. For satellite image classification, these models can be adapted by replacing and retraining the final layers to recognize specific patterns and fine-tuned to classify satellite images. This approach significantly reduces the training time and computational resources required, as the intricate features learned by the networks can effectively discriminate between the nuanced details present in our application.

We trained 3 models using two different base layers. Different dense layers are studied for performance of VGG16.

Table 1 Transfer Learning Models

Model	Base Model	Dense Layers	Accuracy
Model 1	VGG16	128 with activation= relu, 3 with activation = softmax	0.9965
Model 2	ResNet50	128 with activation= relu, 3 with activation = softmax	0.9976
Model 3	VGG16	64 with activation= relu, 3 with activation = softmax	0.9918

ResNet50 and VGG16 need image inputs with three channels. The images we generated have only 1 channel. Hence images were preprocessed to stack the same data into three channels before loading into the model. To handle out of

memory error due to the huge size of images, we used another technique called iterative learning which is done for "Online Machine Learning Models"[11]. We used this property of training technique to our advantage to creatively handle huge datasets. In this, the images were divided into batches of 1000 and then the model was trained upon the saved model in the previous iteration.

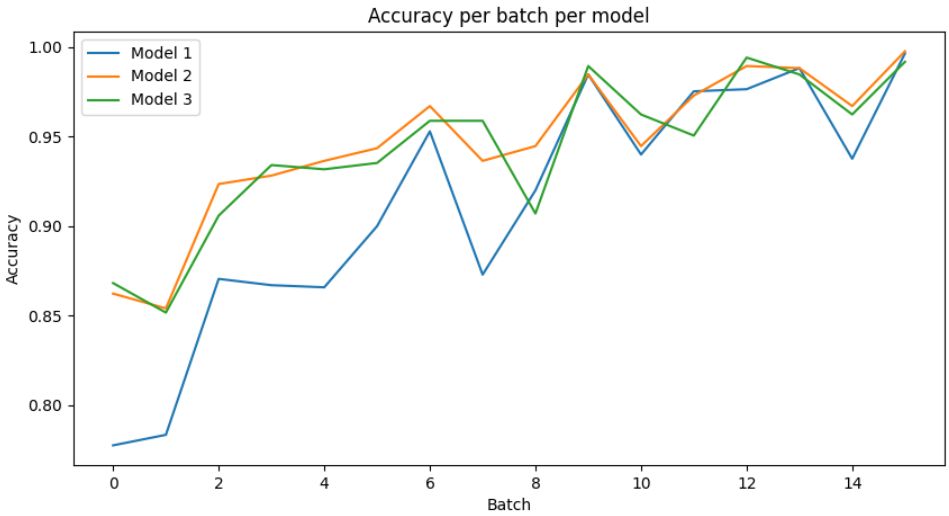

Fig. 12 Accuracy plot of Transfer Learning models with VGG16(Dense128), ResNet50 and, VGG16(Dense64)

B. Ensemble Learning

Ensemble learning is a robust machine learning paradigm where multiple models, often called "weak learners," are trained to solve the same problem and then combined to improve the overall performance[12]. This combination can be done in several ways, such as averaging, weighted voting, or stacking. The underlying principle is that a group of diverse models can complement each other's predictions, leading to better generalization and reduced likelihood of overfitting. By pooling the strengths and mitigating the weaknesses of individual models, ensemble methods, like Random Forests or Gradient Boosting, often achieve higher accuracy than any single model alone. In this project we explore the usage of Voting classifier[13].

After preprocessing the data, the three transfer learning models are used as the weak learners for the ensemble classifier. We use hard voting for this classifier1. In hard voting, the predicted class label is determined by the majority vote across all models; that is, the class which gets the most votes from different models is chosen as the final prediction. This method is simple yet effective, especially when combining models that have different strengths and weaknesses, as it can lead to a more robust and generalized performance than any single model alone.

Implementing the Ensemble ML algorithm generated an accuracy of 0.9952. This is noted to be less than the accuracies of CNN and Model 1, Model 2; but is greater than Model 3.

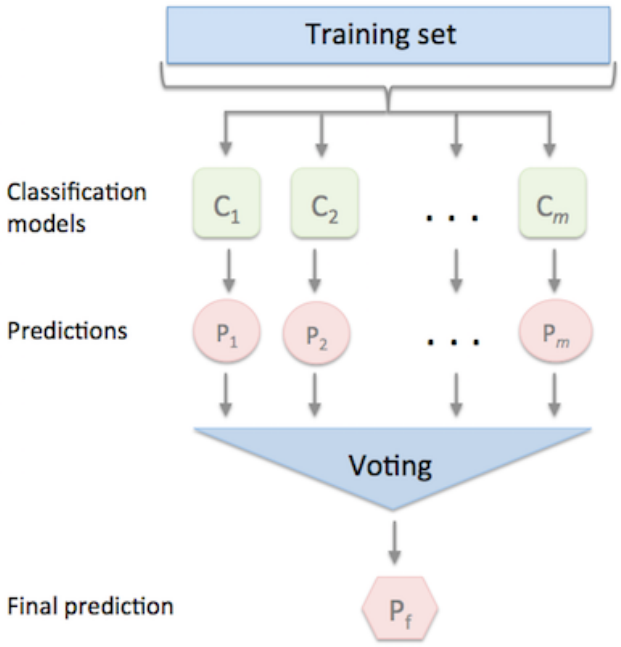

Fig. 13 Ensemble Voting classifier [14]

C. A comparision of performance of different ML methods

Comparing the accuracies of all the models tested side by side:

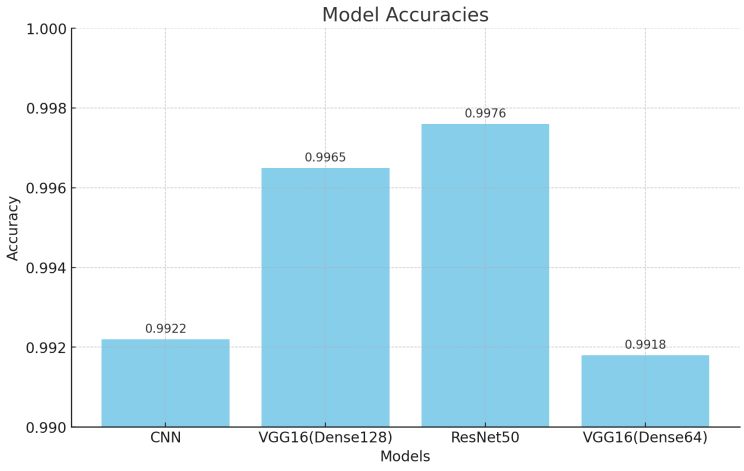

Fig. 14 Model performance comparison

V. Challenges and Improvements

A. Image Generation

While creating the images, we did not take into account the earth oblateness effect, radiation pressure, atmospheric interferences and distortions. Although noise was added to the images, they were generated in ideal conditions. Since

images were taken over a 10 minute time period, the MEO and GEO satellites were barely visible, but the models still managed to predict the class correctly. But for real images, noise might be much higher and irregular and the accuracy might drop significantly due to it.

B. Machine Learning Models

While training the models, there were frequent "out of memory" issues due to the large size of the dataset contaning 16,000 images. To overcome this, we have used iterative machine learning. However, it is important to note that other methods should be explored when the dataset becomes even larger. Another problem ML models frequently face are overfitting and underfitting. When the available data is limited, overfitting can occur in CNNs. To mitigate this, we have employed transfer learning which uses a pre-trained network as a feature extractor and then training a smaller model on top of these features can act as a form of regularization. It prevents the model from learning overly complex patterns that fit the training data too closely. However, it's important to note that transfer learning is not a panacea for overfitting. Overfitting can still occur if the fine-tuning phase is not managed properly, especially if the new dataset is too small or too different from the data on which the model was originally trained. Proper techniques like data augmentation, dropout, and early stopping should still be employed as necessary to further combat overfitting.

VI. Conclusion

These models should be able to predict the orbit classification of an RSO by providing a real image from a telescope taken over a few minutes of time with fairly good accuracy.The project achieved an exceptional accuracy of 99.22% utilizing Convolutional Neural Networks (CNNs) in accurately categorizing satellite orbits into Low Earth Orbit (LEO), Medium Earth Orbit (MEO), and Geostationary Orbit (GEO) based on synthetic satellite images generated from Two-Line Element (TLE). Transfer Learning methods using ResNet50 and VGG16 have shown very good accuracy with quick training rate. Although, the ensemble machine learning approach showed good accuracy standalone, it did not yield greater results than most of these models. This behaviour is expected since the data are very close to each other and the predicted class in one model might not be the same in a different model. This could be solved by using Weighted Voting Classifier where weights can be assigned to each model based on the confidence. Despite the CNN's outstanding performance, the ensemble method did not exhibit the expected classification precision. Furthermore, hyperparameter tuning, including fine-tuning of the pre-trained models and optimizing ensemble methods, could improve classification accuracy further.

More methods of Ensemble Machine Learning like Boosting, Bagging and using Random Forests need to be explored for quicker and robust classification. Using multiple sensors like Laser data along with images and use sensor fusion techniques to improve location and velocity estimation.

Acknowledgments

The research project is done as part of AE8900 course under the guidance of Dr. Brian Gunter. The code for synthetic image generation for this project is derived from the previous work of Rohan Patel who wrote it for his undergraduate research project at Georgia Institute of Technology.

References

[1] Celestrak, "Two-Line Element Set Format," `https://celestrak.org/NORAD/documentation/tle-fmt.php`, 2023. Accessed: 2023-02-12.

[2] Rhodes, B., "Skyfield: High precision research-grade positions for planets and Earth satellites generator," `https://rhodesmill.org/skyfield/`, 2019. Ascl:1907.024.

[3] Hunter, J. D., "Matplotlib: A 2D Graphics Environment," *Computing in Science Engineering*, Vol. 9, No. 3, 2007, pp. 90–95. https://doi.org/10.1109/MCSE.2007.55.

[4] Bradski, G., "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2008.

[5] Yamashita, R., Nishio, M., Do, R., et al., "Convolutional neural networks: an overview and application in radiology," *Insights Imaging*, Vol. 9, 2018, pp. 611–629. https://doi.org/10.1007/s13244-018-0639-9, URL https://doi.org/10.1007/s13244-018-0639-9.

[6] Lydia, A. A., and Francis, F. S., "Multi-Label Classification using Deep Convolutional Neural Network," *2020 International Conference on Innovative Trends in Information Technology (ICITIIT)*, 2020, pp. 1–6. https://doi.org/10.1109/ICITIIT49094.2020.9071539.

[7] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," , 2015. URL https://www.tensorflow.org, accessed: 2023-02-12.

[8] Hosna, A., Merry, E., Gyalmo, J., et al., "Transfer learning: a friendly introduction," *Journal of Big Data*, Vol. 9, 2022, p. 102. https://doi.org/10.1186/s40537-022-00652-w, URL https://doi.org/10.1186/s40537-022-00652-w.

[9] scikit-learn developers, "Ensemble methods - scikit-learn documentation," https://scikit-learn.org/stable/modules/ensemble.html, 2023. Accessed: 2023-02-12.

[10] Simonyan, K., and Zisserman, A., "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv*, 2014. https://doi.org/10.48550/arXiv.1409.1556, URL https://doi.org/10.48550/arXiv.1409.1556.

[11] Emily A. Kringle, C. E., Evan C. Knutson, and Terhorst, L., "Iterative processes: a review of semi-supervised machine learning in rehabilitation science," *Disability and Rehabilitation: Assistive Technology*, Vol. 15, No. 5, 2020, pp. 515–520. https://doi.org/10.1080/17483107.2019.1604831, URL https://doi.org/10.1080/17483107.2019.1604831, pMID: 31282778.

[12] Dietterich, T. G., "Ensemble Methods in Machine Learning," *Multiple Classifier Systems*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000, pp. 1–15.

[13] Shahane, S., "Voting Classifier," https://www.kaggle.com/code/saurabhshahane/voting-classifier, 2021. Accessed: 2023-03-24.

[14] "Ensemble voting classifer," , 2023. URL https://rasbt.github.io/mlxtend/user_guide/classifier/EnsembleVoteClassifier/, accessed: 2023-12-10.