Satellite Collision Risk Prediction

Kedaar Vyas **Steven Su** **Varun Unnithan** **Sathya Gnanakumar** **Pablo Garcia-Arroba**
Department of Computer Science
University of Maryland
College Park, MD 20742
{kvyas,stevensu,varunun,sgnanaku,pgarroba}@terpmail.umd.edu

Abstract

As the amount of space debris orbiting the Earth increases, the risk of collision with satellites becomes an important issue to address. Classical methods suffer from flaws such as a high computational burden and certain inaccuracies. Inspired by the Kelvins Collision Avoidance Challenge, this project aims to develop neural networks to accurately predict the future risk of a conjunction event between 2 objects floating in space and surpass the performance of prior baselines. We use a dataset of Conjunction Data Messages (CDMs), which are the primary format used to alert spacecraft operators of potential collisions, and extract the orbital kinematics of each space object, which includes fields like position and velocity at that time, as well as covariances and the propagated probability of collision. We then develop several types of models, including LSTMs, fully-connected networks, and transformers to utilize this data and make risk predictions. Our best model beats baselines established in the ESA competition and places 8th on the competition leaderboard. Additionally, our work reveals interesting insights on the lack of value in the time series structure of sequential CDMs, consequences of data imbalance in the dataset, and other insights that could be useful for future work.

1 Introduction

Space Surveillance Networks and the European Space Agency (ESA) estimate there are over 170 million pieces of space debris larger than 1 mm currently orbiting the Earth [4]. This raises the concern of satellite collisions. The advent of relatively cheap satellites such as CubeSats, along with an industry trends towards large constellations of low-earth orbit (LEO) satellites (e.g., SpaceX's Starlink, Amazon's Project Kuiper, IntelSat's OneWeb, etc.) will result in increased risk for collision. This is further exacerbated by the fact that most LEO satellites have a relatively short expected life times Office [8]. In these circumstances, it is clear that we may lack sufficiently fast and accurate methods for assessing viable maneuvers and responses to these conjunction events.

To address these shortcomings, we propose a machine learning-based approach to model the time evolution of collision probability. By leveraging the dataset from Kelvins Collision Avoidance Challenge, we frame the problem as a time-series prediction task. The dataset captures detailed, temporal dynamics of conjunction events.

In the following sections, we discuss related works, describe our methodology, present experimental results, and conclude with insights into the implications of our approach for space operations. The code for this project can be found at this link.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

2 Related Works

2.1 Classical methods for computing risk

Accurate risk assessment of satellite collision events is critical for space situational awareness and collision avoidance. Classical methods for quantifying this risk involve transforming observation uncertainties into covariance ellipsoids, which serve as probabilistic representations of satellite positions and velocities. Covariances at the current time are propagated forward until the Time of Closest Approach (TCA) between satellites using statistical algorithms such as Hall's method. Hall's method uses numerical propagation algorithms that predict satellite positions and their associated uncertainties as they approach conjunction events [6].

However, due to the nature of these algorithms, they experience numerical instability, with long-duration propagation yielding inflated covariance ellipsoids. As the time to the closest approach decreases, the evolving covariance ellipsoid often becomes distorted or "diluted." This phenomenon results from the dynamic interplay between decreasing positional uncertainties and the constraints of the propagation models [2]. Until this moment, classical methods may overestimate the risk of collision, potentially leading to suboptimal or premature collision avoidance maneuvers.

2.2 ESA Collision Avoidance Challenge

With this issue being a prevalent one that many major aerospace organizations and constellation operators need to deal with, the European Space Agency (ESA) proposed a challenge in 2019 to "to build a model that makes use of the CDMs (Conjunction Data Messages) recorded up to 2 days prior to the closest approach to predict the final risk" [10]. This challenge provided the dataset that we ended up using to train our model, which was selected features and fields from CDM data provided by the US Space Surveillance Network. Our primary goal in this project was to match or ideally beat the baselines provided in the competition and achieve a high position on the competition leaderboard.

2.3 Machine learning approaches

The competition highlighted significant limitations of baseline machine learning approaches. Teams that relied solely on these pipelines achieved low test scores with high training scores, suggesting extreme overfitting. This discrepancy highlighted the importance of designing specific approaches dictated by the most important features of the dataset, in large part because the test set contained a higher proportion of high-risk events compared to the training set. Furthermore, the dataset's evaluation metric, the F2 score, emphasized recall over precision, making it critical to address false negatives. This is an area where naive approaches often performed poorly.

Top-performing teams incorporated careful feature selection to improve their results. For example, the "Magpies" team reduced the original 103 features to seven key attributes, such as time to closest approach (TCA), Mahalanobis distance, and position covariance determinants. These features were complemented by generating the mean and standard deviation of risk values over prior CDMs and the total number of CDMs issued. The Magpies team employed a Manhattan-LSTM-based Siamese architecture trained on pair examples to capture temporal dependencies between events. This approach, combined with hyper-parameter tuning and ensemble machine learning methods, led to significant performance improvements.

3 Data

This dataset contains 13,154 unique events in the training set with an average of 12 CDMs per event. The test dataset contains 2,167 unique events with an average of 11 CDMs per event. Features in the CDMs include a computed risk score (represented in log base-10 format), the amount of time prior to closest approach the CDM was issued, and numerous physics-related values such as relative positions and velocities, covariances, and eccentricity. For both datasets, following the competition structure, the last CDM (specifically, its risk score) is used as the ground truth, and all prior CDMs are treated as inputs. One final detail to consider is that numerical risk scores can be classified as high (risk >= -6) or low risk (risk < -6).

4 Approach and method

4.1 Feature selection

To improve model efficiency, we selected 20 features from the CDM data using LightGBM and RandomForest Regressor to rank feature importance. Then, we applied known physics formulas pertaining to satellite positioning and trajectory calculation. On average, 4.4% of entries per feature were null. LightGBM was thus applied twice: once dropping null values and once forward-filling them. Each regression yielded similar features, which included specific covariance and orbital features that were biased towards the chaser object, as it tends to be the less rigorously tracked object in the system. We applied a holistic review of features using the CCSDS standards [3] on top of this regressed output and selected the following features: max_risk_scaling, max_risk_estimate, time_to_tca, mahalanobis_distance, miss_distance, relative_position_r, relative_position_t, relative_position_n, relative_velocity_r, relative_velocity_t, relative_velocity_n, c_weighted_rms, t_weighted_rms, c_sigma_tdot, t_span, t_j2k_inc, c_j2k_inc, c_sigma_r, c_crdot_t, c_sigma_n, c_sigma_t, c_sigma_r, c_ctdot_t, c_position_covariance_det and t_position_covariance_det.

4.2 Competition loss function

The competition which inspired this project developed their own custom loss function to optimize. This loss function is $L(r, \hat{r}) = \frac{1}{F_2} MSE*(r, \hat{r})$ where F_2 is an F1 score with $\beta = 2$ computed from converting predictions and ground truth numeric risk values to binary high and low risk classifications. $MSE*$ is the standard mean-square error function. To place an emphasis on high-risk events due to their importance in real-life collisions, this is only calculated on the subset of prediction and ground-truth pairs where the ground-truth risk can be classified high risk. Predictions were clipped near the classification threshold of -6 to minimize penalties from the MSE component, a clamping which we applied to all predictions.

4.3 Data processing and cleaning

Following best practices, several steps to clean and process the data were taken. Only columns from the feature selection process were retained. A handful of other important columns were retained, such as the risk, event ID, and TCA columns. CDMs within events were then placed in descending order by time to closest approach. The filtered columns were then scaled to the training dataset-wide maximum for that specific column. This allows these features to follow the same approximate range and avoid bias towards features that have naturally larger unscaled ranges. Standardization was considered as an alternative, but no meaning impact was observed in model outcomes.

Missing data was handled in 2 ways. First, missing data was imputed using a forward fill, within a given event, to avoid erroneous forward-fills between events. Any events missing a feature entirely were filled with a training dataset-wide average for that feature. Finally, the training set was modified to better represent the test set by dropping certain CDMs based on 2 conditions. Events with less than 2 CDMs were dropped, because the model needs a ground truth (the last CDM where the final risk score is extracted) and a prior CDM to use as model input. Also, the test set only contains events where the last CDM (used for ground truth), had a time to closest approach less than two days.

4.4 Same-length batching

LSTMs require batches that contain samples with uniform sequence lengths. Common practices for resolving this issue include padding or grouping data such that batches naturally contain samples of the same length. Brief experimentation showed poor results for padding, potentially due to techniques such as "zero" data or forward/backward fills distorting the physics relationship of the features. Therefore, we chose to implement a custom dataloader that prepares batches of the same length by grouping data. To allow for randomization to the furthest extent, the following process is used. First, events are grouped according to the number of CDMs in each event. Within each group, the order of events are randomized, and then split into as many batches of fixed, pre-determined size b as possible. The last batch in the group may fewer have than b events. Finally, all batches from all groups are collected and the order of batches are randomized.

4.5 Data oversampling

Both the training and test datasets are heavily imbalanced. High-risk ground-truth samples form only 1% of the overall dataset, with the rest being low-risk samples. Furthermore, high and low-risk ground truth events can be further divided into two subgroups. The first are non-anomalous events, where the last available risk score and the final ground-truth risk score are classified the same (i.e., as high-risk or low-risk). The second subgroup are anomalous events, where there was a change in classification from the last available risk score to the ground truth risk score. For high-risk ground truth events, anomalous events make up 21% of the data while for low-risk ground truth events, anomalous events represent 3% of the data.

Due to this data imbalance, particularly the low representation of high-risk ground truths, which are the primary focus of the competition loss function, we chose to over-sample certain subsets of data points. The specific data points over-sampled will be discussed in the Experiments section. However, the methodology remains the same: the oversampling is controlled by a list of events E to be over-sampled and an oversampling factor t representing how many times E is repeated. In context of the custom dataloader mentioned above, each group of k-length CDMs prior to randomization and batching will have t additional copies of E_k inserted where E_k is all events in E with k CDMs available.

4.6 Model architectures

Three model paradigms were explored in this project. LSTMs and Time-Series transformers were intended to utilize the time-series nature of the data. The fully-connected model was included as a model that ignored any time series component by using only the last available CDM in the model input. We also note that these model architectures to our knowledge are novel in context of the competition. We were unable to find these approaches in published literature from the competition.

4.6.1 LSTM

Recurrent neural networks are a class of neural networks well-suited for sequential data. We chose to use LSTMs, a variant RNN, to better handle long-term dependencies for time-series problems. Two primary variants of LSTM model structures were explored.

The first variant is a relatively simple architecture with a single LSTM layer followed by one or more fully-connected layer. We experimented with having the model directly predict a numerical risk value as well as binary classification (in which the final FC layer would have a sigmoid activation function). For purposes of evaluating with the competition loss, binary predictions were mapped to fixed risk values (e.g., -5 for a positive prediction of high-risk and -6.0001 for a negative prediction of high-risk, i.e., low-risk). These values were tuned to achieve the best performance against the validation set before the models were tested against the final test set.

The second variant is a complex model that predicts both a binary classification of high and low risk along with numerical risk scores. Similar to the first variant, the model begins with a shared LSTM layer and a fully-connected layer. After the shared layers, however, the model architecture diverges into two different fully connected networks forming a "dual architecture". The left side of this architecture is a final fully connected layer with sigmoid activation predicting a binary classification of whether the final risk score is high risk or not. The right side of this architecture is a deeper fully connected network with multiple layers outputting a numerical risk score. However, the input of this fully connected network accepts the outputs of not only the shared layer, but also the binary prediction of the "left side" of the architecture and in some versions, the last CDM in the original input. The inspiration for this architecture is that the binary classification task is easier compared to numerical risk prediction. However, to optimize the MSE loss component of the competition metric, accurate numerical risk predictions are necessary. Therefore, by training both outputs in the same model, the "left side" of the architecture solving the easier problem can help improve the "right side" of the architecture solving the more difficult problem of numerical risk prediction.

Binary predicting models were trained with the natural choice of binary cross-entropy with a positive weight to help account for class imbalances. The positive weight was tuned as a hyperparameter to balance recall and precision to maximize the F2 component of the competition loss metric. Direct numerical risk predictions were trained with Huber loss with the delta parameter also tuned. Huber loss was chosen due to models exhibiting higher performance when trained with this loss function. We believe that higher sensitivity to outliers of MSE makes it an inferior choice to Huber, hence why Huber loss exhibited higher performance. For the “dual-structure" models, BCE and Huber losses were combined to train the model overall, with each loss computed on their respective model outputs.

4.6.2 Fully-connected

The fully-connected model structure used in this project follows the same form as the previous “dual architecture" LSTM. However, the initial LSTM layer is instead replaced with a fully-connected model and the last available input CDM is used instead of the entire time series. All other aspects of the model, including training losses and procedures primarily remain the same.

However, as this model performed the best, we performed further experiments training this model on different subsets of the training data to potentially improve performance. Specifically, as discussed in the Data oversampling section, the training data can be divided into high and low-risk ground truths as well as anomalous and non-anomalous events within. Experimentation involved training the model on various combinations of these subsets and evaluating performance on full validation sets and the final test set. Note that brief experiments were performed training models for multiclass classification against the four subsets of data rather than simple binary high and low risk classification. However, this appeared to hurt model performance significantly (likely due to significant imbalance in the dataset) and these efforts were quickly abandoned.

4.6.3 Time-Series transformer

A Time Series Transformer (TST) is a neural network architecture designed to process sequential data by combining mechanisms like self-attention and positional encoding. This architecture enables us to capture complex temporal dependencies in sequential data, something that is essential for accurately modeling and predicting risk maneuvers in collision avoidance scenarios like this one using Conjunction Data Messages (CDMs).

This model undergoes the same data cleaning and preparation process as the previous LSTM model described above. In order to effectively handle complex patterns required in the time-series nature of the data, the architecture of this model begins with a linear, fully connected layer that projects the data into a higher-dimensional space. The model then generates a positional encoding of the data in order to adapt the transformed with “sense” of time. This process allows the model to undergo a time series analysis. The projected data is then encoded using a multi-layer Transformer encoder that uses self-attention to capture sequential patterns in the data. Finally, the last layer of the model corresponds to a fully connected linear layer that maps the output to a binary classification, distinguishing between high-risk and low-risk events.

While processing the data, it was critical to handle the significant class imbalance that exists in the data (with a majority of low-risk samples and far fewer high-risk samples). This process involved oversampling high-risk samples based on a specific factor given in the creation of the training data (oversample factor = 10), ensuring greater representation during batch creation. The data process also involved the standardization of the data and the generation of attention masks to focus on the most important and relevant features.

Two distinct functions were used for different purposes to train, evaluate, and test the model. During training, the model utilized Binary Cross-Entropy with Logits for binary classification tasks like the output of the model, which predicts high-risk vs. low-risk. A positive weight parameter was added to handle class imbalance and prevent the model from becoming biased toward predicting low-risk. For purposes of evaluating and comparing with the competition loss, binary predictions were mapped to fixed risk values (e.g., -5 for a positive prediction of high-risk and -6.0001 for a negative prediction of high-risk, 4 i.e., low-risk). This loss incorporated the F2 score in order to emphasize the importance of recall over precision to minimize missing a high-risk collision.

5 Experiments and Results

5.1 Key baselines

The competition on which this project is inspired provides several baselines which we will describe here to provide context for model results discussed below. The first baseline is a constant risk prediction of -5, which achieves a competition loss of 2.5. The second baseline is a naive forecast using the risk score of the last available input CDM as the prediction. When clamping predictions to -6.0001, this baseline achieves a competition loss of 0.694. We note that in the competition, this baseline was difficult to beat, with only 12 teams out of 97 able to create models that beat this baseline. Finally, the scores of top performers on the competition leaderboard provide another baseline to compare our results against. The first-place team achieved a competition loss of 0.555, the fifth-place team a loss of 0.614, and the tenth-place team a loss of 0.673. As mentioned in earlier sections, our primary goal was to develop a model that could outperform prior work and the baselines.

5.2 LSTM models

The first variant of the LSTM models did not perform well. The version of this variant making binary predictions achieved a competition loss of approximately 1.1, which is worse than the naive forecast baseline. The variant attempting to directly predict a numeric risk score was worse, with a loss significantly greater than the constant rate baseline. However, the second variant of the LSTM models, i.e., the dual-output architecture performed significantly better, achieving a loss of 0.6799. This beats both baselines and achieves 11th place on the competition leaderboard.

After hyperparameter tuning, in these models, the stochastic gradient descent with momentum optimizer with cosine annealing was found to be superior to Adam with a step scheduler. The optimal learning rate was determined to be 4e-4 with momentum of 0.95 and weight decay 1e-4 for 180 epochs. An oversampling factor on events with high-risk ground truths of 50 to 70 and a positive weight for binary cross entropy between 1 and 3.25 was ideal for addressing class imbalance issues. A single LSTM layer was best when hidden state size of 12. For simple models and the “left-side" of the dual-architecture, 128 neurons per FC layer with 1-2 layers are best depending on the specific architecture. For the “right-side" of the dual-architecture, 64 neurons for two layers was found to be best. Finally, dropout to prevent overfitting of 0.1 to 0.2 was best depending on the specific model.

5.3 Fully-connected model

The fully-connected model surprisingly performed the best out of all models developed and tuned in this project. Most hyperparameters tuned in the previous LSTM models section apply to this model, with the exception of the number of neurons per layer in the “right side" of the dual architecture which was decreased to 64. Additionally, the model appeared to perform better when trained only on a small subset of the dataset, specifically events that indicated high risk in the last available input CDM (regardless of ground truth risk). After tuning, this model achieved a competition loss of 0.6322, which exceeds with comfortable margin both baselines and ranks 8th place on the final competition results leaderboard. This model achieves an F2 score of 0.7544 with recall of 0.817 and precision of 0.6 which shows somewhat of a balance in ability to identify both high and low risk events correctly, although still favoring high risk events.

5.4 Time-series transformer model

The first step in this project was to make the model able to classify high-risk and low-risk events correctly. Throughout the fine-tuning process of this model, a high emphasis was placed on recall due to the severe consequences of a high-risk case being predicted as low-risk.

During training, a learning rate of 0.0001 was used to ensure stable convergence, while a batch size of 32 was used to balance computational efficiency and effective/quick learning. The max sequence length was dynamically set to the current longest sequence in the data to ensure the model’s ability to handle variable-length time-series data. Additionally, dropout and gradient clipping were included in the model to prevent overfitting and ensure stable training.

Training: Over the course of 20 epochs, the training loss (Binary Cross-Entropy with Logits) consistently decreased from 1.2471 in Epoch 1 to 0.7196 in Epoch 20. The F2 also improved to

0.6621, indicating the model's ability to balance precision and recall. However, the model kept producing low precisions (0.4) due to oversampling the high-risks, leading towards a high-risk prediction (that is why the recall was consistently high 0.8).

Validation: On the validation set, the recall remained relatively high (0.6) due to the model's sensitivity to high-risk cases. This reflected the high challenges of generalizing the data. The oversampling strategy introduced some biases that affected the precision of the model.

Test: Similar results were obtained when testing the model. The precision was low, while recall was moderate, demonstrating the model's tendency to evaluate events as high risk, suggesting that the model needed to handle the classification balance better. The competition loss obtained was 1.29, showing a poor performance compared to the LSel and FC models.

Results: While trying to create a model that correctly predicted high-risk and low-risk events, I prioritized recall due to the high imbalance of classes (most of the events were low-risk), even at the cost of precision, to ensure high-risk cases were not missed. However, the model oversampled high-risk cases, introducing bias during training and leading to lower precision than expected while testing. I found it very difficult to handle the class imbalance as changes in the model would either improve recall at the expense of precision or vice versa.

6 Discussion and conclusion

6.1 Dataset limitations and observations

As described in the Data oversampling section, the dataset is severely imbalanced. Additionally, it is important to note, that the events in this dataset were specifically curated for the competition and do not represent a random sample of a real-life distribution of events. Thus, while this project provides useful insights on what models are effective at modeling risk, it remains to be seen how our results may generalize in practice.

Additionally, we encountered an interesting result when training our best model. Using only a limited subset of the training data, specifically only events where the last available input CDM had a high-risk score did not negatively impact performance. In fact, performance as measured against the competition loss metric appeared to improve. This effectively meant that training speed was significantly improved and high performance could be achieved with only 3.6% of the provided data. We hypothesize that this is because the model essentially learns an improved version of the naive forecast and based on the competition metrics, it is key to distinguish whether high-risk score in the last available input CDM results in a high or low-risk ground truth. Under this hypothesis, it may also be beneficial to include events where the last available input CDM had low-risk but a high-risk ground truth. However, the number of samples available for this class of data was extremely small and testing indicated including this data made the model worse. In any case, it was surprising to observe strong model performance using such little data from the training set.

Finally, in training the fully-connected model, it would have been possible to avoid using the custom dataloader and allow full randomization of events as this model only utilizes the last available input CDM. However, brief experimentation showed better results using the dataloader compared to the more standard batching of a fully randomized dataset.

6.2 Competition metric flaws

As admitted by the competition designers, the competition metric suffers a few flaws from its emphasis on high-risk events and penalization of false negatives. Models are biased towards focusing on correctly predicting events with high-risk scores, at the cost of false positives. While the use of an F2 score moderates this tendency to some extent, there are no significant penalties from the MSE portion of the score. Furthermore, there is effectively no benefit to designing a model that accurately predicts a low-risk score; the competition loss does not apply penalties as long as a low-risk ground truth has a low-risk prediction because MSE is only calculated for high-risk ground truths. Perhaps in the real-life context, being cautious can be beneficial to avoid satellite collisions, but unnecessary maneuvers to collisions can waste limited resources and time. We observe that practically all of our models regardless of architecture prefer to increase recall while sacrificing precision.

6.3 Utility of time series

The surprising success of the fully-connected model which did not utilize the time-series nature of the CDMs against the other models indicates that this aspect of the data may not be useful in predicting a final risk value. Indeed, by analyzing the distribution of the data as was done in the Data oversampling section, events that have a risk score in the last available input CDM that is classified as low-risk tend to have a low-risk ground truth score, with a similar for high-risk samples. The strong performance of the naive forecast baseline further provides support for this observation. This raises the question of why this is the case. Naturally, because events driven by physics theory are time dependent, we expect a strong connection between sequential CDMs. However, it may be the case that collision events are too chaotic and for small time steps, this process is best described as a random walk. In any case, this observation provides an interesting insight for future model development.

6.4 Conclusion

In this paper, we explored creating a deep learning model to predict satellite collision risk using Conjunction Data Messages (CDMs). We framed the problem as a time-series prediction task by leveraging Kelvin's Collision Avoidance Challenge dataset. During our exploration we analyzed different model architectures including LSTMs, fully-connected networks, and time-series transformers.

Our findings highlight several key observations. First, the fully-connected model, despite ignoring the sequential nature of the data, outperformed more complex architectures such as LSTMs and transformers. This suggests that the inherent temporal dependencies in CDMs may not provide as much utility for final risk prediction as initially expected. Second, data imbalance posed a significant challenge, as high-risk events only made up a small fraction of the dataset. While oversampling helped mitigate this issue to some extent, achieving a balance between precision and recall remained difficult. Third, we observed that pre-processing and careful feature selection were crucial for enhancing model performance, reinforcing the importance of data exploration in machine learning tasks.

The project's primary objective was to develop models that could match or exceed the baselines obtained in the competition. While our models demonstrated competitive performance, achieving a loss that beat both competition baselines and placed 8th on the leaderboard, they also revealed the limitations of the dataset and evaluation metrics, such as an over emphasis on recall for high-risk events and the lack of penalties for false positives. These limitations highlight areas for improvement in future work, including the development of more balanced evaluation metrics and comprehensive datasets that better reflect real-world challenges.

7 Future Work

Over the next two weeks, we plan to continue fine-tuning the model and optimizing hyperparameters to minimize the competition loss metric. We also aim to experiment with additional model architectures and improve data preprocessing techniques to better address the significant class imbalance, potentially through synthetic data generation or by incorporating external data sources. Over the next two months, we intend to integrate our simulation framework to model covariance propagation through collision-mitigating maneuvers [9] and generate new training data. This dataset would enable us to train models not only to predict collision risks but also to calculate the associated risks after performing specific maneuvers, helping operators make more informed decisions. This was our stretch goal from the initial proposal, which we could not accomplish due to difficulty achieving strong results without this added complexity.

Looking further ahead, we aim to curate a dataset that models real-life CDMs with additional information such as true anomaly and absolute position, which would allow for more accurate modeling of satellite trajectories. Finally, we plan to experiment with traditional machine learning architectures, such as Random Forest and LightGBM, and compare their performance to deep learning-based models like LSTMs and transformers, with the goal of identifying the most effective approach for satellite collision avoidance.

References

[1] R. Abay. Learning from an imbalanced dataset of conjunction data messages. In *Collision Avoidance Competition Workshop*. ESA, 2021. URL https://conference.sdo.esoc.esa.int/proceedings/aicaw01/paper/3/aicaw01-paper3.pdf.

[2] S. Alfano and D. Oltrogge. Probability of collision: Valuation, variability, visualization, and validity. *Acta Astronautica*, 148:301–316, 2018. ISSN 0094-5765. doi: https://doi.org/10.1016/j.actaastro.2018.04.023. URL https://www.sciencedirect.com/science/article/pii/S0094576518304491.

[3] Consultative Committee for Space Data Systems (CCSDS). Mission Operations Services Concept. Technical report, Consultative Committee for Space Data Systems, Dec. 2010. URL https://public.ccsds.org/Pubs/508x0b1e2s.pdf. CCSDS 508.0-B-1. Blue Book. Issue 2. Available:
urlhttps://public.ccsds.org/Pubs/508x0b1e2s.pdf.

[4] ESA. How many space debris objects are currently in orbit? URL https://www.esa.int/Space_Safety/Clean_Space/How_many_space_debris_objects_are_currently_in_orbit.

[5] J. L. Foster, H. S. Estes, L. B. J. S. C. M. O. Directorate, and &. A. D. Lyndon B. Johnson Space Center Navigation, Control. A parametric analysis of orbital debris collision probability and maneuver rate for space vehicles, 1992. URL https://purl.stanford.edu/dg552pb6632.

[6] D. T. Hall. Expected collision rates for tracked satellites. *Journal of Spacecraft and Rockets*, 58(3):715–728, May 2021. ISSN 1533-6794. doi: 10.2514/1.a34919. URL http://dx.doi.org/10.2514/1.A34919.

[7] M. Kisantal, S. Sharma, T. H. Park, D. Izzo, M. Märtens, and S. D'Amico. Satellite pose estimation challenge: Dataset, competition design, and results. *IEEE Transactions on Aerospace and Electronic Systems*, 56(5):4083–4098, 2020. doi: 10.1109/TAES.2020.2989063.

[8] C. B. Office. Large Constellations of Low-Altitude Satellites: A Primer. Reports 58794, Congressional Budget Office, May 2023. URL https://ideas.repec.org/p/cbo/report/58794.html.

[9] C. Schiff. Adapting covariance propagation to account for the presence of modeled and unmodeled maneuvers. In *AAS/AIAA Astrodynamics Specialist Conference*, Keystone, CO, USA, Aug. 2006. URL https://ntrs.nasa.gov/api/citations/20070018023/downloads/20070018023.pdf. NASA Technical Report 20070018023. Available: https://ntrs.nasa.gov/api/citations/20070018023/downloads/20070018023.pdf.

[10] A. C. Team. Collision avoidance challenge data, 2019. URL https://kelvins.esa.int/collision-avoidance-challenge/data/.

[11] T. Uriot, D. Izzo, L. F. Simões, R. Abay, N. Einecke, S. Rebhan, J. Martinez-Heras, F. Letizia, J. Siminski, and K. Merz. Spacecraft collision avoidance challenge: Design and results of a machine learning competition. *Astrodynamics*, 6(2):121–140, Jun 2022. ISSN 2522-0098. doi: 10.1007/s42064-021-0101-5. URL https://doi.org/10.1007/s42064-021-0101-5.

A Appendix A: Link to Github repository

This is the link to the Github repo that contains the most recent versions of all the code that was written for this project.

https://github.com/lakewood999/cmsc472-project